ADVANCES IN *SOFT COMPUTING*

Mieczyslaw A. Kłopotek
Sławomir Wierzchoń · Krzystof Trojanowski
Editors

Intelligent Information Processing and Web Mining

Proceedings of the International IIS:
IIPWM'05 Conference
held in Gdansk, Poland,
June 13–16, 2005

Springer

Intelligent Information Processing
and Web Mining

Advances in Soft Computing

Editor-in-chief
Prof. Janusz Kacprzyk
Systems Research Institute
Polish Academy of Sciences
ul. Newelska 6
01-447 Warsaw, Poland
E-mail: kacprzyk@ibspan.waw.pl

Further books of this series can be found on our homepage: springeronline.com

Mieczysław Kłopotek, Maciej Michalewicz
and Sławomir T. Wierzchoń (Eds.)
Intelligent Information Systems
2001. ISBN 3-7908-1407-5

Antonio Di Nola and Giangiacomo Gerla (Eds.)
Lectures on Soft Computing and Fuzzy Logic
2001. ISBN 3-7908-1396-6

Tadeusz Trzaskalik and Jerzy Michnik (Eds.)
Multiple Objective and Goal Programming
2002. ISBN 3-7908-1409-1

James J. Buckley and Esfandiar Eslami
An Introduction to Fuzzy Logic and Fuzzy Sets
2002. ISBN 3-7908-1447-4

Ajith Abraham and Mario Köppen (Eds.)
Hybrid Information Systems
2002. ISBN 3-7908-1480-6

Przemysław Grzegorzewski, Olgierd Hryniewicz,
María ç . Gil (Eds.)
Soft Methods in Probability, Statistics
and Data Analysis
2002. ISBN 3-7908-1526-8

Lech Polkowski
Rough Sets
2002. ISBN 3-7908-1510-1

Mieczysław Kłopotek, Maciej Michalewicz
and Sławomir T. Wierzchoń (Eds.)
Intelligent Information Systems
2002. ISBN 3-7908-1509-8

Andrea Bonarini, Francesco Masulli
and Gabriella Pasi (Eds.)
Soft Computing Applications
2002. ISBN 3-7908-1544-6

Leszek Rutkowski, Janusz Kacprzyk (Eds.)
Neural Networks and Soft Computing
2003. ISBN 3-7908-0005-8

Jürgen Franke, Gholamreza Nakhaeizadeh,
Ingrid Renz (Eds.)
Text Mining
2003. ISBN 3-7908-0041-4

Tetsuzo Tanino,Tamaki Tanaka,
Masahiro Inuiguchi
Multi-Objective Programming and Goal
Programming
2003. ISBN 3-540-00653-2

Mieczysław Kłopotek, Sławomir T. Wierzchoń,
Krzysztof Trojanowski (Eds.)
Intelligent Information Processing
and Web Mining
2003. ISBN 3-540-00843-8

Ahmad Lotfi, Jonathan M. Garibaldi (Eds.)
Applications and Science in Soft-Computing
2004. ISBN 3-540-40856-8

Mieczysław Kłopotek, Sławomir T. Wierzchoń,
Krzysztof Trojanowski (Eds.)
Intellligent Information Processing
and Web Mining
2004. ISBN 3-540-21331-7

Miguel López-Díaz, María ç . Gil, Przemysław
Grzegorzewski, Olgierd Hryniewicz, Jonathan
Lawry
Soft Methodology and Random Information
Systems
2004. ISBN 3-540-22264-2

Kwang H. Lee
First Course on Fuzzy Theory and Applications
2005. ISBN 3-540-22988-4

Barbara Dunin-Kęplicz, Andrzej Jankowski,
Andrzej Skowron, Marcin Szczuka
Monitoring, Security, and Rescue Techniques in
Multi-Agent Systems
2005. ISBN 3-7908-23245-1

Marek Kurzyński, Edward Puchała,
Michał Woźniak, Andrzej Żołnierek
Computer Recognition Systems
2005. ISBN 3-540-25054-9

Mieczysław Kłopotek, Sławomir T. Wierzchoń,
Krzysztof Trojanowski (Eds.)
Intellligent Information Processing
and Web Mining
2005. ISBN 3-540-25056-5

Mieczysław A. Kłopotek
Sławomir T. Wierzchoń
Krzysztof Trojanowski (Eds.)

Intelligent Information Processing and Web Mining

Proceedings
of the International IIS: IIPWM '05 Conference
held in Gdansk, Poland, June 13-16, 2005

With 191 Figures
and 83 Tables

Springer

Prof. Dr. Mieczysław A. Kłopotek
Prof. Dr. Sławomir T. Wierzchoń
Dr. Krzysztof Trojanowski

Polish Academy of Sciences
Institute of Computer Sciences
ul. Ordona 21
01-237 Warszawa
Poland

Library of Congress Control Number: 2005922820

ISSN 1615-3871
ISBN-10 3-540-25056-5 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-25055-2 Springer Berlin Heidelberg New York

Springer is a part of Springer Science+Business Media
springeronline.com

Printed in Germany

Typesetting: Data conversion by the editors
Final processing by PTP-Berlin Protago-TEX-Production GmbH, Germany
Cover-Design: Erich Kirchner, Heidelberg
Printed on acid-free paper 62/3141/Yu – 5 4 3 2 1 0

Preface

The international conference Intelligent Information Processing and Web Mining IIS:IIPWM'05, organized in Gdańsk-Sobieszewo on 13–16th June, 2005, was a continuation of a long tradition of conferences on applications of Artificial Intelligence (AI) in Information Systems (IS), organized by the Institute of Computer Science of Polish Academy of Sciences in cooperation with other scientific and business institutions.

The Institute itself is deeply engaged in research both in AI and IS and many scientists view it as a leading institution both in fundamental and applied research in these areas in Poland. The originators of this conference series, Prof. M. Dąbrowski and Dr. M. Michalewicz had in 1992 a long-term goal of bringing together scientists and industry of different braches from Poland and abroad to achieve a creative synthesis. One can say that their dream has come to reality. Scientists from five continents made their submissions to this conference. A brief look at the affiliations makes international cooperation visible. The research papers have either a motivation in concrete applications or are off-springs of some practical requests. This volume presents the best papers carefully chosen from a large set of submissions (about 45%). At this point we would like to express our thanks to the members of Programme Committee for their excellent job. Also we are thankful to the organizers of the special sessions accompanying this conference: Jan Komorowski, Adam Przepiórkowski, Zbigniew W. Raś, Henryk Rybiński, Roman Świniarski, Alicja Wakulicz-Deja. But first of all we are deeply indebted the contributors to this volume and those whose papers were not qualified for publication for their hard research work that made this conference such an exciting event.

The conference was addressed especially to those who are active in Artificial Immune Systems (AIS) and other biologically motivated methods, Search Engines (SE) research, Computational Linguistics (CL) and Knowledge Discovery (KD). Frequently it is really hard to draw a border between these subdomains of AI and the formal assignment to the chapters of this book was a bit ambiguous. But this makes just the study of these papers more exciting. The submitted papers covered new computing paradigms, including, but not restricted to biologically motivated methods, DNA computing, advanced data analysis, new machine learning paradigms, reasoning technologies, natural language processing, novelty detection, new optimization technologies, applied data mining using statistical and non-standard approaches, technologies for very large text bases, uncertainty management, among others.

On behalf of the Program Committee and of the Organizing Committee we would like to thank all participants: computer scientists mathematicians, engineers, logicians and other interested researchers who found excitement

in advancing the area of intelligent systems. We hope that this volume of IIS:IIPWM'05 Proceeding will be a valuable reference work in your further research.

We would like to thank Dr. M. Woliński for his immense effort in resolving technical issues connected with the preparation of this volume.

Gdańsk, Poland,
June 2005

Mieczysław A. Kłopotek, Conference Co-Chair
Sławomir T. Wierzchoń, Conference Co-Chair
Krzysztof Trojanowski, Organizing Committee Chair

We would like to thank to the Programme Committee Members for their great job of evaluating the submissions:

- Witold Abramowicz (Poznan University of Economics, Poland)
- David Bell (Queen's University Belfast, UK)
- Peter J. Bentley (University College London, UK)
- Petr Berka (University of Economics Prague, Czech Republic)
- Leonard Bolc (Polish Academy of Science, Poland)
- Andrzej Czyżewski (Gdansk University of Technology, Poland)
- Piotr Dembiński (Polish Academy of Sciences, Poland)
- Włodzisław Duch (Nicholas Copernicus University, Poland)
- Tapio Elomaa (Tampere University of Technology, Finland)
- Floriana Esposito (Bary University, Italy)
- Jerzy W. Grzymała-Busse (University of Kansas, USA)
- Mohand-Saïd Hacid (Université Claude Bernard Lyon 1, France)
- Mirsad Hadzikadic (University of North Carolina at Charlotte, USA)
- Jan Hajič (Charles University, Czech Republic)
- Ray J. Hickey (University of Ulster, UK)
- Erhard Hinrichs (University of Tuebingen, Germany)
- Olgierd Hryniewicz (Polish Academy of Sciences, Poland)
- Janusz Kacprzyk (Polish Academy of Sciences, Poland)
- Samuel Kaski (University of Helsinki, Finland)
- Jan Komorowski (Uppsala University, Sweden)
- Józef Korbicz (University of Zielona Góra, Poland)
- Jacek Koronacki (Polish Academy of Sciences, Poland)
- Witold Kosiński (Polish-Japanese Institute of Information Technologies, Poland)
- Bożena Kostek (Gdansk University of Technology, Poland)
- Geert-Jan M. Kruijff (German Research Center for Artificial Intelligence (DFKI), Germany)
- Stan Matwin (University of Ottawa, Canada)
- Detmar Meurers (Ohio State University, USA)
- Maciej Michalewicz (NuTech Solutions Polska, Poland)
- Zbigniew Michalewicz (NuTech Solutions, USA)
- Ryszard S. Michalski (George Mason University, USA)
- Zdzisław Pawlak (Scientific Research Committee, Poland)
- James F. Peters (University of Manitoba, Canada)
- Adam Przepiórkowski (Polish Academy of Sciences, Poland)
- Zbigniew W. Raś (University of North Carolina at Charlotte, USA)
- Jan Rauch (University of Economics, Czech Republic)
- Gilbert Ritschard (University of Geneva, Switzerland)
- Henryk Rybiński (Warsaw University of Technology, Poland)
- Abdel-Badeeh M. Salem (Ain Shams University, Egypt)
- Kiril Simov (Bulgarian Academy of Science, Bulgaria)
- Andrzej Skowron (Warsaw University, Poland)

- Roman Świniarski (San Diego State University, USA)
- Ryszard Tadeusiewicz (University of Science and Technology, Poland)
- Jonathan Timmis (University of Kent, UK)
- Zygmunt Vetulani (Adam Mickiewicz University, Poland)
- Alicja Wakulicz-Deja (University of Silesia, Poland)
- Hui Wang (University of Ulster, UK)
- Jan Węglarz (Poznan University of Technology, Poland)
- Stefan Węgrzyn (Polish Academy of Sciences, Poland)
- Krzysztof Zieliński (University of Science and Technology, Poland)
- Djamel A. Zighed (Lumière Lyon 2 University, France)
- Jana Zvarová (EuroMISE Centre, Czech Republic)

We would like also to thank additional reviewers:

- Anna Feldman (Ohio State University, USA)
- Xiaofei Lu (Ohio State University, USA)
- Angelina Tzacheva (University of North Carolina at Charlotte, USA)

Table of contents

Part X. Invited Session: KDD and Facial Recognition

Part XI. Invited Session: Recent Developments in Bioinformatics

Part XII. Special Session: Artificial Immune Systems

Part I

Regular Sessions: Knowledge Discovery and Exploration

Feature Extraction by the SCoTLASS: An Illustrative Example

Anna Bartkowiak[1] and Nickolay T. Trendafilov[2]

[1] University of Wrocław, Inst. of Computer Science, Przesmyckiego 20, Wrocław PL–53–502, Poland,
[2] Bulgarian Academy of Sciences, Inst. of Mathematics and Informatics, 8 Acad. G. Bonchev Str, Sofia 1113, Bulgaria.

Abstract. Derivation of new features of observed variables has two important goals: reduction of dimensionality and de-noising. A desired property of the derived new features is their meaningful interpretation. The SCoTLASS method (Jolliffe, Trendafilov and Uddin, 2003) offers such possibility.

We explore the properties of the SCoTLASS method applied to the yeast genes data investigated in (Bartkowiak et al., 2003, 2004). All the derived features have really a simple meaningful structure: each new feature is spanned by two original variables belonging to the same block.

1 Introduction

Usually the available data contain observations in many variables, which are redundant, i.e. repeat partially the same information. Moreover, the gathered data contain also some noise. Quoting Kumar et al. [6]: 'While the majority of learning algorithms perform well on domains with relevant information, they degrade in adverse situations like: data with high noise content, ... , irrelevant or redundant information and non-linearity.' The quoted authors give further references to papers, describing the situation where the efficiency of the learning algorithms has decreased on domains with irrelevant and redundant variables. To circumvent such situation the literature suggest some data preprocessing – having as aim reduction of dimensionality and de-noising. The principal component analysis is a widely used technique serving this goal. Some other techniques serving this purpose are briefly reviewed in Kumar et. al. [6]. A desired property of the derived new features is their meaningful interpretation. The SCoTLASS method (Jolliffe, Trendafilov and Uddin, 2003) offers such possibility.

The aim of our paper is to explore the properties of the SCoTLASS method applied to a real data set: the yeast genome data [7,2]. The data concern $n = 3300$ yeast genes characterized by 13 variables. The variables are redundant and noisy; however they are not directly reducible (e.g. by regression analysis). The data were formerly investigated in Bartkowiak et al. [1]. Using classical principal components and neural networks, the authors got the result that 6 derived features explain about 88% of total variance,

however the extracted features contain still a some small amount of noise; a reasonable approximation by 6 probabilistic principal components explains only about 78% of total variance. Rotation varimax applied to the derived features revealed a relatively simple structure of the the derived variables; however not simple enough to our expectation based on our knowledge about the provenance of the original variables.

Therefore we were looking for other methods providing new features with a simple, parsimonious structure and – in the same time – meaningful in the context of the analyzed data. Such method proved to be the SCoTLASS ([5,8]).

In the following we describe shortly the SCoTLASS in Section 2. Results of application of that method to the yeast genome data are presented in Section 3. Finally, some summarizing remarks are given in Section 4. All the derived features have really a simple meaningful structure: each new feature is spanned by two original variables belonging to the same block of original variables.

2 The SCoTLASS method for finding components with simple structure

In this section we introduce the SCoTLASS method in a general framework. We present also the computational formulae used further in our analysis.

2.1 Introducing the SCoTLASS method

A standard approach in multivariate data analysis is to compose a linear combination of the original variables and transform in such a way the original multivariate problem to a sequence of univariate problems for the derived linear combinations. The standing follow-up question is the interpretation of the constructed linear combinations.

The standard answer is an orthogonal rotation of the loadings to achieve interpretation simplicity. The simplicity is usually defined as a function of the loadings to be optimized. The most popular VARIMAX simplicity measure maximizes the variance of the squared loadings. Ideally this is achieved if one of the loadings is 1 (or -1) and all the rest are 0s. In practice, one obtains few large loadings (about 1 in magnitude) and all the rest relatively small (about 0 in magnitude).

There are at least two problems when applying this approach to PCA:

1. the interpretation is not clear, because the 'small' loadings are not all close enough to 0, and thus one is uncertain about taking or not the corresponding variable into consideration;
2. after rotation the principal components are not principal components any more, i.e. the variance of the first rotated 'principal' component is not the largest among the variances of the rotated components.

The SCoTLASS approach is a new exploratory approach designed to overcome these two weaknesses of the "classical" rotation approach. The LASSO principal components keep their optimal properties: 1) they still explain large amount of the variability of the data; 2) the first LASSO component still has the largest variance, the second component-the second largest variance, etc, and the resulting loadings are really sparse, i.e. they are a great number of exact 0 loadings or at least very small in magnitude, and very few large loadings. This follows from the geometry of the SCoTLASS feasible set: it is an union of "poles" of the unit sphere (not the whole sphere as in PCA) and thus the loadings are forced to have either very small or very large values.

2.2 SCoTLASS – Performing the calculations

Let $\mathbf{x} = (x_1, \ldots, x_p)^T$ denote a data vector for one individual (one instance). For simplicity of presentation we assume, that the data are 0-centered and have unit variances for each variable k, $k = 1, \ldots, p$.

Let $\mathbf{R}$ denote the correlation matrix calculated from the analyzed data.

Principal component analysis (PCA, see e.g. [4]) seeks for linear combinations $z_k = \mathbf{a}_k^T \mathbf{x}$, $k = 1, \ldots, p$, which successively have maximum variance

$$\mathbf{a}_k^T \mathbf{R} \mathbf{a}_k, \tag{1}$$

subject to

$$\mathbf{a}_k^T \mathbf{a}_k = 1, \quad \text{and} \quad (\text{for } k \geq 2) \;\; \mathbf{a}_h^T \mathbf{a}_k = 0, \; h < k\,. \tag{2}$$

The matrix $\mathbf{A}_{p \times p} = (\mathbf{a}_1, \ldots, \mathbf{a}_p,)$ may be viewed as the matrix of loadings of the derived new variables $\mathbf{z} = (z_1, \ldots, z_p)^T$, called also the 'derived features'. An example of such a matrix $\mathbf{A}$ of size 13×9 is given in Table 1. One may see there, that in all columns the majority of the elements decidedly can not be considered as 0-valued; which implies that all the original variables are necessary to form the new features z_k, $k = 1, \ldots, 9$.

Our goal is to make this matrix simpler, i.e. to make more of its coefficients equal to zero. This can be achieved at the expense to add to the conditions expressed in eq. (1) and (2) additional constraints

$$\sum_{j=1}^{p} |a_{kj}| \leq t \tag{3}$$

for some tuning parameter t, where a_{kj} is the jth element of the kth vector $\mathbf{a}_k$, $(k = 1, \ldots, p)$.

The constraint given by eq. (3) was imposed by Jolliffe et al. [5]. They call the new technique SCoTLASS (Simplified Component Technique - LASSO).

They show also that

(a) for $t \geq \sqrt{p}$, we obtain the classical PCA;
(b) for $t < 1$, there is no solution; and
(c) for $t = 1$, we must have exactly one non-zero element a_{kj} for each k.

It can be seen that as t decreases from $\sqrt{p}$ to 1, we move away from the classical PCA solution towards the very parsimonious solution providing in each column only one decidedly pronounced coefficient.

Table 1. Components extracted using the tuning parameter tt = 3.6056 yielding classical principal components. Notice the small simplicity of each component

tt =	No. of the component								
3.6056	1	2	3	4	5	6	7	8	9
ang1	-0.05	-0.39	-0.35	0.07	-0.16	0.53	-0.00	0.15	-0.06
x1	0.35	0.31	-0.07	0.01	-0.14	-0.43	0.10	-0.07	0.13
y1	0.24	-0.27	-0.50	0.13	-0.22	0.07	0.09	-0.10	0.08
rho1	0.43	-0.03	-0.33	0.09	-0.18	-0.21	0.17	-0.13	0.14
ang2	-0.00	0.31	-0.14	0.62	0.29	0.13	-0.34	0.24	0.04
x2	0.29	0.33	0.02	0.33	0.13	0.31	0.09	-0.24	-0.06
y2	-0.36	0.06	-0.32	0.34	0.03	-0.30	0.10	0.04	-0.14
rho2	0.41	0.11	0.23	-0.09	0.09	0.43	0.07	-0.17	-0.04
ang3	-0.08	0.27	-0.38	-0.41	0.42	0.15	0.19	0.26	0.54
x3	-0.10	0.42	0.04	-0.00	-0.47	0.18	0.50	0.45	-0.26
y3	-0.13	0.30	-0.42	-0.34	0.16	0.05	-0.10	-0.41	-0.59
rho3	0.23	-0.34	0.05	0.08	0.58	-0.17	0.50	0.24	-0.37
leng	0.41	0.02	-0.11	-0.25	-0.04	-0.11	-0.52	0.55	-0.29
	Simplicity per Component:								
	0.06	0.04	0.08	0.14	0.13	0.09	0.14	0.10	0.16
	Cumulative Variance in % :								
	29.97	55.20	68.29	77.84	83.37	88.44	91.43	94.01	96.34

An example of a corresponding matrix $\mathbf{A}_{13\times 9}$ obtained for the tuning parameter $tt = 1.5$ is given in Table 2. One may see there that indeed, most of the loadings exhibit now values equal to 0, generally in each column only two coefficients are large, the remaining ones being very near or exactly equal to zero. This means that now each of the new features is composed mainly from two original variables.

Table 2. Components extracted using the tuning parameter tt = 1.5. A big simplicity of the coefficients may be observed - which is reflected in the increase of the simplicity index, as compared with that shown in Table 1. However, the accounted percentage of variance has decreased

tt=	No. of the component								
1.5	1	2	3	4	5	6	7	8	9
ang1	0.00	-0.00	0.00	**-0.71**	-0.00	-0.05	0.11	0.00	0.00
x1	0.00	**0.63**	0.00	0.00	-0.00	-0.00	0.00	0.00	0.00
y1	-0.00	0.00	0.00	**-0.70**	-0.00	-0.00	-0.11	0.10	-0.00
rho1	0.00	**0.77**	-0.00	-0.00	0.00	0.00	0.11	-0.00	0.00
ang2	0.00	0.00	0.00	0.00	**-0.78**	0.01	0.07	-0.01	0.00
x2	-0.08	0.00	0.00	0.00	**-0.63**	0.00	-0.09	-0.00	-0.00
y2	**0.68**	-0.00	-0.00	-0.00	-0.08	0.00	-0.00	0.00	0.00
rho2	**-0.73**	0.00	0.00	0.00	-0.00	-0.00	0.01	0.00	-0.00
ang3	0.00	0.00	**0.70**	0.00	0.00	0.11	0.00	0.05	**0.71**
x3	0.00	0.00	0.00	0.08	-0.01	**-0.44**	0.01	**0.89**	0.00
y3	0.00	0.00	**0.71**	0.00	0.00	0.00	0.00	0.00	**-0.70**
rho3	-0.00	0.00	-0.08	-0.00	0.00	**0.89**	0.01	**0.43**	-0.08
leng	-0.00	0.09	-0.00	-0.00	-0.00	0.00	**-0.98**	0.00	-0.00
	Simplicity per Component:								
	0.42	0.43	0.42	0.42	0.44	0.59	0.83	0.60	0.42
	Cumulative Variance in % :								
	14.34	28.31	41.71	54.81	66.76	77.06	84.42	88.25	90.78

To assess the quality of the representation two statistics are defined. One of them shows the accounted variance, the other – the simplicity of the derived loadings.

The **accounted variance** is defined as follows: Let $\mathbf{C}_{p\times p}$ denote the covariance matrix of the derived new features $z_1, \ldots, z_p$. The matrix $\mathbf{C}$ is obtained as:

$$\mathbf{C} = \mathbf{A}^T\mathbf{R}\mathbf{A}. \tag{4}$$

Let $\mathbf{ss} = diag(\mathbf{C}) = (ss_1, \ldots, ss_p)$. Then the **percentage of variance** of the new features, explained by the first k components, is defined as

$$SS_k = \frac{1}{p}(\sum_{j=1}^{k} ss_k) \cdot 100.$$

The **Simplicity index** is defined separately for each column vector $\mathbf{a}_k$ of the loadings. We use here a statistic appearing in the varimax criterion. Let $\mathbf{b}$ denote squared coefficients appearing in the vector $\mathbf{a}_k$. Then the simplicity index v_k for the kth column of loadings is defined as:

$$v_k = \frac{1}{p} \cdot (p \cdot \sum {b_i}^2 - (\sum b_i)^2), \quad k = 1, \ldots, p.$$

The index v_k evaluated for the column $\mathbf{a}_k$ takes the minimum vakue 0 when all loadings have equal absolute values, and takes the value $(p-1)/p$ when only one loading is non-zero.

3 Simple SCoTLASS components derived for the yeast genome data

3.1 Details of the calculations

The calculations were performed using the yeast genome data (Set2000) composed in the Dept. of Genomics, Inst. of Genetics and Microbiology, University of Wroclaw. It contains data for 3300 yeast genes known in the year 2000. The detailed description of the data may be found at the URL 'Smorfland" [7]. There are 12 variables appearing in 3 blocks and an additional variable 'length' denoting length of the gene, counted in basic codons A, C, G, T. Each gene was visualized in a *spider–plot* and each of the mentioned blocks contains variables describing the corresponding 'leg' of the 'spider' visible in that plot (see Cebrat et al. [3]).

The ordering of the variables appearing in Table 1 and Table 2 corresponds to subsequent blocks. The variables *ang1, x1, y1, rho1* constitute block1, the variables *ang2, x2, y2, rho2* – block2, and the variables *ang3, x3, y3, rho3* – block3 respectively.

For computations of the SCoTLASS components we have used original Matlab software developed by the second author. The description of the implemented algorithm may be found in a forthcoming paper by Trendafilov and Jolliffe [8]. In the following we denote the tuning parameter by tt. The following values of tt are used:

$$tt = \sqrt{13} \approx 3.6055,\ 2.25,\ 2.0,\ 1.75,\ 1.3.$$

3.2 Obtained results

For $tt = \sqrt{13}$ the results are identical with those yielded by the classical PCA [5]. With decreasing values of tt the simplicity index is growing step by step. Exemplary results for $tt = \sqrt{13}$ and $tt = 1.5$ – displaying the loading matrices for $k = 1, \ldots, 9$ components, together with percentages of explained variances and simplicity indices – are shown in Table 1 and Table 2.

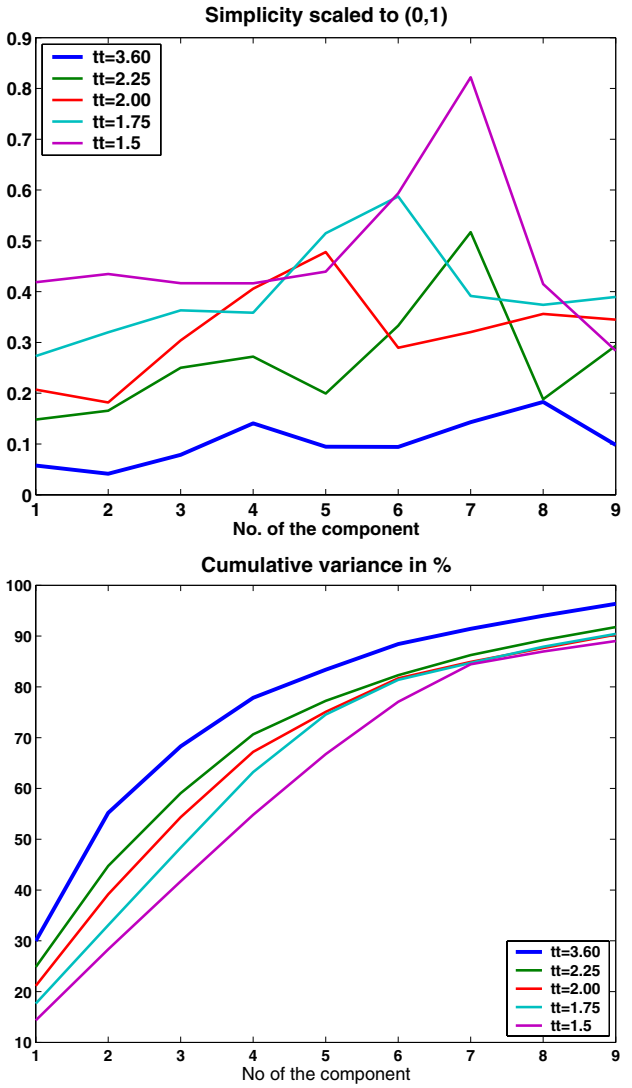


Fig. 1. Simplicity index, scaled to (0,1) (**top**) and Cumulated variance (**bottom**) evaluated for nine sequentially extracted components using the SCoTLASS algorithm. Variants when using the tuning parameters tt = 3.605, 2.25, 2.00, 1.75, 1.5 are shown. For tt=$\sqrt{(}p) = 3.605$ we obtain the classical PCs indicated by the thick line

The explained variances and simplicity indices for all the investigated values of tt are shown in Figure 1 and Figure 2. One may notice there the increase of the simplicity index when moving with the tuning parameter away from the value $tt = \sqrt{13}$ in the direction of the value '1'. With decreasing tt

more and more coefficients in the loadings matrix become equal to zero. This happens at the cost of explained variance.

Since the algorithm has random starts, we repeated the runs several times. The work of the algorithm proceeded fast, and the convergence was achieved for almost the same values of the sought statistics. Only for $tt = 1.3$ we obtained one run yielding 1 unusually big loading in the first component.

The simple structure discovered by the SCoTLASS algorithm is in amazing correspondence with the blocks corresponding to the 'legs' of the spider-plots. This has a deep meaning: the genetic code of the yeast genes appears in triplets: and the three 'legs' of the spider-plots correspond to the 3 places in the triplet-codons. Thus, the SCoTLASS by discovering the simple structure of the variables has discovered in an intermediary way also the ternary structure of the genetic code in the yeast genome.

4 Discussion and closing remarks

The SCotLASS has proven to be a very efficient method to discover meaningful structure in the yeast genome data.

A further problem, not considered in the paper, is the interdependence between the blocks of the variables. It is known among the geneticists that the first block of variables is the most important one when coding of the genetic information; while the third block has the least importance concerning that matter. May be something could be deduced analyzing the interdependencies among the derived simplified components. We have not elaborated that problem.

Still another question, asked by one of the referees, is: What happens, when there are problems with hundreds, not tens, of attributes. To our opinion, this is really a tough problem. High dimensional data are very specific ('the curse of dimensionality', 'the empty space' phenomenon). Generally, today it is believed that for such data special algorithms should be designed, or – at least – the known and currently used methods should be specially adapted. Some considerations on that topic, with suggestions on research directions, may be found in a recent paper by Verleysen et al. [9]. They present some weird facts about high-dimensional space and consequences for neural network learning. They emphasize the local models and preprocessing by non-linear projections.

To our opinion, a very large number of variables is a serious numerical problem. For such applications (typical in gene research) faster versions of the SCoTLASS algorithm must be devised or entirely new approximation of the problem as a whole should be undertaken.

Acknowledgements

The authors are deeply thankful to Prof. Stanisław Cebrat, chair of the Department of Genomics, Institute of Genetics and Microbiology, Wrocław University, for lending the yeast DNA Set2000.

The authors are also expressing thanks to two anonymous referees for their stimulating comments.

References

1. Bartkowiak, A., Cebrat, S., and Mackiewicz, P. (2004) Probabilistic PCA and neural networks in search of representative features for some yeast genome data. In: T. Burczyński, W. Cholewa and W. Moczulski (Eds), Recent Developments in Artificial Intelligence Methods, AI–METH Series, Gliwice, 21–26.
2. Bartkowiak, A., Szustalewicz, A., Cebrat, S., Mackiewicz, P. (2003) Kohonen's self-organizing maps as applied to graphical visualization of some yeast DNA data. Biometrical Letters, **40**, 2003, No. 2, 37–56.
3. Cebrat, S., Mackiewicz, P., Dudek, M.R. (1998) The role of the genetic code in generating new coding sequences inside existing genes. Biosystems, **45**, No 2, 165–176.
4. Jolliffe, I.T. (2002) Principal Components Analysis, 2nd Edition. Springer, New York, Berlin.
5. Jolliffe, I.T., Trendafilov, N.T., Uddin, M. (2003) A modified principal component technique based on the LASSO. J. of Computational and Graphical Statistics, **12**, Nb. 3, 531-547.
6. Kumar R., Jayaraman V.K., Kulkarni B.D. (2005) An SVM classifier incorporating simultaneous noise reduction and feature selection: illustrative case examples. Pattern Recognition **38**, No. 1, 41–49.
7. Smorfland: http://smorfland.microb.uni.wroc.pl/
8. Trendafilov, N.T., Joliffe, I.T. (2005) Projected gradient approach to the numerical solution of the SCoTLASS. To appear in Computational Statistics and Data Analysis.
9. Verleysen M., François D., Simon G., Wertz V. (2003) On the effects of dimensionality on data analysis with neural networks. In: J.Mira (Ed), IWANN 2003, LNCS 2687, 105–112.

Rule Induction for Click-Stream Analysis: Set Covering and Compositional Approach

Petr Berka[1], Vladimír Laš[1], and Tomáš Kočka[2,1]

[1] University of Economics, W. Churchill Sq. 4, Prague, Czech Republic
[2] Adastra, Benesovska 10, Prague, Czech Republic

Abstract. We present a set covering algorithm and a compositional algorithm to describe sequences of www pages visits in click-stream data. The set covering algorithm utilizes the approach of rule specialization like the well known CN2 algorithm, the compositional algorithm is based on our original KEX algorithm, however both algorithms deal with sequences of events (visited pages) instead of sets of attribute-value pairs. The learned rules can be used to predict next page to be viewed by a user or to describe the most typical paths of www pages visitors and the dependencies among the www pages. We have successfully used both algorithms on real data from an internet shop and we mined useful information from the data.

1 Introduction

According to the W3C Web Characterization Activity, click-stream is a sequential series of page view (displays on user's browser at one time) requests. A user session is the click-stream of page views for a single user across the entire web, a server session is a click-stream in a user session for a particular web site. A set of server sessions (visits) is the necessary input for web usage mining tools.

We can categorize web mining into three areas: web content mining, web structure mining and web usage mining [9]. Web usage mining mines the data derived from interactions of the users while browsing the web. The web usage data includes the data from web server access logs, proxy server logs, browser logs, user profiles, registration data, cookies, user queries etc. A web server log is an important source for performing web usage mining because it explicitly records the browsing behavior of site visitors. The typical problem (solved in the data preprocessing step) is thus distinguishing among unique users, server sessions episodes etc.

Web usage mining focuses on techniques that could predict user behavior while the user interacts with the web. The applications of web usage mining could be classified into two main categories: (1) learning user profile or user modeling, and (2) learning user navigation patterns [6]. The methods used for web usage mining are (descriptive) statistical analysis, association rules (to relate pages that are often referenced together), clustering (to build usage clusters or page clusters), classification (to create user profiles), sequential pattern discovery (to find inter-session patterns such that the presence of a

set of page views is followed by another set of page views in a time-ordered set of episodes), or dependency modeling (to capture significant dependencies among various variables in the web domain) [8]. Some systems has already been developed for this area: WebSIFT (that uses clustering, statistical analysis or association rules) [4], WUM (that looks for association rules using some extension of SQL) [7], or WebLogMiner (that combines OLAP and KDD) [10].

The algorithms described in this paper contribute to the area of sequential pattern discovery by learning decision rules to predict next page the user will visit based on his session history (pages visited just before). We describe the algorithms in section 2 and give some experimental results obtained on real web log data of an internet shop in section 3.

2 Learning Rules

2.1 Classical rule learning algorithms

Decision rules in the form

$$Ant \Rightarrow Class$$

where *Ant* (antecedent, condition) is a conjunction of values of input attributes (called categories or selectors) and *Class* is a category of class attribute C, are one of most popular formalisms how to express classification models learned from data.

The commonly used approach to learning decision rules is the *set covering approach* also called "separate and conquer". The basic idea of this approach is to create a rule that covers some examples of a given class and remove these examples form the training set. This is repeated for all examples not covered so far as shown in Fig. 1. There are two basic ways how to create a single rule (step 1 of the algorithm):

1. by rule generalization, i.e. by removing categories from antecedent of a potential rule (starting from a rule with categories of all input attributes in the antecedent) — this method is used in the AQ algorithms by Michalski (see e.g. [5]).
2. by rule specialization, i.e. by adding categories to the antecedent of a potential rule (starting from a rule with empty antecedent) — this method is used e.g. in CN2 [3] or CN4 [2].

The other way how to create decision rules is the *compositional approach*. In this approach the covered examples are not removed during learning, so an example can be covered with more rules. Thus more rules can be used during classification. In compositional approach, all applicable rules are used and their particular contributions to classification are combined into the final decision. To do this, some numerical value is usually added to the rule, the simplest one is the rule confidence (also called validity) defined as $n(Ant \wedge$

$Class)/n(Ant)$, where $n(Ant \wedge Class)$ is the number of examples that match both Ant and $Class$ and $n(Ant)$ is the number of examples that match Ant in the data. Fig. 2 shows an simplified version of the KEX algorithm, we developed in the 90th; this system deploys the compositional approach in a generate-test cycle and uses weight w (based on validity) to express uncertainty of a rule [1]. Let us mention, that in some broader sense, naive bayesian classifier follows the compositional approach as well.

Set Covering algorithm

1. find a rule that covers some positive examples and no negative example of a given class (concept)
2. remove covered examples from the training set D_{TR}
3. if D_{TR} contains some positive examples (not covered so far) goto 1, else end

Fig. 1. Set covering algorithm for two class problems

Compositional algorithm

1. add empty rule to the rule set KB
2. repeat
   2.1 find by rule specialization a rule $Ant \Rightarrow Class$ that fulfils the user given criteria on lengths and validity
   2.2 if this rule significantly improves the set of rules KB build so far (we test using the χ^2 test the difference between the rule validity and the result of classification of an example covered by Ant) then add the rule to KB

Fig. 2. Simplified version of the KEX rule learning algorithm

2.2 Rule learning algorithms for click-streams

We follow the rule learning approach based on rule specialization in our algorithm as well. The main difference to conventional rule learning algorithms is due to the fact that instead of unordered set of categories we deal with an ordered sequence of page views (pages for short). So we are looking for rules in the form

$$Ant \Rightarrow page(p)$$

where Ant is a sequence of pages, $page$ is a page view that directly follows the sequence Ant, and p is the validity of the rule

$$p = \frac{n(Ant//page)}{n(Ant}.$$

In the formula above we denote the number the occurrences of a sequence in the data by $n(sequence)$ and a concatenation of two sequences $s1$ and $s2$ by $s1//s2$.

The main idea of our *set covering* algorithm is to add (for a particular page to be predicted) rules of growing length of *Ant* — we thus create rules by rule specialization. We check each rule against its*generalization* created so far. Adding a new rule to the model is determined by χ^2 test that compares the validity of these two rules. If the rule in question is added to the model, its *generalization is updated* by re-computing the validity by ignoring (removing) sequences that are covered by the newly added rule.

The main idea of our *compositional* algorithm is to add (for a particular page to be predicted) rules of growing length of Ant — we thus create rules by rule specialization. We check each rule against the *results of classification* done by all rules created so far. Adding a new rule to the model is determined by χ^2 test that compares the validity of the rule to be added with the weight of class (predicted page) inferred during classification for a sequence *Ant*. The weight of a class is computed according to the formula

$$w_1 \oplus w_2 = \frac{w_1 \times w_2}{w_1 \times w_2 + (1 - w_1) \times (1 - w_2)}.$$

As we assume that most relevant for prediction of occurrence of page are pages that are closest to page, the specialization of the rule A$Ant \Rightarrow page$ is done by adding a new page to the beginning of the sequence *Ant*. Analogously, a generalization of the rule A$Ant \Rightarrow page$ is done by removing a page from the beginning of the sequence *Ant*.

The set covering algorithm is shown in Fig. 3, the compositional algorithm is shown in Fig. 4. Both algorithms share the user given input parameters l_{max} (max. length of the sequence *Ant*), n_{min} (min. relative frequency of a page — this parameter set to zero will enable to create a rule that page Y never follows after page X). The parameter l_{max} is also used for data preprocessing; we transform each server session of arbitrary length into a set of episodes of length $l_{max} + 1$ using a sliding window. So e.g. the two sessions

```
A,B,E
A,B,C,E,D
```

will be for $l_{max} = 2$ transformed into following set of episodes

```
O,O,A
O,A,B
A,B,E
O,O,A
O,A,B
A,B,C
B,C,E
C,E,D
```

Initialization
for each page page occurring in the data
1. compute its relative frequency in the data as P = (no. of occurrences of *page* in the input episodes)/(no. of all input episodes)
2. if $P \geq n_{min}$
   2.1 add $default \Rightarrow page$ into the list of rules $Rules$
   2.2 add *page* into list of pages $Pages$
   2.3 add $default \Rightarrow page$ into list of implications $Impl$

Main loop
while Impl not empty do
1. take first rule $Ant \Rightarrow page$ from $Impl$
2. if length of $Ant < l_{max}$ then
   2.1 for each page *pp* from $Pages$
      2.1.1 find the most specific generalization of the rule $pp//Ant \Rightarrow page$ in $Rules$ (denote it $AntX \Rightarrow page$)
      2.1.2 compare (using chi2 test) the validity of rules $pp//Ant \Rightarrow page$ and $AntX \Rightarrow page$
   2.2 from all created rules $pp//Ant \Rightarrow page$ select the one with the most significant difference in validity (denote this rule $pp_{best}//Ant \Rightarrow page$)
   2.3 if $pp_{best}//Ant \Rightarrow page$ significantly at a given significance level differs from $AntX \Rightarrow page$ then
      2.3.1 add rule $pp_{best}//Ant \Rightarrow page$ to $Rules$ and $Impl$
      2.3.2 re-compute the validity of rule $AntX \Rightarrow page$ by taking into account only episodes containing $AntX$ and not containing Ant
      2.3.3 recursively update Rules (i.e. find the most specific generalization of $AntX \Rightarrow page$, compare this generalization with $AntX \Rightarrow page$, remove $AntX \Rightarrow page$ from $Rules$ if the difference is not significant etc.)
3. remove $Ant \Rightarrow page$ from $Impl$

Fig. 3. The set covering rule learning algorithm for click-stream analysis

The system is implemented in the Borland's Delphi (version 7.0) for Windows (W-98 — W-XP). The minimal configuration of the computer is not set; it's helpful to use computers with processor at least 600 MHz for more extensive analysis. The system doesn't use more than quadruple of the size of an input file in the memory.

For the set covering algorithm, the main loop is implemented as a recursive function that is called for particular rules. At the beginning, the function generates specializations, after it modifies frequency of actual rule and to the end it evaluates the chi-square test for actual rule. The main loop is called from the initialization with the subsidiary rule `* => *` which assures that all rules `default => page` will be added to the result. For the compositional algorithm, there is no difference between the logic of the algorithm and its

implementation — the algorithm is implemented almost exactly as you could see in the Fig 4. The algorithms for searching rules run in second threads so found rules could be showed immediately in the core thread of the application — user could stop the algorithm when its continuation isn't needful. The user could restrict the space of searching by set rules that will not be extended more. Pages that don't have required relative frequency are added to a set `OTHERS` and the system works with this set as with single page. Found rules can be saved to the file (and loaded back) or exported to Excel. The application enables also to predict a next page in a sequence, the input to the prediction part can be either from keyboard or from a file.

Initialization
for each page page occurring in the data
1. compute its relative frequency in the data as P = (no. of occurrences of *page* in the input episodes)/(no. of all input episodes)
2. if $P \geq n_{min}$
   2.1 add $default \Rightarrow page$ into the list of rules $Rules$
   2.2 add *page* into list of pages $Pages$
   2.3 add $default \Rightarrow page$ into list of implications $Impl$

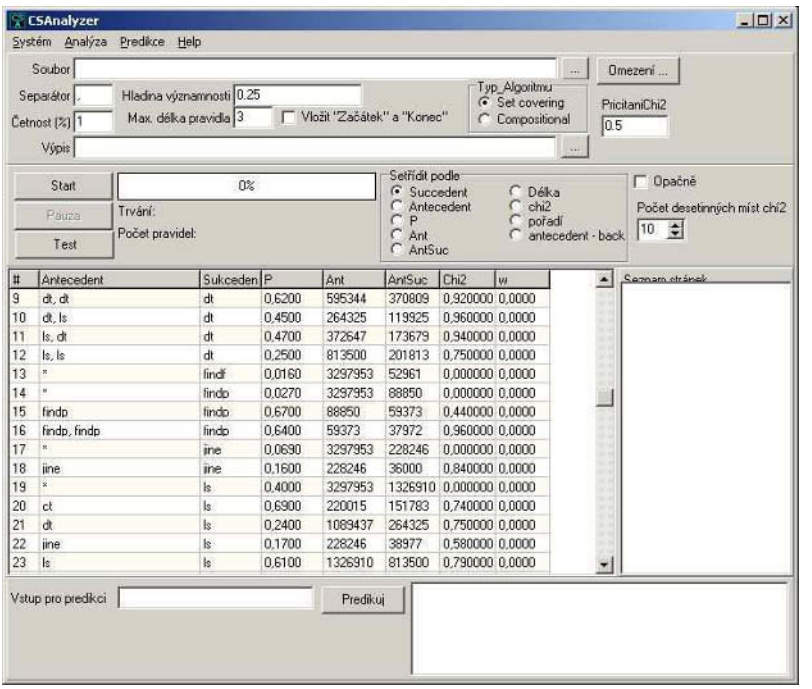
Main loop
while Impl not empty do
1. take first rule $Ant \Rightarrow page$ from $Impl$
2. if length of $Ant < l_{max}$ then
   2.1 for each page pp from $Pages$
      2.1.1 create the rule $pp//Ant \Rightarrow page$ and compute its validity p
      2.1.2 from all rules in $Rules$ that cover the sequence $pp//Ant$ compute the resulting weight for page as $w^{\oplus}(pp//Ant)$
      2.1.3 compare (using chi2 test) p and $w^{\oplus}(pp//Ant)$
      2.1.4 if p significantly differs from $w^{\oplus}(pp//Ant)$ then
         2.1.4.1 compute w from the formula $w \oplus w^{\oplus}(ppAnt) = p$
         2.1.4.2 add rule $pp//Ant \Rightarrow page(w)$ to $Rules$ and $Impl$
      2.1.5 add $pp//Ant \Rightarrow page(w)$ to $Impl$
   2.2 remove $Ant \Rightarrow page$ from $Impl$

Fig. 4. The compositional rule learning algorithm for click-stream analysis

3 Experiments

We tested our algorithms on a real data obtained from one Czech internet shop. The log file (about 3 millions of records — the traffic of 24 days) contained the usual information: time, IP address, page request and referee. In addition to this, the log data contained also a generated session ID so the identification of users was relatively easy — we treated as sequence of

pages with the same ID as a visit of one user . An example of records in the log file is shown in Fig. 6. The log file allowed us to identify two types of information about the visited page: page type and page content. By page type we understand information related to a general structure of the internet shop (detail of a product, shopping chart, product comparison etc), by page content we understand the product (or its category) offered on the page. These two points of view enable us to perform two different types of analyses: analysis of product preferences and analysis of shopping behavior.



#	Antecedent	Sukceden	P	Ant	AntSuc	Chi2	w
9	dt, dt	dt	0,6200	595344	370809	0,920000	0,0000
10	dt, ls	dt	0,4500	264325	119925	0,960000	0,0000
11	ls, dt	dt	0,4700	372647	173679	0,940000	0,0000
12	ls, ls	dt	0,2500	813500	201813	0,750000	0,0000
13	*	findf	0,0160	3297953	52961	0,000000	0,0000
14	*	findp	0,0270	3297953	88850	0,000000	0,0000
15	findp	findp	0,6700	88850	59373	0,440000	0,0000
16	findp, findp	findp	0,6400	59373	37972	0,960000	0,0000
17	*	jine	0,0690	3297953	228246	0,000000	0,0000
18	jine	jine	0,1600	228246	36000	0,840000	0,0000
19	*	ls	0,4000	3297953	1326910	0,000000	0,0000
20	ct	ls	0,6900	220015	151783	0,740000	0,0000
21	dt	ls	0,2400	1089437	264325	0,750000	0,0000
22	jine	ls	0,1700	228246	38977	0,580000	0,0000
23	ls	ls	0,6100	1326910	813500	0,790000	0,0000

Fig. 5. Screenshot of the system

```
unix time; IP address; session ID; page request; referee
1074589200;193.179.144.2;1993441e8a0a4d7a4407ed9554b64ed1;/dp/?id=124;www.google.cz;
1074589201;194.213.35.234;3995b2c0599f1782e2b40582823b1c94;/dp/?id=182;
1074589202;194.138.39.56;2fd3213f2edaf82b27562d28a2a747aa;/;http://www.seznam.cz;
1074589233;193.179.144.2;1993441e8a0a4d7a4407ed9554b64ed1;/dp/?id=148;/dp/?id=124;
1074589245;193.179.144.2;1993441e8a0a4d7a4407ed9554b64ed1;/sb/;/dp/?id=148;
1074589248;194.138.39.56;2fd3213f2edaf82b27562d28a2a747aa;/contacts/; /;
1074589290;193.179.144.2;1993441e8a0a4d7a4407ed9554b64ed1;/sb/;/sb/;
```

Fig. 6. Part of the web log

In the first step of data preprocessing we identified particular users by the session ID and we created a file containing sequences of visited pages for each user. So for the log file shown in Fig. 6, we created the page-type sequence (session)

```
start, dp, dp, sb, sb, end
```

and the page-content sequence (session)

```
start, 124, 148, end
```

for the user `1993441e8a0a4d7a4407ed9554b64ed1`. In this example, `dp` denotes "detail of product", `sb` denotes "shopping basket", `124` denotes "loudspeaker " and `148` denotes "DVD player". Note, that we added two pages to each sequence found in the data, the start page and the end page.

In the second step we excluded sessions of length 1. This is a general recommendation as sessions of length 1 are usually created by web bots crawling the web space and collecting pages. Moreover, as we are interested in prediction next page in a click-stream, we need sequences of length 2 and more. This reduces the number of sessions (sequences) to 200 000; the average number of visits by one user (the average length of session) was 16; the median was 8 and modus 2.

In the third step of data preprocessing, we created 2 basic files that entered to the analyses. In the first one, there were saved sequences of page type regardless of the product offered on this page. In the second one, there were saved sequences of page content (sequences of products) regardless of the page type (pages without products were excluded from these sequences). During this step, the products were divided to 30 categories.

The modeling consisted in searching for rules using the algorithm described in section 2. The algorithm for searching was running repeatedly with various input parameters.

The first set of experiments was performed for the sequences of page types. Among the obtained results we can find the rule

```
dp, sb -> sb (Ant: 5174; AntSuc: 4801; P: 93%)
```

that covers the session of the user `1993441e8a0a4d7a4407ed9554b64ed1` (see above). Another interesting rules were e.g.

```
ct -> end (Ant: 5502; AntSuc: 1759; P: 0.32)
faq -> help (Ant: 594; AntSuc: 127; P: 0.21).
```

In the listing above, `ct` stands for "contact", `Ant` stands for $||Ant||$ and `AntSuc` stands for $||Ant//Suc||$.

The second set of experiments was performed for the sequences of products (page contents). For the request of the data provider, we generated all rules of length 2 for the sequences of products. From these rules transitions between products were seen very well. Examples of such rules are:

```
loud-speakers -> video + DVD (Ant: 14840, AntSuc: 3785, P:
                              0.26)
  data cables -> telephones (Ant: 2560, AntSuc: 565, P: 0.22)
PC peripheries -> telephones (Ant: 8671, AntSuc: 1823, P: 0.21)
```

The models obtained in both sets of experiments can directly be used to predict the behavior of an user. So e.g. for a sequence of pages `dp, sb` the system will predict `sb` as the next page, and for the sequence `loud-speakers` the system will predict `video + DVD`.

We have run our algorithm repeatedly for both page sequences (first set of experiments) and product sequences (second set of experiments). but only about 20% for product sequences. The parameters that mostly contribute to improvement of the accuracy were the significance level and the frequency. On the contrary, the max. length of a sequence (l_{max}) that mostly affects the number of found rules doesn't improve the accuracy.

When looking at the differences between the set covering and compositional algorithms, we can observe different trade off between comprehensibility and accuracy: the set covering algorithm offers higher accuracy but lower comprehensibility (larger number of rules) than the compositional algorithm (see Tab. 1). The default accuracy refers to the accuracy of the "zero rule" model, that always predicts the most frequent page. In all experiments we exceeded this "base line".

Table 1. Empirical comparison of algorithms

dataset	default	set covering		compositional	
	accuracy	no. rules	accuracy	no.rules	accuracy
page sequences	0.40	771	0.63	35	0.55
product sequences	0.14	4132	0.2	206	0.17

4 Conclusions

We present two new algorithms to learn decision rules for web usage mining, set-covering and compositional one. Although we developed our algorithms within a project of click-stream analysis, it can be used more generally, to predict occurrence of an event in a sequence of any types of events (e.g. transactions on banking accounts, or network intrusion). Our experiments with a real data of an internet shop showed usefulness of the proposed approach. In our future work, we plan to perform a thorough comparison of both approaches.

5 Acknowledgements

The work presented in the paper is supported by the grant GACR 201/03/1318.

References

1. Berka, P., Ivánek, J. (1994) Automated knowledge acquisition for PROSPECTOR-like expert systems. In. (Bergadano, de Raedt eds.) Proc. ECML'94, Springer 1994, 339–342
2. Bruha, I., Kočkov, S. (1994) A support for decision making: Cost-sensitive learning system. Artificial Intelligence in Medicine, 6, 67–82
3. Clark, P., Niblett, T. (1989) The CN2 induction algorithm. Machine Learning, 3, 261–283
4. Cooley, R., Tan, P. N., Srivastava, J. (1999) Discovery of interesting usage patterns from web data. Tech.Rep. TR 99-022, Univ. of Minnesota
5. Kaufman, K. A., Michalski, R. S. (1999) Learning from inconsistent and noisy data: The AQ18 approach. In: Proc 11th Int. Symposium on Methodologies for Intelligent Systems
6. Kosala, R., Blockeel, H. (2000) Web Mining Research: A Survey. SIGKDD Explorations, Vol. 2 Issue 1
7. Spiliopoulou, M., Faulstich, L. (1999) WUM: A tool for web utilization analysis. In Proc. EDBT Workshop WebDB'98, Springer LNCS 1590
8. Srivastava, J., Cooley, R., Deshpande, M., Tan, P. N. (2000) Web Usage Mining: Discovery and Applications of Usage Patterns from Web Data. SIGKDD Explorations, Vol. 1 Issue 2
9. Zaiane, O., Han, J. (1998) WebML: Querying the World-Wide Web for resources and knowledge. In: Workshop on Web Information and Data Management WIDM'98, Bethesda, 9–12
10. Zaine, O., Xin, M., Han, J. (1998) Discovering web access patterns and trends by applying OLAP and data mining technology on web logs. In: Advances in Digital Libraries

Family of Instance Reduction Algorithms Versus Other Approaches

Ireneusz Czarnowski and Piotr Jędrzejowicz

Department of Information Systems
Gdynia Maritime University
Morska 83, 81-225 Gdynia, Poland
e-mail: irek, pj@am.gdynia.pl

Abstract. The goal of the paper is to compare the performance of instance reduction algorithms (IRA) with other approaches. The paper briefly describes a family of instance reduction algorithms proposed by the authors. To evaluate their performance the computational experiment is carried out. The experiment involves comparing a performance of IRA with several other approaches using alternative machine learning classification tools.

1 Introduction

As it has been observed in [15], in the supervised learning, a machine-learning algorithm is shown a training set, which is a collection of training examples called instances. Each instance has an input vector and an output value. After learning from the training set, the learning algorithm is presented with additional input vectors, and the algorithm must generalize, that is to decide what the output value should be. It is well known that in order to avoid excessive storage and time complexity, and possibly, to improve generalization accuracy by avoiding noise and overfitting, it is often beneficial to reduce original training set by removing some instances before learning phase or to modify the instances using a new representation.

In the instance reduction approach a subset S of instances to keep from the original training set T can be obtained using incremental, decremental and batch search [7], [15]. The incremental search begins with an empty subset S and adds an instance to S if it fulfills some criteria. Incremental algorithms include CNN [8], IB2 and IB3 [1]. The decremental search begins with $S = T$, and then searches for instances to remove from S. The decremental strategy is used by SNN [13], ENN [15] and DROP1-DROP5 [15]. Finally, in a batch search mode, all instances are evaluated using some removal criteria. Then all those meeting these criteria are removed in a single step. This strategy is represented by All k-NN [14].

This paper presents a family of four instance reduction algorithms denoted respectively IRA1-IRA4. The strategy used by the family belongs to the batch search mode class and is based on calculating for each instance in the original training set the value of the so called similarity coefficient.

The paper describes a family of the proposed instance reduction algorithms. To evaluate their performance the computational experiment is carried out. The experiment involves comparing a performance of IRA with several other approaches using alternative machine learning classification tools. The tools in question are the k nearest neighbor model, the decision-tree classifier and two classifiers based on the artificial neural network.

The paper is organized as follows. Section 2 contains a description of IRA. Section 3 gives details of the computational experiment design and discusses its results. Section 4 includes conclusions and suggestions for future research.

2 Instance Reduction Algorithms

Instance Reduction Algorithms (IRA1-IRA4) aim at removing a number of instances from the original training set T and thus producing reduced training set S. Let N denote the number of instances in T and n — the number of attributes. Total length of each instance (i.e. training example) is equal to $n+1$, where element numbered $n+1$ contains the output value. Let also $X = x_{ij}$ $(i = 1, \ldots, N,\ j = 1, \ldots, n+1)$ denote a matrix of $n+1$ columns and N rows containing values of all instances from T.

The general idea of IRA involves the following steps: calculating for each instance from the original training set the value of its similarity coefficient, grouping instances into clusters consisting of instances with identical values of this coefficient, selecting the representation of instances for each cluster and removing remaining instances, thus producing the reduced training set. For selecting instances four different methods are used.

2.1 IRA 1

IRA 1, proposed in the earlier paper of the authors [4], involves the following steps:

Stage 1. Transform X normalizing value of each x_{ij} into interval $[0, 1]$ and then rounding it to the nearest integer, that is 0 or 1.

Stage 2. Calculate for each instance from the original training set the value of its similarity coefficient I_i:

$$I_i = \sum_{j=1}^{n+1} x_{ij} s_j, i = 1, \ldots, N, \tag{1}$$

where:

$$s_j = \sum_{i=1}^{N} x_{ij}, j = 1, \ldots, n+1. \tag{2}$$

Stage 3. Map input vectors (i.e. rows from X) into t clusters denoted as Y_v, $v = 1, \ldots, t$. Each cluster contains input vectors with identical value of the similarity coefficient I_i, where t is a number of different values of I_i.

Stage 4. Set value of the representation level K, which denotes the maximum number of input vectors to be retained in each of t clusters defined in Stage 3. Value of K is set arbitrarily by the user.

Stage 5. Select input vectors to be retained in each cluster. Let y_v denote a number of input vectors in cluster v. Then the following rules for selecting input vectors are applied:

- If $y_v \leq K$ then $S = S \cup Y_v$.
- If $y_v > K$ then the order of input vector in Y_v is randomized and the cluster partitioned into subsets denoted as D_{vj}, $j = 1, \ldots, q$. Generalization accuracy of each subset denoted as A_{vj} is calculated using the so called *leave-(q-1)-out test* with $X' = X - Y_v + D_{vj}$ as a training set. Subset of input vectors from cluster v maximizing value of A_{vj} is kept to store in the reduced training set.

2.2 IRA 2

IRA 2, proposed in the earlier paper of the authors [5], differs slightly from IRA1. Mapping of instances into clusters in IRA2 is based on rounding the value of each x_{ij} (granularity level is chosen by the user). Similarity coefficient is than rounded recursively until the required representation level is reached.

2.3 IRA 3

IRA3 is based on implementing the population-learning algorithm (PLA) for selecting instances. Steps 1-4 in IRA 3 are identical as in IRA1 but Step 5 is different:

Step 5. Select instances to be retained in each cluster:

- If $y_v \leq K$ and $K > 1$ then $S = S \cup Y_v$.
- If $y_v > K$ and $K = 1$ then $S = S \cup x_v$, where x_v is a selected reference instance from the cluster Y_v, where the distance $d(x^v, \mu^v) = \sqrt{\sum_{i=1}^{n} (x_i^v - \mu_i^v)^2}$ is minimal and $\mu^v = \frac{1}{y_v} \sum_{j=1}^{y_v} x_v$ is the mean vector of the cluster Y_v.
- If $y_v > K$ and $K > 1$ then $S = S \cup \{x_j^v\}$, where x_j^v $(j = 1, \ldots, K)$ are reference instances from the cluster Y_v selected by applying the PLA algorithm.

The selection process (in case when $y_v > K$) is based on a distance criterion. When $y_v > K$ and $K > 1$ IRA3 uses PLA algorithm, introduced in [9]. The PLA algorithm implemented for selection of reference instances maps instances x^v from Y_v into K subset D_{vj}, where the sum of the squared Euclidean distances between each instances x^v and the mean vector μ^j of the subset D_{vj} is minimal. This selection method can be associated with one of the clustering technique known as k-means algorithm [10].

2.4 IRA 4

IRA 4 is identical to IRA 3 except that for selection of reference instances the PLA with real numbers representation is used, where a solution is a sequence of real numbers representing the K reference instances. The length of an individual is elements $K \cdot n$, where the first n positions represent the n dimensions of the first reference vector, the next n positions represent those of the second reference vector, and so on. In this case fitness of an individual is calculated based on Manhattan distance function.

3 Computational Experiment Results

To validate the proposed IRA it has been decided to use generalization accuracy as a criterion. The obtained results were compared with results of several well-known instance reduction strategies (i.e. CNN, IB2, IB3, SNN, ENN, DROP1-DROP5, All k-NN). The performance of instance algorithms is experimentally evaluated using four classification tools — the k nearest neighbor model, the decision-tree classifier (C4.5) [12], the parallel-PLAANNs algorithm and cascade-PLAANN algorithm. The third algorithm is based on an artificial neural network which is trained using an implementation of the population learning algorithm. Possibility of applying population learning algorithms to train ANN has been investigated in earlier paper of the authors [3]. The last algorithm is based on the idea of the Cascade Correlation (CasCor) algorithm proposed by Fahlman [6]. The cascade-PLAANN uses for both — network adjustments and candidate neurons training an implementation of the population learning algorithm [2].

The experiment involved three datasets from UCI Machine Learning Repository [11], which are Wisconsin breast cancer (699 instances, 9 attributes, 2 classes), Cleveland heart disease (303, 13, 2) and credit approval (690, 15, 2). In the computational experiment "10 cross-validation" approach was used. Each dataset was divided into 10 partitions and each reduction technique was given a training set T consisting of 9 of the partitions, from which it reduced a subset S. Each machine learning algorithm was trained using only the instances in S. Ten such trials were run for each dataset and for each reduction algorithm, using a different one partitions as the test set for each trial. This methodology was proposed in [15].

Figures 1-4 present experiment results obtained for each of the investigated classifiers, respectively. The results are averaged over all benchmark instances. In case of IRA family algorithms the reduced instance sets have been generated with the representation levels set, respectively, to 1, 2, 3, 4 and 10.

On Figure 1-4 the horizontal axis (S/T) shows the percentage of compression of the training set, where 100% represents the whole training set. The vertical axis ($\Delta\Phi$) corresponds to accuracy changes depending on instance selection algorithm. The level $\Delta\Phi = 0\%$ is defined by the accuracy obtained

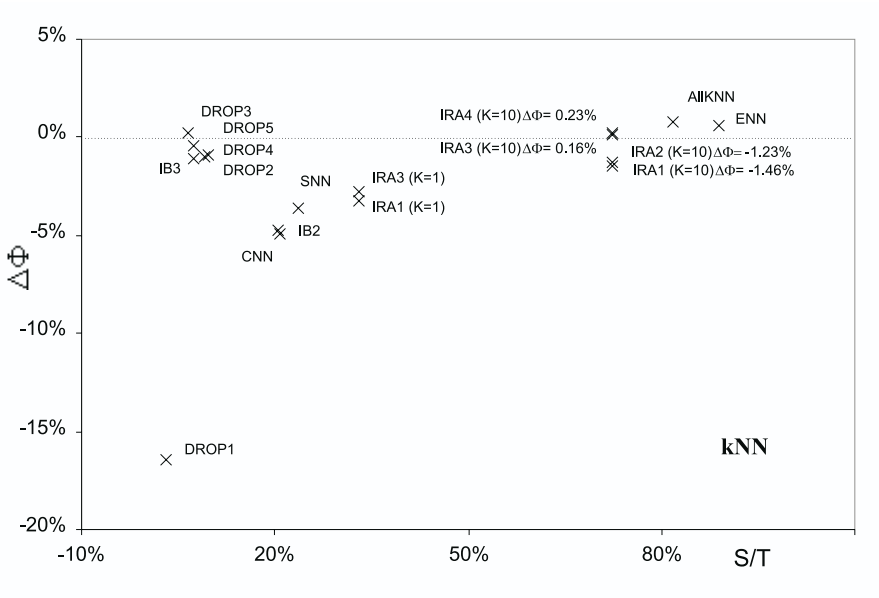


Fig. 1. Results for kNN classifier ($\Delta\Phi = 0\%$ corresponds to accuracy 87.42%)

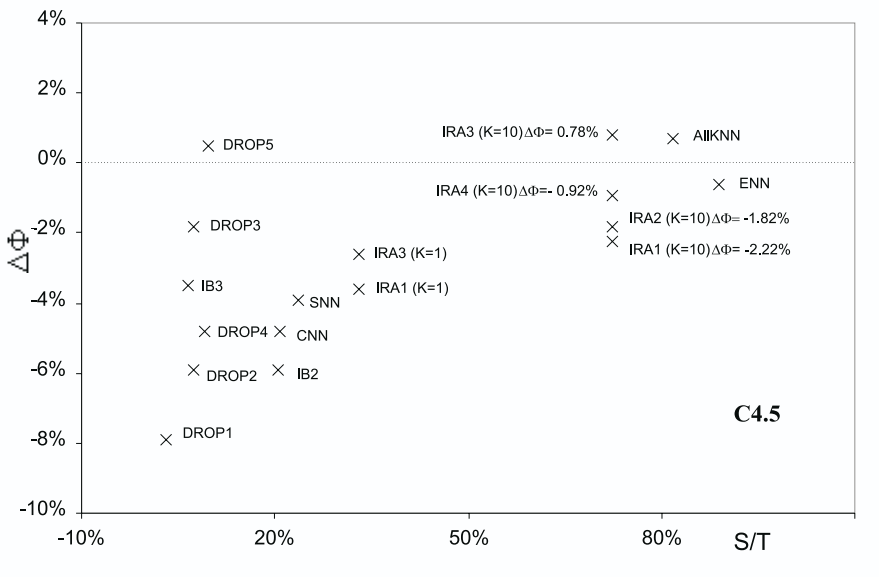


Fig. 2. Results for C4.5 classifier ($\Delta\Phi = 0\%$ corresponds to accuracy 86.02%)

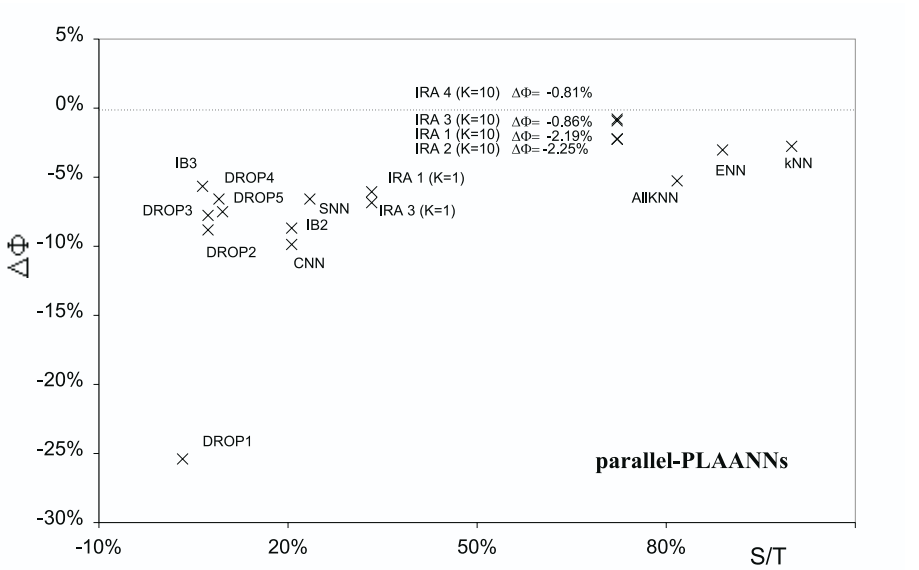


Fig. 3. Results for parallel-PLAANNs classifier ($\Delta\Phi = 0\%$ corresponds to accuracy 89.33%)

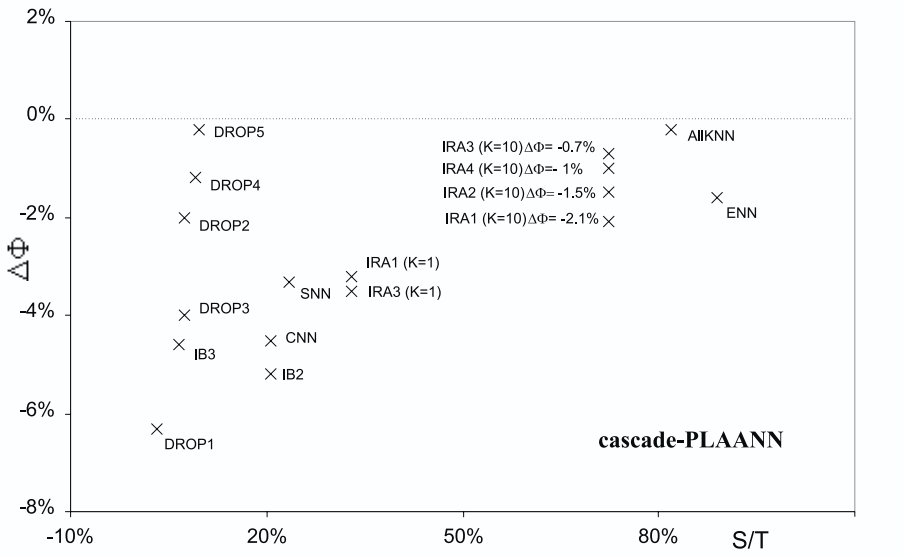


Fig. 4. Results for cascade-PLAANN classifier ($\Delta\Phi = 0\%$ corresponds to accuracy 88.4%)

by the respective classification algorithm trained on the original, that is not reduced, training set.

It can be observed that IRA family performs well with respect to classification accuracy but certainly is no leader as far as compression of the training dataset is concerned. If a problem of choosing the right instance reduction approach was considered as the bicriteria optimization problem (classification accuracy versus a compression ratio) then IRA solutions can be considered as Pareto-optimal and belonging to a non-dominated set of methods.

The best results were obtained using IRA3 and IRA4 with the representation level set to 10. Representation level is a basic parameter of the IRA family directly influencing the compression rate. Hence, setting its value the user can find a required combination of accuracy and data compression rate.

4 Conclusions

The paper shows that the family of the instance reduction algorithms proposed by the authors can be a viable option when choosing an instance reduction algorithm at the data pre-processing stage. Computation experiment results confirm that the proposed algorithms can help reducing training set size and still preserving basic features of the analysed data. The reduced training set guarantees high generalization accuracy, sometimes only slightly worse and often even better then the accuracy achieved using the whole training set. Moreover, the IRA family are often competitive to other instance reduction algorithms. It can be also concluded that the classifier choice does not have a significant impact on performance of instance reduction algorithms. The future direction of research will focus on establishing decision rules for finding an appropriate representation level suitable for each cluster allowing different representation levels for different clusters.

References

1. Aha, D.W., Kibler, D., Albert, M.K. (1991) Machine Learning 6, 37-66
2. Czarnowski, I. (2004) Application of the Population Learning Algorithm to Artificial Neural Network Training. Ph.D. Thesis, Poznań University of Technology (in Polish)
3. Czarnowski, I., Jędrzejowicz, P. (2002) An Approach to Artificial Neural Network Training. In Max Bramer, Alun Preece and Franc Coenen (eds.): Research and Development in Intelligent Systems XIX, Springer, 149-162
4. Czarnowski, I., Jędrzejowicz, P. (2003) An Instance Reduction Algorithm for Supervised Learning. Proceedings of the Intelligent Information Systems. In Kłopotek M.A., Wierzchoń S.T., Trojanowski K. (eds.): Intelligent Information Processing and Web Mining, Advances in Soft Computing, Springer Verlag, 241-250

5. Czarnowski, I., Jędrzejowicz, P. (2004) An Approach to Instance Reduction in Supervised Learning. In Coenen F., Preece A. and Macintosh A. (eds.): Research and Development in Intelligent Systems XX, Springer, London, 267-282
6. Fahlman, S.E., Lebiere, C. (1990) The Cascade-Correlation Learning Architecture. Technical Report CMU-CS-90-100, School of Computer Science, Carnegie Mellon University, Pittsburgh
7. Gates G.W. (1972) The Reduced Nearest Neighbour Rule. IEEE Trans. on Information Theory, IT-18-3, 431-433
8. Hart, P.E. (1968) The Condensed Nearest Neighbour Rule. IEEE Transactions on Information Theory, 14, 515-516
9. Jędrzejowicz, P. (1999) Social Learning Algorithm as a Tool for Solving Some Difficult Scheduling Problems. Foundation of Computing and Decision Sciences 24, 51-66
10. Likas, A., Vlassis, N., Verbeek, J.J. (2003) The Global k-means Clustering Algorithm, Pattern Recognition 36(2), 451-461
11. Merz, C.J., Murphy, P.M. (1998) UCI Repository of Machine Learning Databases [http://www.ics.uci.edu/∼mlearn/MLRepository.html] Irvine, CA: University of California, Department of Information and Computer Science
12. Quinlan, J.R. (1996) Improved Use of Continuous Attributes in C4.5. Journal of Artificial Intelligence Research 4, 77-90
13. Ritter, G.L., Woodruff, H.B., Lowry, S.R., Isenhour, T.L. (1975) An Algorithm for a Selective Nearest Decision Rule. IEEE Trans. on Information Theory, 21, 665-669
14. Tomek, I. (1976) An Experiment with the Edited Nearest-Neighbour Rule. IEEE Trans. on Systems, Man, and Cybernetics, 6-6, 448-452
15. Wilson, D. R., Martinez, T. R. (2000) Reduction Techniques for Instance-based Learning Algorithm. Machine Learning, Kluwer Academic Publishers, Boston, 33-3, 257-286

Minimum Spanning Trees Displaying Semantic Similarity

Włodzisław Duch[1,2] and Paweł Matykiewicz[1,3]

[1] Department of Informatics, Nicholaus Copernicus University, Toruń, Poland
[2] School of Computer Engineering, Nanyang Technological University, Singapore
[3] Department of Biomedical Informatics, Children's Hospital Research Foundation, Cincinnati, Ohio, USA
Google: Duch; emailpawelm@phys.uni.torun.pl

Abstract. Similarity of semantic content of web pages is displayed using interactive graphs presenting fragments of minimum spanning trees. Homepages of people are analyzed, parsed into XML documents and visualized using TouchGraph LinkBrowser, displaying clusters of people that share common interest. The structure of these graphs is strongly affected by selection of information used to calculate similarity. Influence of simple selection and Latent Semantic Analysis (LSA) on structures of such graphs is analyzed. Homepages and lists of publications are converted to a word frequency vector, filtered, weighted and similarity matrix between normalized vectors is used to create separate minimum sub-trees showing clustering of people's interest. Results show that in this application simple selection of important keywords is as good as LSA but with much lower algorithmic complexity.

1 Introduction

Maps presenting connections, associations, clustering or similarity are a popular way to present information. Google knowledge management directory contains links to dozens projects on "mind maps", graphs that map associations between concepts and facilitate thought processes [4]. Mind maps are inspired by the associative nature of human memory and thinking. Projects in this area are focused on graphical representations and manual entry of data. Graphs presenting connectivity of web pages are the simplest to create. A commercial interface called "The Brain" [3] displays graphs similar to mind-maps helping to organize the web links, local files and user notes.

Automatic construction of mind maps containing semantic relations between text documents such as Web pages is more difficult. Various document clusterization and visualization methods may be used to display relations and similarity of text documents. For example, hierarchical SOM maps displaying document clusterization, called WebSOM [6], have been used in a number of applications. Presenting similarities between text objects (web pages or documents) on a graph is a useful technique allowing for a quick overview of a large amount of information.

In large organizations, such as universities or big companies, it is quite difficult to learn who works on similar projects. Analyzing people's homepages and their lists of publications is a good way to find groups and individuals who share common interest. Our work has been motivated by a practical need to bring people with similar interest in contact by showing them a map that links them to other people. We have performed a number of experiments collecting data from people's webpages and investigating the influence of keyword selection and dimensionality reduction via latent semantic analysis on the accuracy of clusterization and the structure of resulting graphs. In the next section algorithms used to derive similarity between people's interests are presented, and in the third section difficulties encountered in this type of application and results of computational experiments are described.

2 Algorithms

Conversion of HTML web pages to pure text requires removing all HTML tags, Javascript programs and comments. `HTML::Parser` [1] written in Perl is used for extracting text that is displayed by web browsers. After converting n documents the text is broken into terms and all terms from a stop-word list are removed. Then Porter's algorithm [12] is used for stemming, counting occurrence of every stemmed term in each document. Results are collected in a matrix $\mathbf{F}$ containing n vector columns, each containing frequency of m stemmed words in a given document. Finally term weighting is applied to create the final term-document matrix.

Let f_{ij} be the number of occurrences of a term j in a document i. Normalized term frequency tf_{ij} is obtained by dividing each column of the $\mathbf{F}$ matrix by the maximum element in that column:

$$tf_{ij} = \frac{f_{ij}}{\max_i(f_{ij})} \tag{1}$$

Inverse document frequency idf_j is a measure of uniqueness of term j. Let n be the number of all documents and d_j a number of documents in which term j occurs, then:

$$idf_j = \log_2\left(\frac{n}{d_j}\right) + 1 \tag{2}$$

The weight w_{ij} of a term i in a document j is defined by:

$$w_{ij} = tf_{ij} \cdot idf_j = \frac{f_{ij}}{\max_i(f_{ij})}\left(\log_2\left(\frac{n}{d_j}\right) + 1\right) \tag{3}$$

Relations between documents may be derived and visualized calculating similarity between columns of the $\mathbf{W} = (w_{ij})_{m\times n}$ matrix. This weight matrix is usually quite large, with many terms, therefore calculation of similarity between documents is influenced by many small contributions from accidental terms. Two dimensionality reduction techniques have been used to avoid it, simple term selection and latent semantic analysis.

2.1 Simple selection of terms

In the m-dimensional space of keywords a hypercube $\mathcal{H} = [0,1]^m$ may be defined. Documents correspond to vectors $\mathbf{W}_j = (w_{ij})_{m\times 1}$ pointing towards some vertices. The direction of this vector should be defined by the largest components of $\mathbf{W}$. Mapping of $\mathbf{W}_j \in \mathcal{R}^m$ vectors to $\mathbf{H}_j \in \mathcal{H}$ vectors that point to the vertex of that hypercube, e.g. $\mathbf{H}_j = (1,0,1,0,...)$, should preserve all important keywords. The average weight value for document j is given by the expectation $\mathbf{W}_j$ over non-zero weights:

$$\theta_j = E\left(\mathbf{W}_j | w_{ij} > 0\right) \tag{4}$$

Using the step function Θ with average as the bias:

$$h_{ij} = \Theta(w_{ij} - \theta_j), \tag{5}$$

a binary $\mathbf{H}_j = (h_{ij})_{m\times 1}$ matrix is produced; rows that contain only zeros are removed, reducing its dimensionality to m'. This matrix is used instead of the $\mathbf{W}$ matrix to calculate similarity between documents:

$$s^h_{ij} = \cos(\overrightarrow{H_j}, \overrightarrow{H_k}) = \frac{\sum_{i=1}^{m'} h_{ij}h_{ik}}{\sqrt{\sum_{i=1}^{m'} h^2_{ij}}\sqrt{\sum_{i=1}^{m'} h^2_{ik}}} \tag{6}$$

The set of all cosines defines a symmetrical similarity matrix between documents $\mathbf{S}^h = (s^h_{ij})_{n\times n}$, where every element s^h_{ij} reflects the degree of semantic similarity between documents i and j in the reduced space. Thresholds θ_j may also be treated as adaptive parameters and optimized using some cost function, but for our purpose simple selection is sufficient.

2.2 Latent Semantic Analysis

Classical LSA algorithm [8] is often used to improve document similarity estimations. Singular value decomposition (SVD) is a robust way of dimensionality reduction. In this algorithm matrix $\mathbf{W}$ is presented as:

$$\mathbf{W} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^T \tag{7}$$

where $\mathbf{U}$ is a matrix with eigenvectors of a $\mathbf{W}\mathbf{W}^T$ matrix, representing the original row entries as vectors of derived orthogonal factor values, $\mathbf{V}$ is a matrix with eigenvectors of the transposed $\mathbf{W}^T\mathbf{W}$ matrix, representing the original column entries in the same way, and $\mathbf{\Lambda}$ is a matrix with diagonal elements equal to eigenvalues of the $\mathbf{W}^T\mathbf{W}$ matrix, acting as a scaling coefficients. Reduction of dimensionality is done by zeroing small eigenvalues, creating reduced $\overline{\mathbf{\Lambda}}$ matrix:

$$\overline{\mathbf{W}} = \mathbf{U}\overline{\mathbf{\Lambda}}\mathbf{V}^T \tag{8}$$

$\overline{\mathbf{W}}$ is the best reconstruction of the original matrix that may be obtained from the information in the reduced r-dimensional space, where r, called the rank of the matrix $\mathbf{W}$, is the number of non-zero elements in the $\overline{\mathbf{\Lambda}}$ matrix. Removing eigenvectors corresponding to small eigenvalues leads to reconstruction that is largely noise-free, capturing important regularities of the word distribution. The $\overline{\mathbf{W}}$ matrix is used to calculate similarity between documents corresponding to $\overline{\mathbf{W}}_j$ columns. Similarity matrix $\overline{\mathbf{S}}$ with elements $\overline{s}_{ij} = \overline{\mathbf{W}}_i \cdot \overline{\mathbf{W}}_j / \|\overline{\mathbf{W}}_i\| \|\overline{\mathbf{W}}_j\|$ captures semantical relationships between documents in the collection.

2.3 Minimum Sub-Trees Clusterization

A graph algorithm is used to visualize document clusters. First the similarity matrices are replaced by dissimilarities $d_{ij} = 1 - s_{ij}$ for both $\mathbf{S}^h$ and $\overline{\mathbf{S}}$ matrices. Because $\mathbf{D}^h$ and $\overline{\mathbf{D}}$ matrices are symmetrical they may represent weights (arcs) in a fully connected graph. A modified Kruskal minimum spanning tree (MST) algorithm [7] is used for finding a collection of *minimum sub-trees* that represent document clusters. Such collection of trees is a decomposition of a *minimum spanning tree*. Connecting these minimum sub-trees to their nearest sub-trees via the shortest arc a minimum spanning tree is obtained.

The original Kruskal algorithm in application to our problem creates a single MST tree containing all documents, independent of the number of documents. Kruskal algorithm first sorts arcs, then marks the shortest arcs blue, avoiding cycles by marking other arcs as red. Blue arcs form the MST. The algorithm considers arcs in ascending order of their weights – if both endpoints of an arc are in *the same blue subtree*, then the arc becomes red, else it is blue.

In order to have a number of blue sub-trees depend on relations between documents a modification of the original Kruskal's algorithm is proposed. Marking of the blue arcs that have endpoints in different blue trees are preserved, obtaining separate blue trees. A minimum sub-tree clusterization algorithm considers arcs in the ascending order of their weights – if both endpoints of an arc are in *a blue tree*, color it red, else color it blue. This modification does not allow to form not only cycles but also to merge different minimum trees. The number of such trees p depends on the weight matrix and the number of documents.

Every minimum sub-tree can be considered as a $\tilde{C}_i$ cluster for a subset of documents. Such a cluster holds documents that have similar semantic representation and the number of documents cannot be lower then 2. Further the set of all clusters $\tilde{C} = \tilde{C}_1 \cup \tilde{C}_2 \cup \ldots \cup \tilde{C}_p$ containing all documents will be used for evaluation of the accuracy of visual representation obtained from algorithms used. It should be mentioned that minimum sub-tree clusterization algorithm has the same low computational cost that Kruskal algorithm that is $O(n \log n)$ for sorting and $O(n \log n)$ for processing the edges.

The accuracy of this algorithm in tests presented below was higher than the accuracy of threshold-based MST decomposition.

3 Testing accuracy on Reuter's data

To estimate the accuracy of clusterization provided by the sub-trees experiments with Reuters-21578 [9] datasets were conducted. The original format of these documents is SGML, therefore conversion and some preprocessing was performed. The title and the text body for documents containing more then 600 bytes and a single, unique label were used. These labels t_j were employed to group documents into topics T_j, that is sets containing documents with the same label. The minimum sub-tree cluster $\tilde{C}_i$ is assigned to a topic that the majority of documents in this cluster belong to. For every cluster $\tilde{C}_i \in \tilde{C}$ with $|\tilde{C}_i|$ elements, $n(\tilde{C}_i)$ documents belong to the topic that the cluster is assigned to. The number of clusters is typically larger than the number of topics, therefore the same topic may be assigned to several clusters. Accuracy is measured by summing $n(\tilde{C}_i)$ over all clusters and dividing by the number of documents, $A = \sum_i n(\tilde{C}_i)/n$.

Two tests are reported here. First 600 documents that passed through the pre-processing were used to create $\mathbf{W}$ matrix. The number of documents in 41 topics ranged from 176 to 1, so perfect clusterization is not possible (our clusters have at least 2 elements). Moreover, SVD revealed rank($\mathbf{W}$) = 595. The accuracy was evaluated without dimensionality reduction, with selection and with LSA with the number of eigenvectors equal to 0.8 and 0.6 times the rank of the original matrix (Table 1). The number of clusters for each case was similar, and the accuracy did not differ much, with simple selection achieving slightly better results at much lower computational cost comparing to LSA.

Table 1. Results of parsing first 600 documents, 41 topics.

Method	♯ topics	♯ clusters	accuracy
No dim red.	41	129	0.782
LSA dim red. 0.8 (476)	41	124	0.762
LSA dim red. 0.6 (357)	41	127	0.752
Simple Selection	41	130	0.785

An easier test, with the first 600 documents selected from 10 topics: *acq, coffee, crude, earn, GNP, interest, money-fx, ship, sugar, trade*, and 60 documents in each topic, is summarized in Table 2. Accuracy is slightly higher, with small differences in the number of clusters and simple selection again giving somewhat improved results.

LSA has slightly reduced the number of clusters in both experiments. Although more sophisticated clusterization methods may improve these results

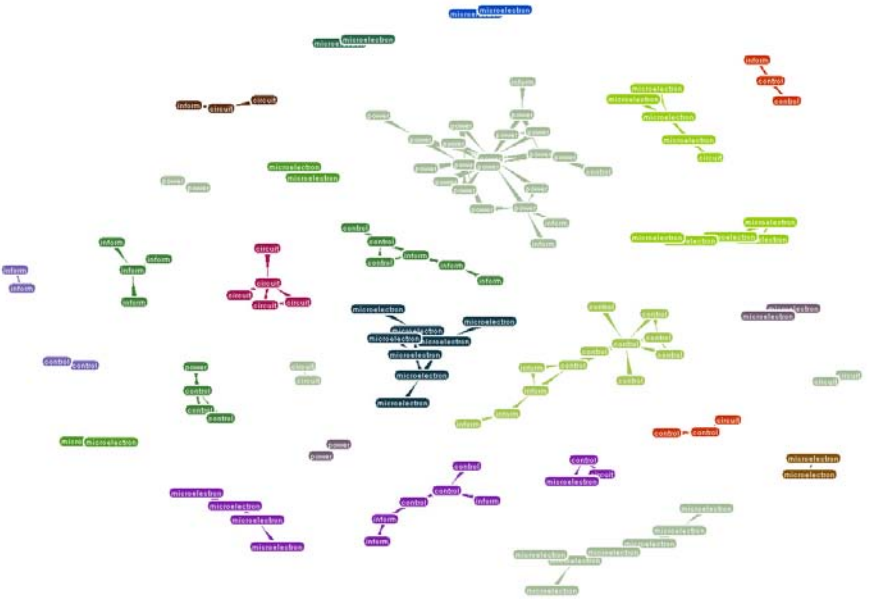


Fig. 1. TouchGraph LinkBrowser screenshot of 124 homepages from the EEE School of the Nanyang Technological University, Singapore, without dimensionality reductions.

for our purpose the simple selection approach, fully automated, without any adaptive parameters, seems to be sufficiently accurate.

Table 2. Results of parsing first 600 documents with selected topics.

Method	♯ topics	♯ clusters	accuracy
No dim red.	10	142	0.847
LSA dim red. 0.8 (467)	10	129	0.847
LSA dim red. 0.6 (350)	10	137	0.828
Simple Selection	10	145	0.855

4 Real application

Our main motivation was to discover groups of experts sharing common interest in large institutions, such as universities. 124 web pages of the School of Electrical and Electronic Engineering (EEE) of the Nanyang Technological University, Singapore, were used to create the weight matrix **W**. Before applying algorithms described above additional preprocessing was necessary. Only the pages that were at least 600 bytes long and contained a regular expression with the name of a *division, school* or an *institute*were used. These

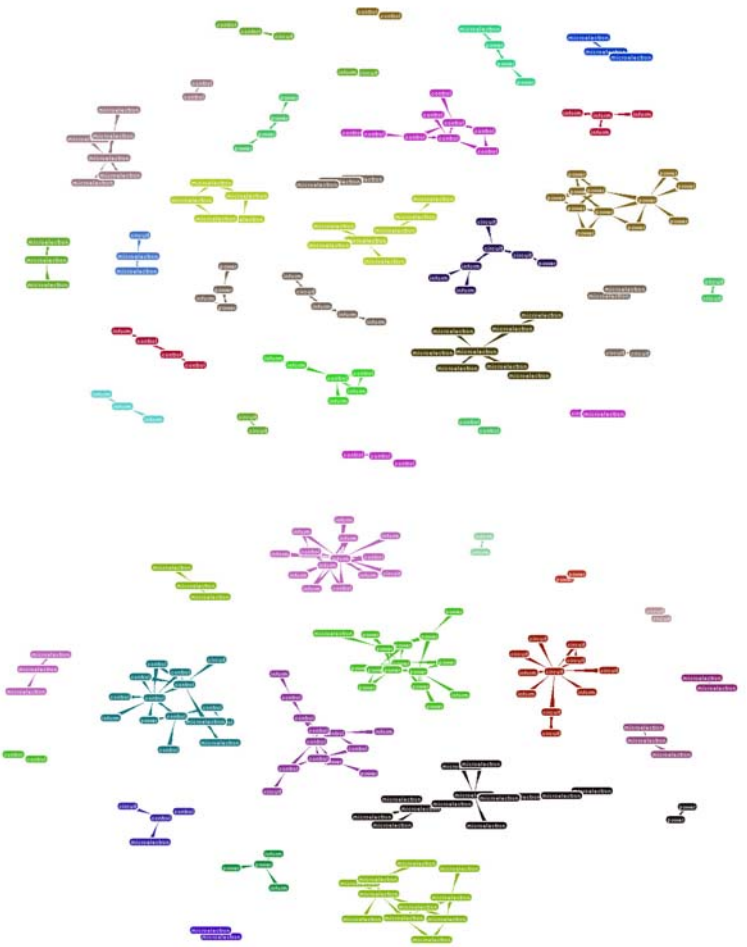


Fig. 2. Same webpages after selection (left) and LSA (right) with reduction factor 0.8.

names were used as topic labels for the Web pages. 5 topics were found, *microelectronics, information, circuit, power, control*, with the number of pages ranging from 14 to 41. The terms were cleaned using a stop-word list consisting of standard words (*and, the, for* ...) extended by words that occurred only once or were present on all pages.

Figures 1 and 2 show screenshots of the TouchGraph LinkBrowser [2] applet with graphical representation of the similarity information (stored in the XML format). Every cluster (minimum sub-tree) was colored for better visualization. Moreover, only first one or two shortest edges inside every cluster are presented. Figure 1 shows similarity between homepages with dimension-

ality reduction. Clustering of experts into different research groups is clearly visible. Clicking on an individual label centers the graph on a given person showing links to people with highly overlapping interest [10]. Graphs show names and divisions of people, and clicking on them leads to their web pages.

There is no guarantee that someone working in the division of microelectronics does not belong to a cluster dominated by control people, therefore accuracy in Table 3 is only approximate. Figures 1, 2 suggest that simple selection is a good approach for that kind of clusterization.

Table 3. Accuracy of clustering personal home shown in Fig. 1

Method	♯ topics	♯ clusters	*accuracy*
no dim red.	5	28	0.831
LSA dim red. 0.8 (98)	5	19	0.815
LSA dim red. 0.6 (74)	5	22	0.766
Simple Selection	5	30	0.887

5 Discussion and conclusions

This paper has been driven by a practical application, visualization of shared interest based on similarity analysis of homepages and publication records. A modified minimum spanning sub-trees were used to present relations among different interest groups. It has been found that in this case simple selection of important keywords improves accuracy of clustering and thus the structure of semantic connectivity graphs more than latent semantic analysis LSA. There are many knowledge-based clustering methods [11] that could also be used but problems encountered in practical applications cannot be solved by modification of clustering methods.

Homepages frequently contain meaningless titles, and may contain a lot of irrelevant information. Bibliographies are very useful source of important keywords but on some pages they are generated on demand and their retrieval requires sophisticated agents gathering information. They also contain many names, acronyms and abbreviations that may not be easy to map uniquely to full journal titles. Publishing in the same journals is rather weak indication of similarity of interest. To determine how relevant a given word found in a web page is *a priori* inverse document frequencies may be calculated using some large corpus. Most words in the list of a top few thousands highest frequency words are not useful to determine similarity of expert's interest. Instead of a stop-list, a go-list could be compiled in some applications, including only specialized keywords that are found in medical subject headings, science abstracts, or hierarchical classification schemes used by libraries identify different fields of science. Using a specialized corpus of texts that belong

to a general category would be helpful in determination of *a priori* weights. Selection of relevant information for clustering is a real challenge.

Statistical approach to similarity should benefit from knowledge-based analysis based on an attempt to understand some information contained in personal pages. Such knowledge based clustering should include synonyms, noun phrases, and similarity between concepts. This may be realized by using concept vectors $\mathbf{V}_i$ instead of terms i, and modifying frequencies $f_{ij} = \sum_k S(\mathbf{V}_i, \mathbf{V}_k)$ by summing over all concepts k in the document j, where $S(\mathbf{V}_i, \mathbf{V}_k)$ will be 1 for concepts that are identical or synonymous, and 0 for concepts that are significantly different. Some users may be interested in finding a very specific interest group, not just a group that shares their general interest. For example, the fact that people are members of the same organization, have obtained their degrees from the same university, have links to the same Web pages, may all be of interest to the user. This will require addition of inference mechanisms.

Minimum spanning trees may not be the best way to display similarity information and we have also experimented with multidimensional scaling algorithms [5]. Depending on the actual objective functions global or local structure may be preserved in a better way in MDS maps. While MST graphs may show several unrelated clusters MDS may reveal that some members of a cluster are in fact similar to those of other clusters.

Software for creation of graphs displaying homepages of people sharing common interest may have numerous applications. Obviously it should be coupled with a web crawler that visits sites within some domain, checks if they have characteristics of homepages, follows links to separate pages containing bibliographies and retrieves them. Graphs may be created from different perspectives, displaying homepages of people that share particular interest. For example, asking for all people interested in "neural networks for control" a virtual group of experts that work on this topic in some region or large institution may be created. Trying to derive the same information from search engines proved to be quite difficult. We have identified the main problems and made the first steps towards practical application but the algorithms presented here may still be improved in many ways.

References

1. G. Aas. Html-parser. http://search.cpan.org/~gaas/HTML-Parser, 2004.
2. G. Aas. Html-parser. http://www.touchgraph.com, 2004.
3. The Brain. The brain. http://www.thebrain.com, 2004.
4. T. Buzan. Mind maps. http://www.mind-map.com, 2004.
5. P. Groenen I. Borg. *Modern Multidimensional Scaling. Theory and Applications.* Springer Series in Statistics, Heidelberg, 1996.
6. T. Kohonen. Websom. http://websom.hut.fi, 1999.
7. J. Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. In *Proceedings of the American Mathematical Society*, volume 7, pages 48–50, 1956.

8. T. K. Landauer, P. W. Foltz, and D. Laham. Introduction to latent semantic analysis. *Discourse Processes*, 25:259–284, 1998.
9. D.D. Lewis. Reuters-21578 text categorization test collection. http://www.daviddlewis.com/resources/testcollections/reuters21578/, 1997.
10. P. Matykiewicz. Demonstration applet. http://www.neuron.m4u.pl/search, 2004.
11. W. Pedrycz. *Knowledge-Based Clustering: From Data to Information Granules*. John Wiley and Sons, Chichester, 2005.
12. M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):48–50, 1980.

Concept Description Vectors and the 20 Question Game

Włodzisław Duch[1,2], Julian Szymański[1], and Tomasz Sarnatowicz[1]

[1] Department of Informatics, Nicholaus Copernicus University, Toruń, Poland
[2] School of Computer Engineering, Nanyang Technological University, Singapore

Abstract. Knowledge of properties that are applicable to a given object is a necessary prerequisite to formulate intelligent question. Concept description vectors provide simplest representation of this knowledge, storing for each object information about the values of its properties. Experiments with automatic creation of concept description vectors from various sources, including ontologies, dictionaries, encyclopedias and unstructured text sources, are described. Information collected in this way is used to formulate questions that have high discriminating power in the twenty questions game.

1 Introduction

Since the famous "Eliza" program of Weizenbaum [1] chatterbot programs attempt to discover keywords and sustain dialog by asking pre-prepared questions without understanding the subject of conversation. This is quite evident from the Loebner prize chatterbot competition [2] and the lack of progress in text understanding and natural language dialogue systems. Programs that are based on semantic networks work well only in narrow domains [3].

One of the basic problems that remain unsolved is the poverty of representation of symbols. Thinking about an object – an elephant, a car, or a rose, for example – we can imagine and immediately describe general and specific properties of that object, discuss these properties and create new instances of such objects, by inventing unusual properties, for example a "pink elephant". Artificial natural language processing (NLP) systems have to make a lot of inferences to determine that a car has cylinders, while for humans interested in sports cars question "how many cylinders does it have" is quite natural. Ontologies are quite helpful to organize knowledge in hierarchical way, but it is very difficult to use ontologies to generate object description in terms of all properties it has and find which properties are not applicable to an object. Semantic networks may create a faithful representation of an episode, capturing relations in a story, but people ask questions that are outside the scope of the story because they can imagine objects and actors in more details that semantic network model is able to provide. Reasoning with rich prior knowledge may be shallow and associative, while reasoning with symbols found in ontologies and semantic networks has to be deep, and therefore difficult to perform. Tradeoffs between richness of concept representation, efficiency

of use, and the depth of reasoning needed to understand questions deserve careful exploration.

Meaning of the words is reflected in their use and in the similarity of the concepts words refer to. Thinking about an object creates a set of activations of various brain modules, facilitating associations and building of semantic relations [4]. Seeing a large cat we do not need to reason that all large cats are carnivorous, hunt and may be dangerous, we immediately know it. The simplest approach to add more prior knowledge to NLP systems is to replace linguistic symbols by Concept Description Vectors (CDV). These vectors should contain information about all properties that make sense for a given concept and may be associated with it. They should contain prior knowledge that humans use to understand text, not only context knowledge that may be derived from text analysis. Although vector representation cannot do justice to all language symbols, not even to nouns, it is instructive to see how far one can go in this direction.

For the purpose of this article discussion will be restricted to CDVs for common and proper nouns only. A dictionary of concepts, and CDVs associated with them, may form a basis for intelligent selection of questions that maximizes the information gained after each answer. In this paper we state the problem, present attempts to solve it, and show some potential applications of this approach. In the next section the Concept Description Vector idea is discussed, followed by a section describing algorithms used to derive CDVs in an automatic way from text corpora. An application of these ideas to the 20 question game is discussed in section four.

2 Concept Description Vectors

Context vectors are a popular way to disambiguate word senses and capture information contained in local relations between word pairs. They contain statistical information about word co-occurrences derived from text corpora. Concept Description Vectors provide more systematic representation of prior knowledge. Previous attempts to build lexical knowledgebases from dictionaries were focused on structures of semantic relations [5,6] and analysis of noun sequences [7]. Our goal is much simpler: finding in the list of all dictionary entries those that may characterize a given concept. To achieve this goal sophisticated frame-based representations are not necessary.

How detailed should the CDV representation be? Complex objects cannot be represented in all possible details, for example CDV for some animal should not include details about all known proteins that may be found in cells of this animal. Some ontological hierarchy must be introduced. Properties that are essential to define the meaning of words appear in dictionaries and encyclopedias and should obviously be included. Unfortunately these definitions do not contain exhaustive descriptions; reading that a horse is a "solid-hoofed herbivorous quadruped domesticated since prehistoric times" (Wordnet def-

inition [8]) does not tell us much about the horse. It is doubtful that anyone who has not seen the horse will form a reasonable idea what a horse really is, and how does it differ from a cow that may be described in identical terms. Wordnet [8] definition of a cow, a "mature female of mammals of which the male is called bull" is even less informative.

Explanation of a new concept involves description, or a set of words related to this concept. Such a description will be called a gloss. Keywords in the gloss should explain the essence of the concept, facilitating discrimination of this particular concept from all others. Dictionaries provide short definitions saturated with keywords, while encyclopedias use richer, self-explanatory descriptions. Every concept has its unique set of keywords that distinguishes it from all other concepts. Perfect synonyms should have the same set of keywords because they represent the same concept. Concepts and keywords used for their description are collected in two sets, called $\mathcal{C}$ and $\mathcal{K}$, respectively. Many words will appear in both sets, allowing for recursive expansion of descriptions. Concept Description Vectors $c(k)$ may contain numbers reflecting strength of relation between a concept and particular keyword – it can be a binary number (applicable/irrelevant), a tertiary number (yes/no/irrelevant), a small set of nominal values, a real number, an interval, or a fuzzy number. CDVs are collected in $\mathbf{S}(c, k)$ matrix containing $|\mathcal{C}|$ rows and $|\mathcal{K}|$ columns.

Binary $\mathbf{S}(c, k)$ values may be ambiguous but from computational point of view they are the easiest to handle. Real-valued matrix elements increase the expressive power of vector representation at the expense of storage and computational power needed to process such matrices. The property "color" is inapplicable to the concept of "electron", therefore the element $\mathbf{S}$(electron,color) =0. Horses have color, therefore $\mathbf{S}$(horse,color) = 1, but in the binary representation more information, such as $\mathbf{S}$(horse,color-blue) = 0 is needed, meaning that there are no blue horses. Thus negative answer to the question "can it have a blue color?" in the 20 question game will immediately eliminate horses from the list of potential animals. Binary-valued CDVs are simple but require more specific keywords than CDVs with fuzzy or nominal subset values. If some property, like "color", is irrelevant for electron, than also more specific properties like "color-blue" should be irrelevant.

The number of concepts and keywords may easily reach 10^5, making creation of CDV matrix quite a challenge. The matrix $\mathbf{S}(c, k)$ is obviously quite sparse, with most entries equal to "irrelevant". In the binary representation each concept is defined by a subset of hypercube vertices that is relatively close to the "undefined" concept (vector with zeros). In the ternary representation if the value "irrelevant" is coded as 0 then almost all concepts will lie on the centers of hypercube walls relatively close to the $\mathbf{0}$ vertex.

3 Semi-automatic creation of CDVs

Initially all keywords for CDVs have indefinite value $\mathbf{S}(c, k) = 0$. Dictionaries are the first source of reliable information, but information contained even in the best linguistic resources, such as the Wordnet electronic dictionary [8], manually created over a period of many years, is very poor. Nevertheless we have tried to use several on-line dictionaries, performing the following procedure for each entry found there:

1. create a unique list of all words used in the concept's description (gloss), ignoring duplicates and order of words;
2. convert every word to its base form, eliminate duplicates;
3. filter out all words other than nouns, verbs, adjectives and adverbs;
4. eliminate common words using stop-list (words like "be, have, thing");
5. use remaining words as indices to build CDV vectors.

Processing multiple dictionaries gives several options while creating the CDV vectors:

1. using only keywords that appeared in every dictionary;
2. using keywords from all dictionaries and merging results by a bit-wise sum;
3. using keywords from all dictionaries and creating final vectors as weighted sum of all individual vectors.

An identical procedure was used to create semantic vectors from encyclopedias. Glosses in encyclopedias usually contain full sentences, often complex ones. When analyzed, such sentences often contain "subconcepts" with their individual mini-glosses. There are also parts of a gloss which do not describe the concept directly, giving for example historical background. This makes encyclopedia-based vectors less adequate for direct creation of concept descriptions. On the other hand, using larger blocks of text reduces the risk of missing important keywords.

WordNet (ver. 2.0) and a number of other electronic dictionaries were analyzed: The American Heritage Dictionary of the English Language (4th Ed, 2000), and the Dictionary of Idioms (1997), Easton's 1897 Bible Dictionary, Merriam-Webster Medical Dictionary (2002), Webster's Revised Unabridged Dictionary (1998) and the New Millennium Dictionary of English (2003). Electronic edition of Encyclopedia Britannica (2004) was also used in some queries to expand word definitions.

CDVs were initially generated from WordNet and restricted to words used for description of animals to speed up computational experiments. Attempts to improve CDVs using context derived from larger corpora, such as a collection of books, were not successful, leading to a large number of meaningless correlations. Vectors may contain a small number of keywords that were assigned to identical set of concepts (eg. "hoofed" and "hoofed mammal").

Such duplicate vectors corresponding to perfect synonyms were removed immediately after the whole set was created, reducing the size of CDV matrix. Obvious features that apply to all concepts, and void features that are not applicable to any concepts, should be removed. Vectors created from dictionaries and free text blocks never contain void features and seldom contain obvious features, however this step should not be ignored. Another simple filter was applied to remove all keywords that were shorter than 3 characters, or appeared only in the WordNet dictionary; appearance in at least one more dictionary was a required condition confirming importance of the keyword.

Selected information about the amount of data generated for the "animal" domain by this procedure is presented below.

- Initial size of the CDV matrix was 7591 concepts and 15301 keywords
- Initial filtration reduced the size to 3995 concepts and 7108 keywords, leaving CDV matrix with about 28 million elements.
- Keywords gathered from WordNet glosses fill 0.11% of **S** matrix.
- These keywords propagated down the ontology tree fill 0.72% of **S**.
- Ontology nodes for words propagated down the tree fill 0.25% of **S**.
- Meronym relations ("has part" and "is made from") fill 0.34% of **S**.

Altogether these algorithms assign values to slightly more than 1% of the **S** matrix elements. So far learning CDVs from information contained in books and other large blocks of text proved to be difficult. Some improvements that will be explored include: 1) recognition of the parts of speech and filtering only nouns, verbs, adjectives and adverbs (as done for dictionaries); 2) analyzing noun phrases to discover concepts rather than using single words; 3) filtering weak associations; 4) bootstrapping from simple to complex concepts.

Glosses generated from dictionaries or blocks of free text are a "descriptive" source of semantic information. Another important source useful to create CDVs is derived from ontologies that contain hierarchical relations between pairs of words. Finding that one concept is a member of more general category, for example finding that a dog is a mammal, a set of concepts defined by the concept of mammal may be inherited from this higher-level concept. In the ontology tree each node (except for root) has only one parent, so ontology itself is not a good source of information. Assigning just a parent node as a single feature would not be sufficient. Parent nodes of every concept are propagated down the ontology tree (note that concepts become features or keywords in this way - e.g. "mammal" is a concept, but "being a mammal" is a feature of all its direct and indirect ontology children). Furthermore, we can propagate also features gathered from concept glosses (e.g. "vertebrate", "skin", "milk" which are features of "mammal" itself, will also be assigned to "canine" and "dog").

Wordnet project [8] provides one of the most extensive ontologies. Besides defining an ontology it includes also several other types of relations. Two types of relations have been used here: the hierarchical relation called in

Wordnet "hypernym vs. hyponym", and the meronym relation, "has a part" vs. "is made from". These two relations have been used to build separate sets of semantic vectors. Below two sample sets of features obtained for the concept "buffalo" are presented. Four groups of keywords are listed, obtained from several sources. In Wordnet "buffalo" is defined as: "large shaggy-haired brown bison of North American plains".

1. dictionary glosses
   wrong: north
   correct: brown, large, shaggy, bison, plain, american
2. gloss keywords propagated down the ontology tree
   wrong: each, similar, enclose, even, less, number, north, various, young, small, column, relate, compartment, man, subclass, hollow, divide, voluntary, characterize, functional, three, short, four, bear, except, several, marsupial, monotreme
   correct: warm, toe, brown, segment, skeleton, blood, alive, skin, skull, stomach, spinal, shaggy, foot, brain, cranium, bison, nourish, chordata, cartilaginous, notochord, milk, mammal, hair, hump, large, placenta, head, phylum, movement, organism, hoof, bovid, cover, ruminant, cud, chew, bony, live, horn
3. ontology headers
   correct: eutherian, bison, bison bison, craniate, ruminant, chordate, bovid, ungulate, brute, mammal, artiodactyl
4. meronyms
   correct: coat, belly, vertebrate foot, pedal extremity, pectus, dactyl, psalterium, caudal appendage, fourth stomach, first stomach, abomasum, cannon, digit, caput, animal tissue, thorax, face, shank, hock, hoof, tail, chest, head, hair, pilus, pelage, third stomach, second stomach, rumen, reticulum, omasum, costa, rib

A lot of useful information has been collected here, although some of it is contradictory and confusing (ex. "large" and "small"). Adjectives should be kept with nouns, ex. "(bison,large)", or "(bison,color-brown)", and the same goes for numbers. For some applications rarely used words known only to experts in some fields ("artiodactyl" or "monotreme") may be omitted. Most of the wrongly recognized keywords begin to make sense when properly grouped in phrases (e.g. north + american + plains).

4 20 question game

The original motivation for creation of CDVs came from the need to find optimal questions in the popular 20 questions game. In this game one person thinks about a word and another person has to guess this word asking no more than 20 questions. Answers should be just yes or no, although in some variants of the game a selection from a small subset of answers is allowed.

The first question that is being asked is usually: "Is this an animal, plant, mineral, or other?"

This game is interesting for several reasons. Answering queries by search engines requires the ability to ask questions that resolve ambiguities. In the 20 question game nothing is initially known, but after obtaining answers to several questions the definition of the subject becomes more precise, eg. "it is an animal, it is big, it is carnivorous", and an imprecise query may be formed. The search system facing imprecise query may either return a lot of irrelevant material or ask additional questions. The ability to ask relevant questions is basic to any dialog. The best question should reduce ambiguity and lead to the maximum information gain, dividing the space of relevant concepts into two subspaces containing approximately the same number of concepts. Turing test is still beyond the reach of computer dialog systems, but a program that would ask interesting questions and guess what one has in mind at the human level of competence should be called "intelligent". It is not clear how such competence would result from computing power alone, as in the case of chess or other board games. Thus the 20 question game may be presented as a challenge to the artificial intelligence community, an intermediate step on the way to the full Turing test.

The present approach to the 20 question games [9] is based on predefined questions, with some elements of learning to determine how important the question is. A matrix of objects times questions is defined, initially with some values entered manually, and zero values representing unknown importance of questions for a given object. The program is placed in the Internet and learns from each new play what is the answers to a specific question, increasing the weight for (object, question) element if the player gave the expected answer, or decreasing it if the answer was different than expected. This approach is inflexible, relying on predefined questions, similar to the chatterbot guessing answers that fits to a template, without explicit semantic representation of concepts.

The algorithm based on concept description vectors selects the best possible question in the following way. Initially nothing is known, and for each keyword the information gain has to be calculated: assuming k discrete values and $P(kw = v_i)$ being the fraction of concepts for which the keyword kw has value v_i, the information in this probability distribution is $I(kw) = -\sum_{i=1}^{k} P(kw = v_i) \log P(kw = v_i)$. The vector of currently received answers A defines a subset of concepts $O(A)$ that are the most probable answers, with a uniform prior probability distribution $P(A) = 1/|O(A)|$. This distribution may be changed if additional information is accumulated from many games about the *a priori* probability of selecting various concepts. All vectors in $O(A)$ subset have zero distance in the subspace spanned by A keywords. To take into account the possibility of errors in the answers a larger subset $O(A_{+k})$ of concepts at a distance k from the $O(A)$ concepts may also be taken into account, with smaller probability $P(A) = 1/|O(A_{+k})|$. Select-

ing next keyword that maximizes $I(kw)$ a question that has a simple form is asked: "Is it related to ...", or "Can it be associated with ...". Questions could be formed in much more human-like way, but for our purpose this awkward form carries sufficient information.

In our computer experiments all concepts stored in the CDV matrix were parts of a single ontology restricted to animal kingdom to simplify the experiments. The first few steps based on binary splits give information gain close to 1, indicating that the two subsets have similar number of elements. Unfortunately CDV vectors created automatically are very sparse, with only 5-20 definite values (on average 8 throughout the whole set) out of several thousand keywords. As a result in the later stages of the game, in the reduced $O(A)$ subspaces, each answer to a question may eliminate only a few concepts. This requires either using other methods of limiting the number of concepts or improving information in the CDVs.

Three algorithms for the 20-question game have been implemented. The first one is based on the algorithm described above and is the simplest of the three. If there are keywords that have definite values for at least half of the concepts (are applicable to these concepts) in $O(A)$ subset choose the keyword that has the largest information index. Sample game is presented below. Answer and $I(kw)$ are given in parenthesis for each keyword used in the question; the concept "buffalo" was discovered after 12 questions.

- wing (0.635)[**NO**], coat (0.680)[**YES**]
- carnivore (0.623)[**NO**], hoof (0.674)[**YES**]
- ruminant (0.692)[**YES**]
- withers (0.566)[**NO**], bovine (0.629)[**NO**], antelope (0.679)[**NO**], goat (0.538)[**NO**], bovid (0.555)[**YES**]
- wild sheep (0.628)[**NO**], buffalo (0.681)[**OK**]

If the $\mathbf{S}(c,k)$ matrix in the $O(A)$ subspace has too few definite elements, the second algorithm is used, based on the most common feature. Choose the keyword that has definite value in the largest number of concepts, and reduce the subspace of candidate concepts depending on the answer for this keyword. Because even the most common feature is assigned to a small number of concepts only, this can be either a good or a bad choice. This methods implicitly defines a prior in favor of common concepts. Most frequent keyword are usually associated with the most common concepts, so if the user has chosen a common word rather than a rare word this is a good approach. An important fact here is that void and obvious features are not present in the matrix $\mathbf{S}(c,k)$. Filtering has been done initially for the whole matrix but it should be repeated after each reduction of the $O(A)$ subspace during the game. Sample game is presented below, won in 15 questions:

- throax (0.326)[**YES**], belly (0.441)[**YES**], coat (0.666)[**YES**], eutherian (0.178)[**YES**], carnivore (0.625)[**NO**]
- hoof (0.676)[**YES**], artiodactyl (0.673)[**YES**], ruminant (0.375)[**YES**], bovid (0.505)[**YES**], bovine (0.619)[**NO**]

- antelope (0.665)[**NO**], sheep (0.603)[**NO**], goat (0.595)[**NO**], wild sheep (0.628)[**NO**], buffalo (0.681)[**OK**]

Third algorithm is based on an associative matrix memory implemented by a neural network without hidden layers [10]. The matrix $\mathbf{S}(c,k)$ is treated here as a binary weight matrix that codes the existence of association between keywords $K(k)$ (inputs, row vectors) and concepts $C(c)$ (outputs). The main steps of the algorithm are:

1. Set all elements of C to 1 (all concepts are equally probable).
2. Calculate $K = C \cdot \mathbf{S}$ (keywords strength in concepts).
3. Find maximal value element Key=$\max_k K(k)$.
4. Ask the question about the keyword Key.
5. Set K(Key) $= 1$ or -1, depending on the yes or no answer.
6. Calculate $C = \mathbf{S} \cdot K$
7. Repeat steps $2-5$ until maximal element of Co indicates the answer.

The key in this algorithm is step 2. Here it is just a result of the vector times matrix product, but it can be replaced with other ways of choosing next keyword for query. K vector stores history of the answers and its values can be modified had the user made a mistake. Unfortunately we have no space here to analyze performance of all these algorithms here, they are presented only as an illustration of the usefulness of CDV representation.

5 Conclusions and plans

In this paper an important challenge has been stated: creating concept description vectors from analysis of information in dictionaries, text corpora and ontologies. Without such information NLP systems will never have sufficient prior knowledge to reach high level of linguistic competence. Several approaches were used to create CDVs using Wordnet dictionaries, ontologies and other information sources. Inferring even the simplest description, with CDV feature values that indicate which keywords may be applied to a given concept, proved to be difficult.

The 20 question game has been presented here as a next important step on the road to pass the Turing test and as a great test to increase precision of questions. Three algorithms based on CDVs have been presented for selection of the most informative questions. The quality of these algorithms in real games depends critically on the quality of CDVs. In collaboration with the Department of Technology in Education, Nicholaus Copernicus University (Torun, Poland), experiments are being conducted to determine human competence in the 20 question game and benchmark our algorithms against people in real games.

Several new ideas to improve the 20 question game algorithms are worth exploring. Similarity between CDV vectors may be used to define semantic

space areas of high concept density. Centers of such areas could represent the whole sets of concepts in the first steps of the algorithm and used as a single object with set of features common to all individual objects in the area, reducing the number of concepts/keywords to be processed. This may be achieved using either clusterization techniques or dimensionality reduction techniques based on latent semantic analysis. Performing principal component analysis for large matrices (ca. 3000 concepts and 10000 features) is computationally intensive. However, using the fact that the CDV matrices are very sparse (with only about 1% of non-zero values) an algorithm that performs all necessary calculations within minutes on an average PC may be formulated.

Further experiments along these lines are in progress. So far all large NLP projects, such as creation of Wordnet databases, relied heavily on human labor. Although our results on automatic creation of CDVs may be useful for some applications a lot of human corrections may be needed to create knowledge-rich lexical databases that are essential for the progress in many NLP subfields.

References

1. J. Weizenbaum, Computer Power and Human Reason: From Judgment to Calculation. W. H. Freeman & Co. New York, NY, USA 1976.
2. See transcripts at: http://www.loebner.net/Prizef/loebner-prize.html
3. H. Brandt-Pook, G.A. Fink, B. Hildebrandt, F. Kummert, and G. Sagerer. A Robust Dialogue System for Making an Appointment. In: Int. Conf. on Spoken Language Processing, Vol. 2, pp. 693-696, Philadelphia, PA, USA, 1996.
4. G. Hickok, D. Poeppel, Dorsal and ventral streams: A new framework for understanding aspects of the functional anatomy of language. Cognition, 92: 67-99, 2004.
5. W.B. Dolan, L. Vanderwende and S. Richardson, Automatically Deriving Structured Knowledge Base from On-line Dictionaries. PACLING 93, Pacific Assoc. for Computational Linguistics, pp. 5-14, 1993.
6. S. Richardson, Determining Similarity and Inferring Relations in a Lexical Knowledge Base. Ph.D. thesis, 187 p, The City University of New York, 1997.
7. L. Vanderwende, The Analysis of Noun Sequences using Semantic Information Extracted from On-Line Dictionaries. Ph.D. thesis, 312 p, Georgetown University, 1995.
8. C. Fellbaum (Ed), WordNet. An Electronic Lexical Database. MIT Press, 1998.
9. See www.20q.net
10. T. Kohonen, *Self-Organizing Maps.* Springer-Verlag, Heidelberg Berlin, 1995.

Automatic Scripts Retrieval and Its Possibilities for Social Sciences Support Applications

Yali Ge, Rafał Rzepka, Kenji Araki

Graduate School of Information Science and Technology,
Hokkaido University,
Kita-ku Kita 14-jo Nishi 9-chome, 060–0814 Sapporo, Japan

Abstract. This paper introduces our method for automatic Schankian-like scripts retrieval from the Internet resources and its preliminary results which might be interesting for Social Sciences researchers. We describe the first module of our system, which is supposed to automatically retrieve commonsensical knowledge from the Web resources by using web-mining techniques. It retrieves minimal "object — action — action" scripts which show humans' common activities changing due the origin of a webpage author. Such data can be used in fields of economics, psycholinguistics, sociolinguistics, psychology, sociology or in language education. By this paper we would like to make NLP researchers notice the potential of commonsense retrieval and encourage them to consider creating such tools for their languages.
Keywords: commonsense, web-mining, social sciences support.

1 Introduction

1.1 Need for Commonsense Retrieval

Nowadays, amount of information increases quickly with rapid growth of the Internet. As an enormous database, it is convenient if the information which humans need is acquired from the huge source of World Wide Web. Related information can be retrieved in a matter of seconds but it is mostly knowledge which we, humans, need. Obvious knowledge called "commonsense" does not become an object of search queries as we do not need it. However the computers are told to lack this knowledge which we gather our whole lives and it is a reason that people do not treat machines as intelligent partners, especially during conversations. There are several researches of gathering commonsense as CyC [1] or OpenMind Commonsense [2]. CyC contains over a million hand-crafted assertions [3] and OpenMind commonsense enabled construction of a 700,000 assertion commonsense knowledge base, gathered through a web community of collaborators. But they concentrate on manual or half-automatic processing and are done only for English language. As we assume that there is too much of such knowledge to be inputted, we try to

make this process automatic by using Web-mining techniques[1]. As we claim that Japanese language to have the best predispositions for such processing thanks to its particles, we concentrate on Japanese WWW resources. However

Table 1. Main Japanese particles and their functions

Particle	Role
WA	Topic-Indicating
GA	Linking-Indicating
NO	Possessive-Indicating
WO	Object-Indicating
NI	Direction-Indicating
DE	Place or Means of Action-Indicating
HE	Destination-Indicating
TO	Connective
MO	Addition-Indicating
YORI	Comparison-Indicating
NODE	Reason-Indicating
V-KARA	Reason Indicating
N-KARA	Lower Limit-Indicating
MADE	Upper Limit-Indicating
DEMO	Emphasis-Indicating

such research does not have to be restricted to Japanese, based on the same principles, an application could work with other languages – for example by using prepositions in English or regular expressions for non-gender counting in Polish. Before we started our commonsense retrieval project, we wanted to observe how is Japanese usable for our purposes and how the commonsense differs depending on a language. In this paper, we introduce our definition of a script, our algorithm, the basic system architecture, the experiment and the results. Then we compare Japanese results with Chinese, Polish and English. This paper introduces our preliminary system with small sets of data but we decided it is enough to introduce the results to the NLP colleagues from different language areas.

1.2 Definition of a Script

For retrieving scripts from WWW we had to simplify the original script [4] definition. Here, the scripts are the memory structures which summarize a typical phenomenon sequence into one frame, and it is defined as what expressed the standardized action which human is performing every day in the form of the chain of a phenomenon in alignment with the passage of time. For

[1] by which we mean here text-mining techniques using data sets being retrieved from WWW instantly.

example, the "restaurant script" can usually express that the customer "enters into restaurant", "sits down on a chair", "places an order", "eats some food", "pays money" and "leaves". As it is the very beginning of our research, we treat any verb chain as a script if it has at least one noun (object) with a particle joining a noun and a verb. Therefore the smallest unit as "watch the TV then go to bed" we also treat as a script and first we experimented with such simple units calling them "minimal scripts".

2 Our System

2.1 Algorithm

In the beginning of our research, we decided to work with nouns as keywords for collecting the scripts. We retrieved them from sentences and extracted the related sequences for creating dictionaries as verb dictionary, noun dictionary and n-gram dictionaries using 1,907,086 sentences WWW corpus retrieved by Larbin robot. The noun and verb dictionaries consist of 79,460 verbs and 134,189 nouns retrieved with help of morphological analyze tool for Japanese — ChaSen [5]. For creating scripts automatically, our system must search for the relationships between verbs and nouns and also between particular verbs. In this step, we used the verbs and nouns which had the highest occurrence, as they are often used in our everyday lives, for example *television*, *movie*, *food*. As mentioned above, we used Japanese language, which has useful grammar features like *rentaikei* where the verb suffix *te* usually joins verbs in a time sequence e.g. *gohan wo tabe*-**te** *neru* (to sleep after having a dinner). We can distinguish and change verb suffixes of Japanese to retrieve other, for example casual, strings for the commonsense, however here we concentrated on *rentaikei* for collecting actions following one after another. This process is the first step for the more complex categorization–using scripts creation proposed by Rzepka at al. [6]:

(dinner, chicken soup, apple...)–*wo* [2]-eat–*te*→
then
(cafeteria, restaurant, garden...)–*wo*–leave–*te*→...

Objects and actions (nouns and verbs joined by a particle) can easily create categories for objects. For example, "to eat" creates a food category, and "to leave" creates a place category. We also can use the pair of verbs to make a category smaller or combine to make a new script – "sit" and "fly" gather other objects than "sit" and "eat".

2.2 Architecture

Basically, our system's architecture for creating scripts can be summarized into the following processing steps:

[2] *wo*: object-indicating particle, see Table. 1.

a) The user inputs a noun as a keyword;
b) The system uses the web corpus for frequency check to retrieve 3 most frequent verbs following the keyword noun;
c) The most frequent particle between noun keyword and 3 most frequent verbs is discovered;
d) For creating bi-gram the system retrieves a list of most frequent verbs occurring after the previously chosen verb;
e) By using Google resources [7], the system checks if the noun-particle unit occurs with new verb-verb unit;
f) If yes — the minimal script is remembered in frequency order, as following:

$$\mathbf{Ms} = \mathbf{N} + \mathbf{P}_{\max} + \mathbf{V}_{\max 1} + \mathbf{V}_{\max 2}$$

N: *Noun keyword;*
$P_{\max}$: *the most frequent particle joining noun and verb;*
$V_{\max 1}$: *most frequent verb occurring after the N;*
$V_{\max 2}$: *most frequent verb occurring after* $V_{\max 1}$*;*

2.3 Experiment and Results

The steps a)-c) showed that text processing on our Web corpus takes too much time (average 52 seconds for a query) and for longer sequences (steps d and e) even bigger version of Web corpus (9,802,434 sentences) did not have enough data, therefore we decided to use Google API. Because of problems with Unicode we gave up Linux and made next steps on Windows system (Pentium 4, CPU 3GHz, 512 MB RAM) using Perl for the text processing and Java for connecting to Google servers. For example, retrieving an action following "watching TV" took 3 seconds (165 hits) but selecting the first pair of noun and verb still must be done with our corpus due to the Google limitations (1,000 searches per day) and takes up to one hour in very common keywords like *book*. We considered three most frequent "following actions" as commonsensical and inputted only common nouns (*subway, book, work, newspaper, baseball, bath, flu, swimming, water, bread, movie, television, music, home, car, word* and *meal*). We managed to confirm the naturalness of all retrieved trigrams based on them, though it was not always possible to collect the whole set of three verbs due to the Google limits and encoding errors, as in a "baseball" case in the examples below:
— fill with **water** and (boil, use, mix)
— get on the **subway** and (go back home, move, go)
— eating **meal** and (go back home, sleep, lose weight)
— play **baseball** (make living)
— take a **bath** and (*encoding error*, sleep, go out)
— hear a **music** and (feel happy, cry, to be touched)

— read a **book** and (spend time, learn, think)
— go back **home** and (sleep, see, have a meal)
— drive a **car** and (go back home, move, run away)
Here is a specific example of a case where "TV" was the keyword:
— terebi *wo* mi*te* neru — {*watching TV* → *sleeping*}: 165 times
— terebi *wo* mi*te* warau — {*watching TV* → *laughing*}: 90 times
— terebi *wo* mi*te* omou — {*watching TV* → *thinking*}: 74 times
Because the three above-mentioned actions were most frequent, we assumed that they are the minimal scripts. During using the Google API, we discovered that the data set for Google API apparently differs from the data set used by Google search engine but we confirmed that the percentage of occurrences is similar. The difference rate of Google API and Google search engine is shown in Fig.1.

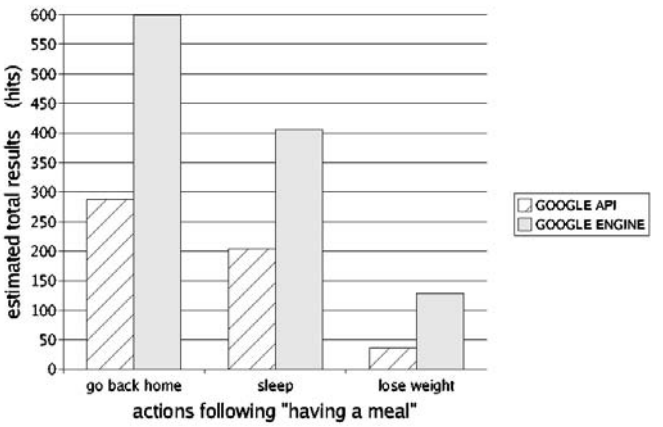


Fig. 1. Example: comparison for the data set of Google API and Google search engine.

3 Comparison with Other Languages

As mentioned in the introduction, we were interested how much the different language results differ from the Japanese ones. For example, in Japanese, we discovered that 3 most frequent actions which follow *watching a movie* are:
— watching movie → crying: 226 times
— watching movie → thinking: 135 times
— watching movie → being touched: 112 times
Then we used the Google searching engine for manual frequency checks for

English, Polish and Baidu searching engine for Chinese and we noticed that in most cases the proportions differ from Japanese results. The percentage of the example is shown in Fig.2. We confirmed that the 4 fullest sets of

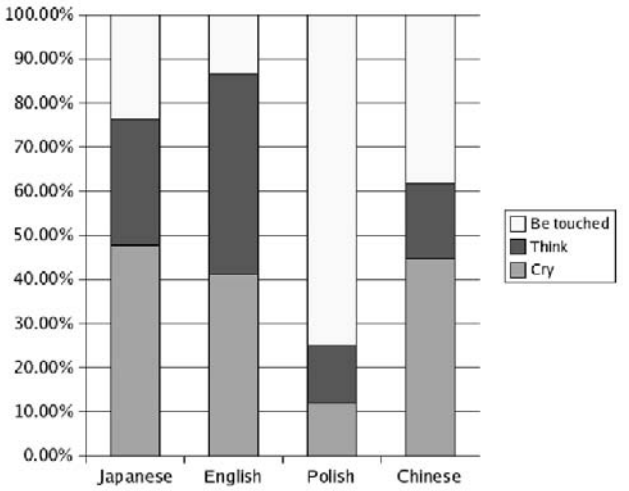


Fig. 2. Example: actions following "watching a movie" in Japanese, English, Polish and Chinese

trigrams had different proportions. Examined examples are shown in Tab. 1.

4 Possible Applications

Commonsensical knowledge data is very difficult to be collected manually as people in most cases do not realize things that happen too commonly. However, as our preliminary trials show, automatic retrieval of scripts might be useful to the researches in education, psycho-, socio-linguistics, economics (marketing), psychology and sociology. To suggest a few ideas:

- Language Education:
  — scripts can be used in automatic example sentences retrieval for foreign language students (they may differ depending on the context – if the categorization is successful);
  — when implemented into the talking agent, the script engine can correct the grammar of the learner and suggest the most common (natural) way of creating utterances;
- Economics and Marketing:
  — companies can retrieve differences of habits of their possible clients

Table 2. Proportions of actions in different languages

act1	verb2	JPN	ENG	POL	CHN
fill with water	boil	305	22	3	11900
	use	179	354	2	2
	mix	102	40	5	3710
take the subway	go home	120	14	7	311
	move	30	9	10	181
	went to	27	170	60	883
have a meal	go back home	288	6	0	769
	sleep	204	13	0	269
	lose weight	36	3780	48	1060
watch a movie	cry	226	1430	13	190
	think	135	1580	14	71
	be touched	112	467	81	162
drive a car	go back home	283	1590	517	7140
	move	180	125	17	67
	run away	64	276	35	117
hear a music	feel happy	150	424	1	318
	cry	119	752	99	659
	to be touched	82	1060	39	842
read a book	spend time	539	290	35	132
	study	392	588	41	34600
	think	240	2840	92	2590

depending on a culture;
— scripts made out of Internet resources are alive — the hit numbers can change together with customer's habits;

- Sociology, Anthropology:
  — especially useful for comparative studies about developed countries and societies changing due to the Internet's growth.

5 Conclusions

We have created minimal scripts automatically by using Web resources and Web-mining techniques. We described our experiment with Japanese and we confirmed that Japanese language has comparatively useful structure for the commonsense retrieval from the enormous data sets and partially confirmed that the results would differ if we make the automatic script retrieval for other languages. This time we only compared Japanese top 3 scripts with the same scripts in 3 other languages but as we checked manually these top 3 results may vary depending on the language. We also shared our ideas how the results could be utilized by social sciences if the algorithms for other languages were developed.

6 Future Work

Currently we concentrate on creating noun categories made by verbs and combine simple scripts into the bigger, more complicated and more elastic strings which would be useful for creating plans and discovering goals. Our next step is to automatically retrieve causative relationships also for resolving problems which appear when a script is broken and something unusual happens. We suppose that also these results will differ depending on a culture and we plan to prove it.

References

1. D. Lenat et al., **Common Sense Knowledge Database CYC**, 1995, http://www.opencyc.org/, http://www.cyc.com/
2. P. Singh., **The public acquisition of commonsense knowledge**, Proceedings of AAAI Spring Symposium on Acquiring (and Using) Linguistic (and World) Knowledge for Information Access. Palo Alto, CA: AAAI, 2002.
3. H.Liu.,P.Singh., **MAKEBELIEVE: Using Commonsense Knowledge to Generate Stories**, Edmonton, Alberta, Canada AAAI Press, 2002, pp.957-958.
4. Schank R.C. and Abelson R.P. **Scripts, Plans, Goals and Understanding: an Inquiry into Human Knowledge Structures (Chap. 1-3)**, L. Erlbaum, Hillsdale, NJ, 1977.
5. M.Asahara.,Y.Matsumoto., **Extended Models and Tools for High-performance Part-of-Speech Tagger**, COLING 2000,July,2000, pp.21-27.
6. Rzepka, R., Itoh, T., Araki, K. **Rethinking Plans and Scripts Realization in the Age of Web-mining**, IPSJ SIG Technical Report 2004-NL-162, pp 11-18, Sapporo, 2004.
7. http://www.google.com/apis/
8. Rzepka.R, Itoh.T., Araki.K., **Toward Fully Automatic Categorization for Commonsense Processing**, PPS 2004, Auckland, New Zealand, August 2004, 40-46.
9. P.Singh et al., **Open Mind Common Sense:Knowledge Acquisition from the General Public**, Lecture Notes in Computer Science: Vol. 2519. On the Move to Meaningful Internet Systems 2002: DOA/CoopIS/ODBASE 2002 pp. 1223-1237.
10. Shank R.C., **Dynamic Memory-A Theory of Reminding and Learning in computers and people**, Cambridge University Press, Cambridge, 1982.
11. Rzepka,R.,Araki,K.,Tochinai,K., **Bacterium Lingualis-The Web-Based Commonsensical Knowledge Discovery Method**.
12. Araki,K.,Tochinai,K., **Effectiveness of Natural Language Processing Method Using Inductive Learning.**, IASTED International conference Artificial Intelligence and Soft Computing, ACTA Press, (2001) Cancun.
13. Mueller,Erik t., **Understanding script-based stories using commonsense reasoning**, Cognitive Systems Research, 5(4), pp.307-340.
14. P.Singh., **The public acquisition of commonsense knowledge**, Proceedings of AAAI Spring Symposium on Acquiring (and Using)Linguistic (and World) Knowledge for information Access. Palo Alto, CA:AAAI, 2002.

The Analysis of the Unlabeled Samples of the Iron Age Glass Data

Karol Grudziński[1,2] and Maciej Karwowski[3,4]

[1] Department of Physics, Academy of Bydgoszcz
Plac Weyssenhoffa 11, 85-072 Bydgoszcz, Poland
E-mail: kagru@ab.edu.pl
[2] Institute of Applied Informatics
Wyższa Szkoła Gospodarki,
ul. Garbary 2, 85-070 Bydgoszcz, Poland
[3] Institute of Archeology, University of Rzeszów
Hoffmanowej 8, 35-016 Rzeszów, Poland
E-mail: mkar@univ.rzeszow.pl
[4] Vorgeschichliches Seminar, Philipps Universität
Marburg, Biegenstr. 11,
D-35037 Marburg, Germany

Abstract. The late iron age glass database consists of a significant proportion of the samples, classification of which is unknown. The data-mining methods such as the rule induction, the clusterization and the visualization are used in this paper to classify these samples to the one of the three main chronological periods (La Tene C1, La Tene C2, La Tene D1) of the glass artifacts. The results of the experiments performed with the C4.5 and the Ridor algorithms followed by the analysis conducted by domain experts indicate, that the unlabeled samples constitute a mixture of all classes in which LT C2 and LT D1 are in majority.

1 Introduction.

The late iron age glass database has been obtained during the realization of the interdisciplinary project "Celtic Glass Characterization", under the supervision of Prof. G. Trnka (Institute of Prehistory, University of Vienna) and Prof. P. Wobrauschek (Atomic Institute of the Austrian Universities in Vienna). A concentration of the following 26 chemical compounds have been measured using the Energy Dispersive X-ray Fluorescence Spectroscopy [1,2]: Na_2O, MgO, Al_2O_3, SiO_2, SO_3, K_2O, CaO, TiO_2, Cr_2O_3, MnO, Fe_2O_3, CoO, NiO, CuO, ZnO, SeO_3, Br_2O_7, Rb_2O, SrO, ZrO_2, MoO_3, CdO, $SnO2$, Sb_2O_3, BaO, and PbO. Three main chronological periods to be classified are of interest to archaeologists:

1. LT C1 - La Tene C1 period, 260 - 170 B.C.
2. LT C2 - La Tene C2 period, 170 - 110 B.C.
3. LT D1 - La Tene D1 period, 110 - 50 B.C.

The original database consists of the description of 555 glass samples. The glass composition has been measured usually in several places: on the original surface of the artifact and on the broken parts. Therefore, in the original database, several instances correspond to a single glass object and thus this data presents an interesting challenge, confusing some classification methods that rely on the similarity of samples. A thin corrosion layer always covers the surface of the archaeological glass and the broken parts are usually cleaner or at least less corroded. In [3] it has been shown, that it is possible to date the glass artifacts using a spectroscopic analysis. The most interesting result of the numerical experiments that had been performed there is the confirmation, that the place of the measurement (the original surface or the broken part of the glass) has no influence on the results of the analysis, and what follows, classification of the membership of the sample to the one of the chronological classes. This conclusion had been reached because a separate calculation performed on the surface and on the broken side of the glass artifact data leads to a similar classification accuracies attained by most classification systems used.

In the earlier experiments, which had been conducted by us on this dataset, the samples which had corresponded to the glass artifacts of the uncertain chronology or archaeologically had not belong to the rest of the data, were excluded from the original data. Assigning them to the one of the chronological periods is of a great importance for archaeologists.

The paper consists of 4 sections. First, the summary of the earlier work on the archaeological glass is done (this section). In the second section, the data is shortly described. The next section, titled 'Numerical Experiments', consists of two parts. In the first, the clustering analysis of the data with the k–means [4] model is performed. This permits to confront whether the class labels assigned during the preparation of the data more or less correspond to concepts learned by the k-means model. In the second part, the experiments with the J48.PART method, which is a version of the C4.5 rule induction system based on a C4.5 decision tree and the Ridor system available in the WEKA system [4], are presented. The last section concludes the paper.

2 The Data Description

In order to prepare a dataset for our computational studies, we have excluded from the original dataset a relatively small number of samples containing measurements of additional decorations on the glass. Those decorations are usually made of a different kind of glass than the main body of the glass artifact.

Since the chemical analysis has been made for the most glass samples in the several areas of the glass, several entries in the database may correspond to a single glass object. Usually two measurements are made on the surface and one on each of the two broken sides. Both the side and the surface

measurements have been included to prepare the dataset, as in [3] it has been shown that the experiments on the separated side and surface data lead to similar prediction accuracies.

The samples for which classification is unknown form two groups. To the first one belong 220 cases, which correspond to the glass artifacts of the uncertain chronology. The second group consists of only 18 samples, which do not fit the three La Tene chronological periods. This last group has not been included in our data.

The summary of the class membership for this data is given below:

1. LT C1 - La Tene C1 period, 260 - 170 B.C., 84 samples (16.7%)
2. LT C2 - La Tene C2 period, 170 - 110 B.C., 95 samples (18.9%)
3. LT D1 - La Tene D1 period, 110 - 50 B.C., 104 samples (20.7%)
4. X - the class of unknown chronology, 220 samples (43.7%)

The total number of cases in a dataset for our study is 503. The random choice value of the arbitrary class for this data equals 25% and the base rate value equals 43.7%.

3 Numerical Experiments

In this section the numerical experiments are performed to help decide to which class the unlabeled samples should be assigned. The first subsection is devoted to the clustering analysis. In the second part, the rule discovery analysis is performed.

3.1 A Clustering Analysis

In this section the clustering analysis has been performed to make sure that the class labels provided by domain experts are not significantly different from the classes learned by the k-means [4] algorithm. The calculation has been performed setting the number of clusters to four and the class X has been treated as a separate class. Visual inspection of this clustered data indicates, that the cluster 0 (in which the class 2 - LT C2 is in majority) the cluster 1 (the class 3 - LT D1) and the cluster 3 (the class X), form a one big cluster with all the member clusters significantly overlapping. The cluster two (LT C1) is well separated from the rest of the data. From the point of view of the overall number of samples wrongly clustered, the best is to create the three classes: LT C1, X, and LT C2 and the class (LT D1) should be more or less equally distributed among the classes LT C2 and X. Removing of the class X from the Fig. 1 allows us to see a beautiful coherent cluster of LT D1 which is covered by the samples from the class X. Hence, one can arrive at a conclusion that the most of the samples from the class X belong to the period LT D1 or LT C2. It is worth mentioning, that according

to some archaeologists, the periods LT D1 and LT C2 should not be treated separately but as the one chronological period LT C2/D1. Due to the lack of space the confirmation of this hypothesis with the computational intelligence methods will be left for further papers. Visual inspection of the clustered data, however gives preference for the membership of the class X neither to the LT C2 or LT D1 chronological period.

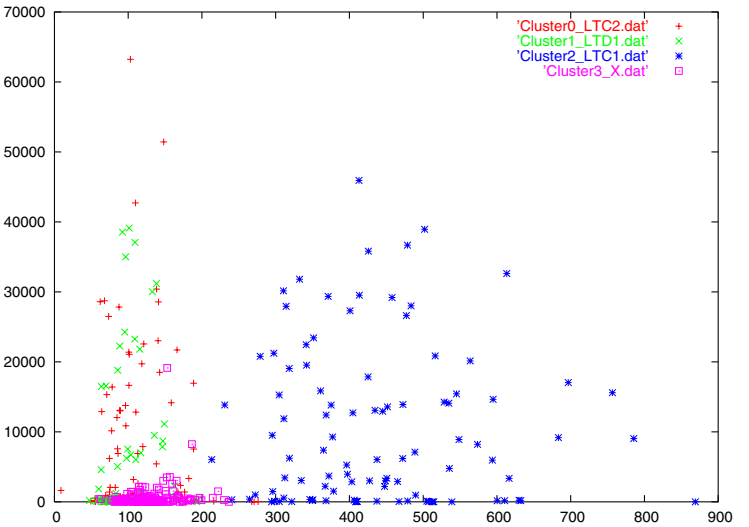


Fig. 1. The four clusters of the glass data obtained by running the k-means model. On the X axis the concentration of ZrO_2 and on the Y axis, Sb_2O_3 is marked.

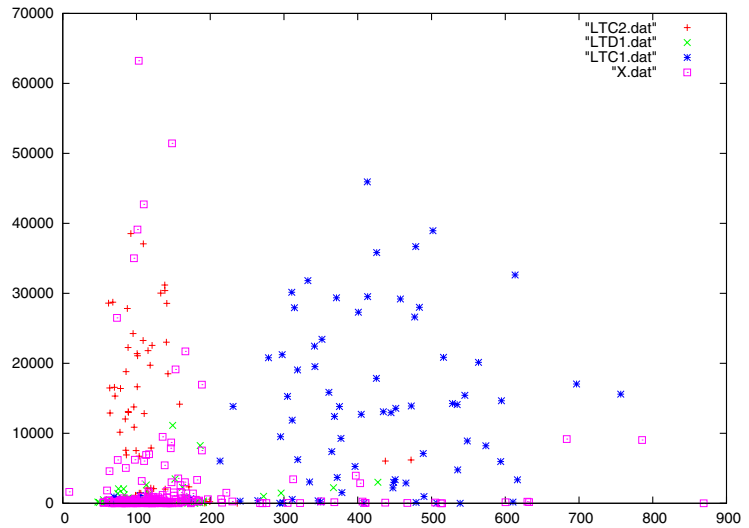


Fig. 2. Original data (the three classes labeled by domain experts and the class X).On the X axis the concentration of ZrO_2 and on the Y axis, Sb_2O_3 is marked.

On the Figure 2 the original data has been plotted. Comparing the Figure 1 and the Figure 2 one may arrive at a conclusion that the classes obtained by assigning labels by domain experts more or less correspond to learned concepts and that no significant misclassifications have been made during the data preparation process.

3.2 A Rule Discovery Analysis

Three rule induction experiments have been performed. First, the unlabeled cases have been treated as a separate class. Next, the calculations have been repeated 2 times, each time by treating the X class as LT C2 and LT D1. When performing the rule discovery experiments, the stress was put on the generation of the smallest rule sets (preferably one rule per class) with the highest prediction ability.

Below the smallest set of the four best rules obtained from the J48.PART system is given for the data, for which the unlabeled samples have been treated as the separate class. In the brackets the information about the number of covered cases followed by the number of errors each rule makes is provided.

1. IF $ZrO_2 \leq 199.38$ & $Sb_2O_3 \leq 6241.7$ & $CuO \geq 1704.94$ THEN X (234.0/92.0)
2. IF $ZrO_2 \geq 199.38$ THEN LT C1 (96.0/32.0)
3. IF $Sb_2O_3 \leq 3552.45$ THEN LT D1 (67.0/11.0)
4. Default: LT C2 (56.0/21.0)

In the all experiments, the listed rules have been obtained for the entire data. The best model has been selected by performing cross–validation test and observing the internal cross–validation on the training partitions. When treating X as the separate class, the training accuracy equals 66.6%, the cross–validation training accuracy – 66% and the cross–validation accuracy, providing information about generalization of the system, equals 64.6%.

A confusion matrix is a good source of valuable information about the data. 172 samples labeled as X have been classified correctly. This indicates, that the class X constitutes a coherent cluster, however without conducting further experiments in which X is treated as LT D1 and LT C2, one can not tell if this class could be treated as a new category of the glass artifacts.

Below, the rules for the data in which the unlabeled samples are treated as the LT C2 class are provided. The training accuracy is equal to 74.77%, the cross–validation training accuracy attains 72.84% and the cross–validation accuracy is equal to 72.76%.

1. IF $ZrO_2 \leq 229.92$ AND $MnO \leq 14452.22$ THEN LT C2 (302.0/63.0)
2. IF $MnO \leq 554.38$ THEN LT C1 (87.0/24.0)
3. Default: LT D1 (64.0/24.0)

Finally, the rules for the data in which X is treated as LT D1 class are given.

1. IF $Al_2O_3 \geq 17552.35$ AND $Sb_2O_3 \leq 9517.73$ THEN LT D1 (338.0/73.0)

2. IF $ZrO_2 \geq 199.38$ THEN LT C1 (72.0/16.0)
3. Default: LT C2 (43.0/11.0)

The results are slightly better. The training accuracy is equal to 74.82%, the cross–validation training accuracy equals 75.28% and the cross–validation accuracy attains 75.28%.

The listed above rules are very valuable as they provide information for the archaeologists about the attributes they should take under consideration in order to perform analysis.

3.3 The Prediction of the Unlabeled Cases

In this section an attempt to predict the classification of X taking the three LT classes as training data is made. This lets us to avoid some noise which is introduced by treating the entire X class as one of the LT classes in the experiments conducted so far. J48.PART and Ridor [4] models have been used in this study. In the calculation the stress was put on attaining the best possible generalization in cross-validation on the training set consisting of the three LT classes. The best J48.PART model produced 10 rules and attains the training accuracy of 99.3% and 85.5% in stratified 10-fold cross validation test on the training data. The J48.PART rules are:

1. IF $ZrO_2 \geq 199.38$ AND $CoO \leq 1603.74$ THEN LT C1 (66.0)
2. IF $Fe_2O_3 \geq 15738.8$ THEN LT C1 (8.0/1.0)
3. IF $SiO_2 \geq 826509.28$ AND $TiO_2 \leq 95.97$ THEN LT C1 (7.0)
4. IF $Sb_2O_3 \geq 3694.23$ AND $SnO_2 \leq 91.58$ THEN LT C2 (39.0)
5. IF $CuO \leq 2178.71$ AND $SeO_3 \leq 7.67$ AND $CaO \geq 77771.1$ THEN LT D1 (68.0)
6. IF $SnO_2 \geq 15.71$ AND $CuO \leq 5708.42$ AND $Cr_2O_3 \leq 39.63$ AND $Rb_2O \geq 34.74$ AND $MnO \leq 10009.21$ THEN LT C2 (26.0)
7. IF $Br_2O_7 \geq 34.38$ AND $PbO \geq 110.69$ AND $Br_2O_7 \leq 66.65$ AND $BaO \leq 400.44$ THEN LT D1 (29.0)
8. IF $Br_2O_7 \geq 24.23$ AND $ZnO \leq 173.94$ AND $MgO \leq 8890.94$ THEN LT C2 (30.0/1.0)
9. IF $SnO2 \geq 17.63$ THEN LT D1 (6.0)
10. Default: LT C1 (4.0)

The best Ridor model attained classification accuracy of 83.4% in cross-validation test on the training set and the training accuracy of 94.0%. The rules obtained by Ridor are:

1. IF $SrO \geq 801.02$ AND $ZnO \leq 189.33$ THEN LT D1
2. IF $Sb_2O_3 \geq 5692.38$ THEN LT C2
3. IF $NiO \geq 83.27$ AND $Rb_2O \geq 26.31$ AND $MnO \leq 10704.56$ THEN LT C2
4. IF $MgO \leq 4488.045$ AND $CuO \geq 2503.555$ THEN LT C2
5. IF $ZrO_2 \leq 199.55$ AND $SO_3 \leq 5615.025$ THEN LT D1
6. IF $Sb_2O_3 \geq 995.395$ AND $MnO \leq 19480.245$ THEN LT C2
7. IF $CuO \geq 2069.855$ AND $Rb_2O \geq 31.065$ THEN LT C2

8. IF $SiO_2 \geq 785497.5$ AND $NiO \geq 60.9$ THEN LT C2
9. IF $NiO \geq 61.99$ AND $Fe_2O_3 \leq 14167.76$ THEN LT D1
10. IF $Rb_2O \geq 23.215$ AND $MnO \leq 19539.77$ AND $TiO_2 \leq 120.67$ THEN LT C2
11. Default: LT C1

The summary of the results obtained in this section is given in Table 1.

Table 1. 10-fold Stratified Cross-Validation results of the 283 samples used as a training set for classification of the 220 unseen cases.

System	**# rules**	**Training Acc. (%)**	**Cross-Validation Acc. (%)**
J48.PART (C4.5)	10	99.3	85.5
Ridor	11	94.0	83.4

J48.PART assigned 41 unseen cases to the class LT C1 (1), 91 to the class LT C2 (2) and 88 cases have been assigned to the class LT D1 (3). Ridor assigned 37 unseen cases to the class LT C1 (1), 95 to the class LT C2 (2) and 88 cases have been assigned to the class LT D1 (3).

4 Conclusions and Further Work

The hypothesis formulated by the domain experts, that the unseen cases (the class X) is a mixture of all classes in which LT C2 and LT D1 classes are in majority has been confirmed by the J48.PART and the Ridor rule based methods. It seems that the best is not to treat the LT C2 and LT D1 as separate classes but as a one class LT C2/D1.

Dating glass artifacts is a highly nontrivial task and usually involves decades of study under a supervision of experts who are not easily available. The rules experts use to determine the chronological phase in which a given glass artifact has been manufactured can not be expressed in a form of concise list. Hence the great demand to acquire the knowledge of experts with the help of computational intelligence methods. This paper is therefore a next step towards building an expert system for the analysis of the archaeological glass data. It is planned to employ other computational intelligence methods and their ensembles in the analysis of this data in the future in order to get a higher confidence in machine learning predictions.

Acknowledgments: The research on chemical analysis of the archaeological glass was funded by the Austrian Science Foundation, project No. P12526-SPR. We are very grateful to our colleagues from the Atomic Institute in Vienna for making this data available to us. Karol Grudziński acknowledges support from the Department of Physics, Bydgoszcz Academy, which made participation of him in this conference possible.

References

1. P. Wobrauschek, G. Halmetschlager, S. Zamini, C. Jakubonis, G. Trnka, M. Karwowski. *Energy-Dispersive X-Ray Fluorescence Analysis of Celtic Glasses.* In: Special Millennium Issue on Cultural Heritage (Ed. E.S. Lindgren), X-Ray Spectrometry 29, pp. 25-33, 2000.
2. C. Jokubonis, P. Wobrauschek, S. Zamini, M. Karwowski, G. Trnka, P. Stadler. *Results of Quantitative Analysis of Celtic Glass Artifacts by Energy Dispersive X-ray Fluorescence Spectrometry.* Spectrochimica Acta Part B 58, pp.627-633, 2003.
3. Grudziński K., Karwowski M., Duch W. *Computational intelligence study of the iron age glass data.* International Conference on Artificial Neural Networks (ICANN) and International Conference on Neural Information Processing (ICONIP), Istanbul, June 2003, 17-20
4. I.H. Witten, E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations.* Morgan Kaufmann Publishers, 2000.

Discriminant versus Strong Rule Sets

Jerzy W. Grzymala-Busse[1,2], Witold J. Grzymala-Busse[3], and Jay Hamilton IV[4]

[1] Department of Electrical Engineering and Computer Science University of Kansas, Lawrence, KS 66045, USA
[2] Institute of Computer Science Polish Academy of Sciences, 01-237 Warsaw, Poland
[3] FilterLogix, Lawrence, KS 66044, USA
[4] Cerner Corporation, Kansas City, MO 64117, USA

Abstract. The main objective of our research was to compare two completely different approaches to rule induction. In the first approach, represented by the LEM2 rule induction algorithm, induced rules are discriminant, i.e., every concept is completely described and rules are consistent. In the second approach, represented by the IRIM rule induction algorithm, a few strong and simple rules are induced. These rules do not necessarily completely describe concepts and, in general, are inconsistent. Though LEM2 frequently outperforms IRIM, the difference in performance is, statistically, insignificant. Thus IRIM, inducing a few strong but simple rules is a new and interesting addition to the LERS data mining system.

1 Introduction

In this paper we study two approaches to rule induction. The first approach, traditionally used in machine learning, is based on inducing rule sets that are complete and consistent. The rule set is complete if every concept (class) is completely covered by rules. On the other hand, the rule set is consistent if it consistently covers (describes) every concept. Rule sets that are complete and consistent are called discriminant by R. Michalski [7]. Our disciminant rule sets were induced by the LEM2 (Learning from Examples Module, version 2). LEM2 is a component of the LERS (Learning from Examples based on Rough Sets) data mining system [2] and [3].

In the second approach we were not interested in inducing complete and consistent rule sets. Instead, induced rules would exhibit other properties, for example, strength, i.e., should cover a large number of training cases. These rules may cover small number of cases from other concepts and, even altogether, may be unable to cover all cases from the training data set. Such rules are called strong. In our experiments we used a new component of LERS called IRIM (Interesting Rule Induction Module) [4]. Rules induced by IRIM reveal important regularities in the data and may offer an interesting and surprising insight to experts in the domain area, hence the name of the module. IRIM resembles the ALL RULES algorithm, part of the LERS and the EXPLORE algorithm [10]. Inducing very few rules per concept, for

example, inducing only one rule per concept, was introduced in [7], see also [6]. A similar idea is rule truncation [7], i.e., removing weak rules. In generating decision trees a similar technique is called pruning [9].

The main objective of this research was to compare the performance of LEM2 with the performance of IRIM. As a result of running a number of experiments our conclusion is that LEM2 usually does a better job, but IRIM is worthy of attention because it produces not only interesting rules but also its performance is occasionally much better than the performance of LEM2. Furthermore, the difference in performance of LEM2 and IRIM is statistically insignificant.

2 Data Mining Tools

Both algorithms, LEM2 and IRIM are components of the same data mining system LERS and both use the same classification system for classifying unseen cases. The input data set for the LEM2 algorithm must be consistent, hence for inconsistent data LERS computes lower and upper approximations of all concepts and provides these sets to LEM2. For a detailed description of LEM2 see, e.g., [2]. An idea of approximation is taken from rough set theory [8]. Additionally, the input data with numerical attributes should be discretized before being processed by the LEM2 algorithm. Discretization is a process of converting numerical attributes into symbolic attributes, with intervals as symbolic values.

2.1 LEM2

LEM2 explores the search space of attribute-value pairs. In general, LEM2 computes a local covering and then converts it into a rule set. We will quote a few definitions to describe the LEM2 algorithm [2] and [3].

For an attribute-value pair $(a,\ v) = t$, a *block* of t, denoted by $[t]$, is a set of all cases from U such that for attribute a have value v. Let B be a nonempty lower or upper approximation of a concept represented by a decision-value pair (d, w). Set B *depends* on a set T of attribute-value pairs $t = (a, v)$ if and only if

$$\emptyset \neq [T] = \bigcap_{t \in T} [t] \subseteq B.$$

Set T is a *minimal complex* of B if and only if B depends on T and no proper subset T' of T exists such that B depends on T'. Let $\mathcal{T}$ be a nonempty collection of nonempty sets of attribute-value pairs. Then $\mathcal{T}$ is a *local covering* of B if and only if the following conditions are satisfied:

1. each member T of $\mathcal{T}$ is a minimal complex of B,
2. $\bigcup_{t \in \mathcal{T}} [T] = B$, and
3. $\mathcal{T}$ is minimal, i.e., $\mathcal{T}$ has the smallest possible number of members.

The procedure LEM2 is presented below.

Procedure LEM2
(**input**: a set B,
output: a single local covering $\mathcal{T}$ of set B);
begin
$G := B$;
$\mathcal{T} := \emptyset$;
while $G \neq \emptyset$
begin
$T := \emptyset$;
$T(G) := \{t | [t] \cap G \neq \emptyset\}$;
while $T = \emptyset$ **or** $[T] \not\subseteq B$
begin
select a pair $t \in T(G)$ such that $|[t] \cap G|$ is maximum; if a tie occurs, select a pair $t \in T(G)$ with the smallest cardinality of $[t]$;
if another tie occurs, select first pair;
$T := T \cup \{t\}$;
$G := [t] \cap G$;
$T(G) := \{t | [t] \cap G \neq \emptyset\}$;
$T(G) := T(G) - T$;
end {while}
for each $t \in T$ **do**
if $[T - \{t\}] \subseteq$ B **then** $T := T - \{t\}$;
$\mathcal{T} := \mathcal{T} \cup \{T\}$;
$G := B - \cup_{T \in \mathcal{T}} [T]$;
end {while};
for each $T \in \mathcal{T}$ **do**
if $\bigcup_{S \in \mathcal{T} - \{T\}} [S] = B$ **then** $\mathcal{T} := \mathcal{T} - \{T\}$;
end {procedure}.

For a set X, $|X|$ denotes the cardinality of X.

2.2 IRIM

Like LEM2, IRIM computes first the set of blocks for all attribute-value pairs. However, IRIM induces rules during discretization. IRIM recognizes integer and real numbers as values of attributes, and labels such attributes as numerical. For numerical attributes IRIM computes blocks in a different way than for symbolic attributes. First, it sorts all values of a numerical attribute. Then it computes cutpoints as averages for any two consecutive values of the sorted list. For each cutpoint c IRIM creates two blocks, the first block contains all cases for which values of the numerical attribute are

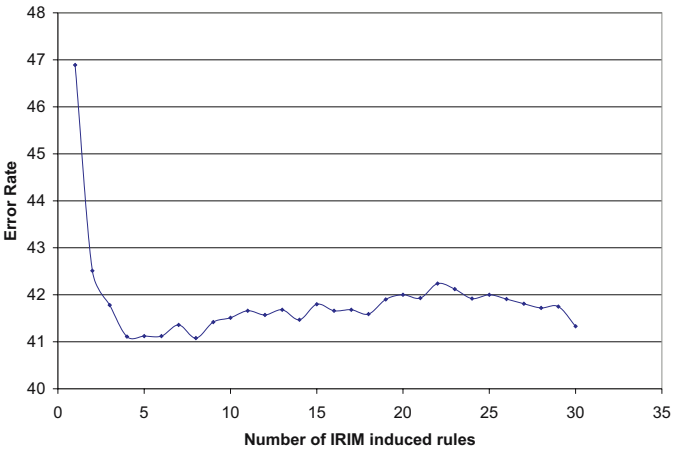


Fig. 1. Bupa data set

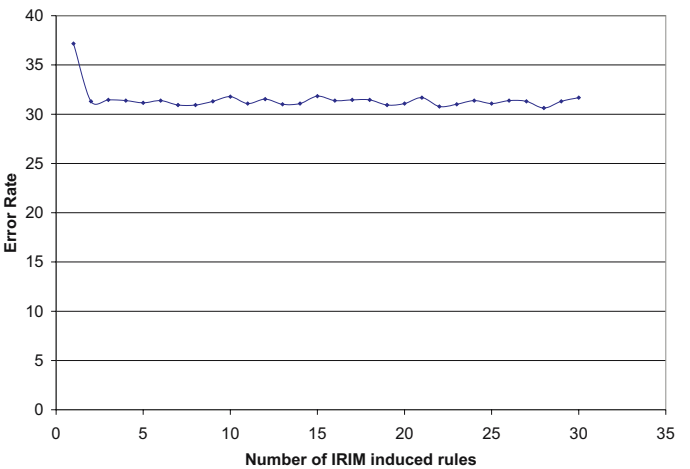


Fig. 2. German data set

smaller than c, the second block contains remaining cases, i.e., all cases for which values of the numerical attribute are larger than c. The search space of IRIM is the set of all blocks computed this way, together with blocks defined by symbolic attributes. Then IRIM combines attribute-value pairs relevant to a concept and creates all rules describing the concept, taking into account pre-defined, by the user, input parameters. To be more specific, for every concept IRIM creates a rule set with all rules satisfying the following three pre-defined input parameters:

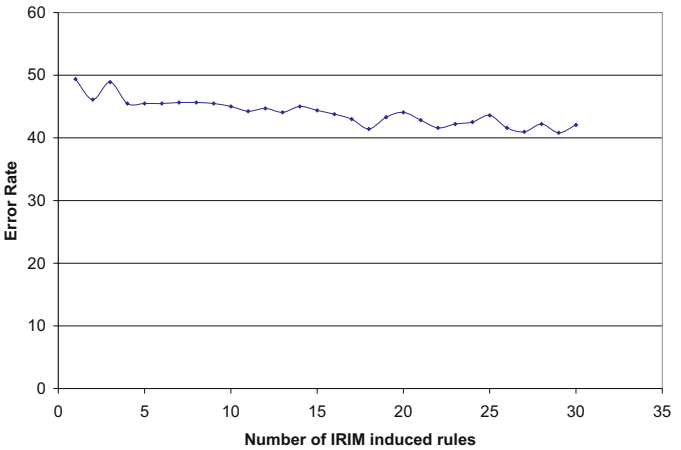


Fig. 3. Glass data set

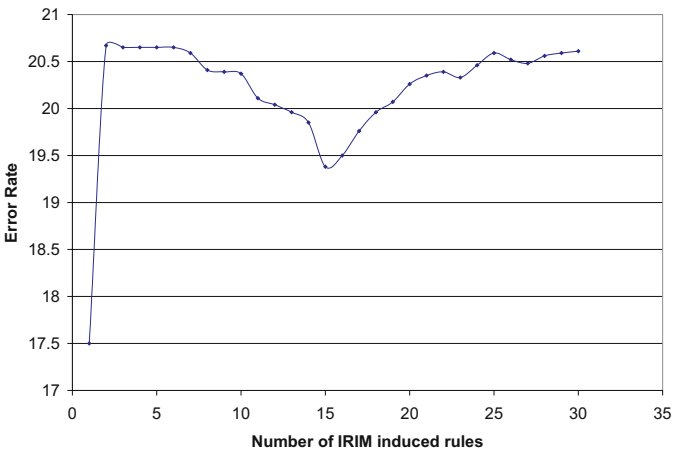


Fig. 4. Hepatitis data set

- the minimum rule length (i.e., number of rule conditions),
- the maximum rule length, and
- the minimum of conditional probability of the concept given rule domain.

The rule domain is the set of all cases satisfying the left hand side of the rule. For brevity, the minimum of conditional probability of the concept given rule domain will be called a ratio parameter.

The output of IRIM is the set of all rules satisfying input parameters. In general, IRIM generates very large rule sets. The worst time complexity of

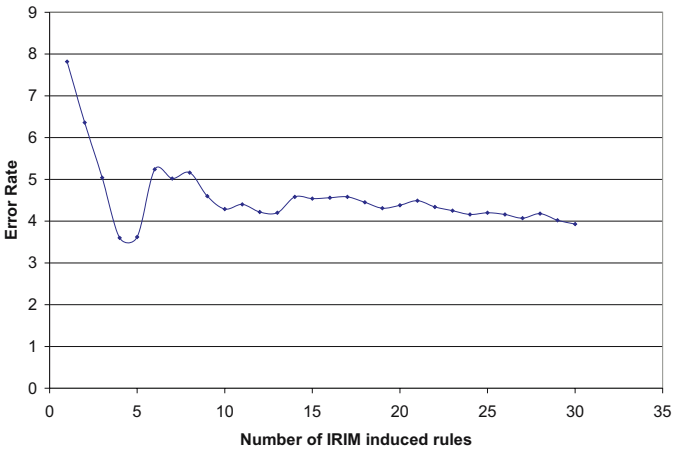


Fig. 5. Iris data set

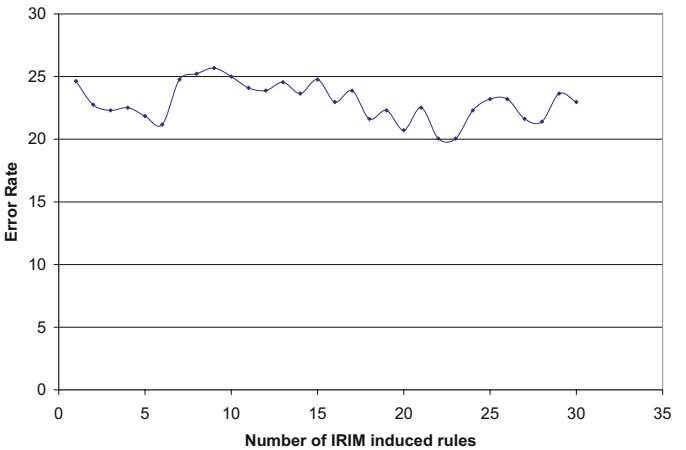


Fig. 6. Lymphography data set

IRIM is exponential with respect to the number of attributes. Hence, in our experiments, we selected the maximum rule length to be equal to two.

In addition, IRIM handles missing attribute values during rule induction. For any attribute with missing values, blocks are computed only from the existing attribute-value pairs, assuming that missing attribute values are lost (or erased) [5].

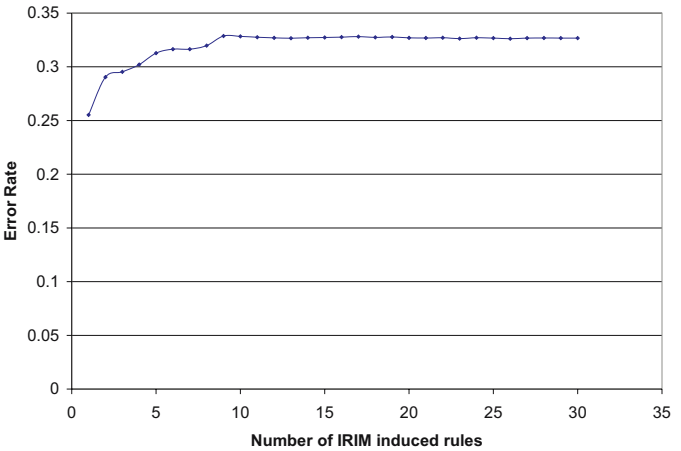


Fig. 7. Pima data set

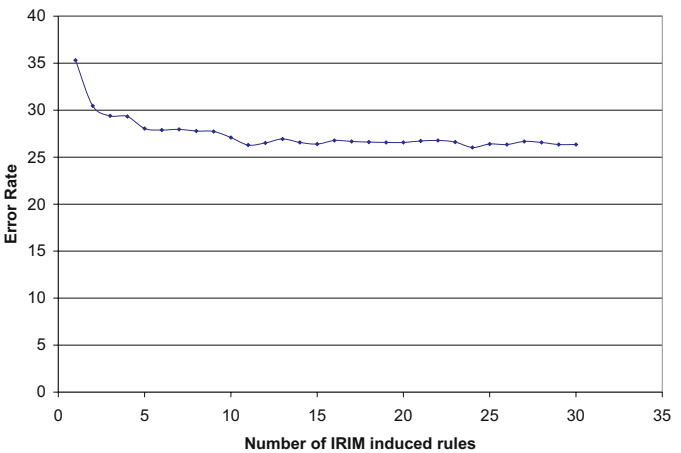


Fig. 8. Wisconsin data set

2.3 Classification system of LERS

The classification system of LERS [3] is a modification of the bucket brigade algorithm. The decision to which concept a case belongs is made on the basis of three factors: strength, specificity, and support. They are defined as follows: Strength is the total number of cases correctly classified by the rule during training. Specificity is the total number of attribute-value pairs on the left-hand side of the rule The third factor, Support, is defined as the sum of scores of all matching rules from the concept, where the score of the rule is

Table 1. Performance comparison for LEM2 and IRIM

Data set	Data statistics		Total number of rules	
	Number of caes	Number of concepts	LEM2	IRIM
BUPA	345	2	164	6
German	666	2	175	56
Glass	214	6	82	174
Hepatitis	155	2	21	2
Iris	150	3	11	15
Lymphography	148	4	26	16
Pima	768	2	287	2
Wisconsin	625	9	164	216

Table 2. Performance comparison for LEM2 and IRIM

Data set	LEM2		IRIM	
	Error rate	Standard deviation	Error rate	Standard deviation
BUPA	35.11	1.979	41.08	0.419
German	29.47	1.257	30.63	0.212
Glass	31.95	1.935	40.81	1.946
Hepatitis	15.07	1.690	17.5	1.054
Iris	3.26	0.201	3.62	0.419
Lymphography	20.13	2.396	21.96	2.135
Pima	34.11	0.805	25.52	7.316E-7
Wisconsin	20.51	0.682	26.03	0.333

the product of its strength and specificity. The concept for which the support is the largest is the winner and the case is classified as being a member of that concept. Every rule induced by LERS is preceded by three numbers: specificity, strength, and rule domain size.

3 Experiments

We conducted our experiments on eight data sets, all were taken from the Repository at the University of California, Irvine, CA. Some of the original data sets, used for our experiments, contained numerical attributes. These attributes were discretized using cluster analysis. Clusters were first formed

from data with numerical attributes. Then those clusters were projected on the attributes that originally were numerical. The resulting intervals were merged to reduce the number of intervals and, at the same time, to preserve consistency [1]. Even though IRIM can handle numerical attributes without preliminary discretization, all data sets were discretized for two reasons: to compare the performance of LEM2 and IRIM on the same data sets and because of the time complexity of running IRIM on data with numerical attributes.

The BUPA data set was created by the BUPA Medical Research Ltd. German stands for the German Credit data, donated by H. Hofmann and Glass stands for the Glass Identification data set, created by B. German. The hepatitis data set was donated by G. Gong. This data set was the only one with some missing attribute values. The iris data set was created by R. A.Fisher, while the lymphography data set was donated by I. Kononenko. Pima stand for Pima Indians Diabetes data set. Finally, Wisconsin stands for the Wisconsin breast cancer data set. Note that the Wisconsin data set was inconsistent, only certain rules [2] and [3], induced from the lower approximations of all concepts, were used in our experiments.

We conducted 30 ten-fold cross validation experiments, using LEM2 for rule induction on each of the eight data sets, to compute an error rate. Each of these 30 experiments returned slightly different error rate because partitioning of the original data set into ten subsets was different for each experiment.

Then we conducted another round of ten-fold cross-validation experiments, this time using IRIM to induce rules. The IRIM module was used with the following parameters:

- the minimum rule length was equal to one,
- the maximum rule length was equal to two,
- the ratio was set to 0.7,
- the rules induced by IRIM were ordered first according to their strength, then, for rules with the same strength, by ratio,
- the number of rules per concept was 1, 2,..., 30.

For Bupa, Hepatitis, Iris, Lymphography and Pima and for every number of rules per concept equal to 1, 2,..., 30, ten-fold cross validation experiments were conducted 30 times. Thus, for these data sets the total number of ten-fold cross validation experiments using IRIM was equal to 900. For German, Glass and Wisconsin and for every number of rules per concept equal to 1, 2,..., 30, ten-fold cross validation experiments were conducted three times because of time constraints. Hence the total number of ten-fold cross validation experiments for these data sets was equal to 90. Results of experiments are presented in Tables 1–2. Graphics of the error rate versus the number of rules per concept for IRIM are presented on Figures 1–8.

4 Conclusions

As displayed in Tables 1–2, LEM2 outperforms IRIM in seven out of eight cases. However, for the Pima data set, the performance of IRIM is significantly better than the performance of LEM2 (as follows from the standard statistical two-tailed test to compare two means, with 5% level of significance). Moreover, using the Wilcoxon matched-pairs signed-rank test on all eight data sets, the difference in performance between LEM2 and IRIM is insignificant (with a 5% level of significance, two-tailed test).

Taking into account that the peak performance of IRIM happens frequently for very few rules (in the case of PIMA it is a single rule per concept) and that these rules are extremely simple (with no more than two conditions), it is clear that IRIM should be considered as a viable addition to the LERS system.

References

1. Chmielewski, M. R. and Grzymala-Busse, J. W. Global discretization of continuous attributes as preprocessing for machine learning. Int. Journal of Approximate Reasoning 15 (1996), 319–331.
2. Grzymala-Busse, J. W.: LERS—A system for learning from examples based on rough sets. In Intelligent Decision Support. Handbook of Applications and Advances of the Rough Sets Theory. Slowinski, R. (ed.), Kluwer Academic Publishers, Dordrecht, Boston, London (1992) 3–18.
3. Grzymala-Busse, J. W.: A new version of the rule induction system LERS. Fundamenta Informaticae 31 (1997), 27–39.
4. Grzymala-Busse, J. W., Hamilton, J. and Hippe, Z. S.. Diagnosis of melanoma using IRIM, a data mining system. Proceedings of the ICAISC'2004, the Seventh International Conference on Artificial Intelligence and Soft Computing, Zakopane, Poland, June 7–11, 2004. Lecture Notes in Artificial Intelligence 3070, Springer-Verlag 2004, 996–1001.
5. Grzymala-Busse, J. W. and Wang A. Y.: Modified algorithms LEM1 and LEM2 for rule induction from data with missing attribute values. Proc. of the Fifth International Workshop on Rough Sets and Soft Computing (RSSC'97) at the Third Joint Conference on Information Sciences (JCIS'97), Research Triangle Park, NC, March 2–5, 1997, 69–72.
6. Holte R. C.:. Very simple classification rules perform well on most commonly used datasets. Machine Learning 11 (1993) 63–90.
7. Michalski, R. S., Mozetic, I., Hong, J. and Lavrac, N.: The AQ15 Inductive Learning System: An Overview and Experiments. Intelligent System Group, University of Illinois at Urbana-Champaign, ISG 86–20, 1986.
8. Pawlak, Z.: Rough Sets. International Journal of Computer and Information Sciences 11 (1982) 341–356.
9. Quinlan, J. R.: C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers, San Mateo, CA (1988).
10. Stefanowski, J.: Algorithms of Decision Rule Induction in Data Mining (in Polish). Poznan University of Technology Press, Poznan, Poland (2001).

IDARM — Mining of Indirect Association Rules

Przemysław Kazienko

Wrocław University of Technology, Institute of Applied Computer Science,
Wybrzeże S. Wyspiańskiego 27, 50-370 Wrocław, Poland

Abstract. Typical association rules, called in the paper "direct", reflect relationships existing between items that relatively often co-occur in common transactions. In the web domain items correspond to pages and transactions to user sessions. The main idea of new approach is to discover indirect associations existing between pages that rarely occur together but there are other, "third" pages, called transitive, with which they appear relatively frequently. Two types of indirect associations rules are described in the paper: partial indirect associations and complete ones. The former respect a single transitive page, while the latter cover all existing transitive pages. The presented IDARM algorithm extracts complete indirect association rules with their important measure — confidence, using pre-calculated direct rules.

1 Introduction

Mining association rules is one of the most important and widespread data mining techniques [20] also in the web environment. There are many papers related to algorithms for mining association rules: classical apriori [2,3], parallel ones based on apriori [4], Eclat [27], FP Growth [9]. An incremental algorithm FUP was presented in [6] and improved in [7]. Another incremental method DLG was proposed in [26].

The implementation of data mining into web domain (web mining) has been considered for several years [5]. Especially association rules discovered from HTTP server log data or user sessions (web usage mining) have been studied [1,19,25]. Web sessions are gathered without any user's involvement and they fully reflect user behavior while navigating throughout a web site. For that reason, sessions are important source of information about users.

Incremental algorithms appears to be the most suitable for the extraction of association rules in the web domain, taking into account the nature of web user behavior and great changeability of content and structure of the web. The problem of diversification between old and new user sessions was considered in [11,15].

Previous research work on mining indirect associations was carried out by Tan and Kumar [21,22] and next by Wan and An [23]. However, their indirect patterns differ from those presented in this paper. We have not assumed that two pages must not be directly correlated like Tan *et al.* did. Additionally, their rules need to have the assigned cardinality of the set of transitive pages

(called a mediator set) and this set is treated as one whole. In such approach both considered pages have to co-occur with a complete set of other pages instead of with a single transitive page. There are also no partial rules in that approach while in the described below concept they are components of complete rules. Tan *et al.* proposed that one pair of pages may possess many indirect rules with many mediator sets, which may overlap. In many application domains e.g. in recommendation systems [11], we need one measure that helps us to find out whether the considered page should or should not be suggested to a user on the particular page.

The presented in the paper method is part of the larger project concerning web recommendation systems [10–16].

2 Problem Description

Association rules are one of the most popular data mining methods. Besides many advantages, this method has also some limitations, which can lead to lose some vital information. Typical association rules focus on the co-occurrence of items (purchased products, visited web pages, etc) in transaction set. A single transaction may be a payment for purchased products and services, an order with the list of items, and user session in the web portal. The mutual independence of items (products, web pages) is one of the most important assumptions of the method but it is not fulfilled in the web environment. Web pages are connected each other with hyperlinks and they usually determine all possible navigational paths. A user admittedly is able to enter requested page address (URL) to his or her browser, nevertheless most navigations are done with hyperlinks designed by site authors. Thus, the web structure gravely restricts visited sets of pages (user sessions), which are not so independent one another as products in the store. To get to a page the user is often forced to navigate through other pages, e.g. home page, login page, etc. Additionally, web site content is usually organized by designer into thematic blocks, which not always are suitable for particular users.

For all these reasons, some personalized recommendation mechanisms are very useful in most web portals [15]. However, if they used typical mining techniques for association rules to historical user sessions [1,19,25], they would often only confirm "hard" connections resulting from hyperlinks and they may avoid some relationships between pages, which do not occur in the same user sessions (Fig. 1). It concerns especially pages not being connected directly with hyperlinks.

Original association rules (called in this paper *direct*) reflect relationships existing "within" user sessions (transactions). Standard parameters of direct association rules (support and confidence) have usually the greatest value for pages "hard" connected with hyperlinks because of the hypertext nature of the web. To explore significant associations between pages that rarely occur together, we suggest mining *indirect* association rules that occur "between"

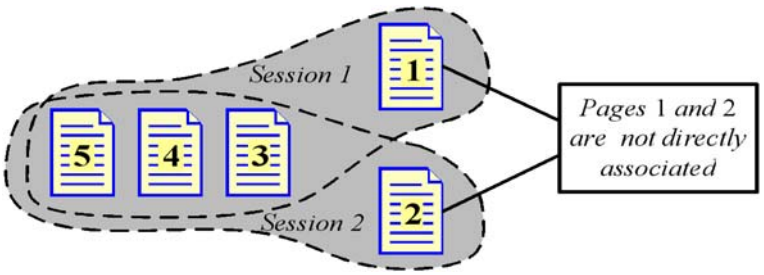


Fig. 1. Sessions with two documents (1 and 2), which are associated only indirectly

sessions. Two pages, which both separately, relatively frequently co-occur in sessions with another, third page, can be considered as "indirect associated". Similar idea was investigated in scientific citation analysis [8,18] and hyperlink (structure) analysis of the web [24]. Two scientific papers or web pages, in which another, third document (page) is cited (linked), are supposed to be similar. The analogue case occurs while two documents are cited or linked by the third one.

3 Direct Association Rules

Definition 1. Let d_i be an independent *web page* (document) and D be web site content (web page domain) that consists of independent web pages $d_i \in D$.

Definition 2. A set X of pages $d_i \in D$ is called a *pageset* X. Pageset does not contain repetitions: $\forall(d_i, d_j \in D)\ d_i, d_j \in X \Rightarrow d_i \neq d_j$. The number of pages in a *pageset* is called the *length* of the pageset. A pageset with the length k is denoted by *k-pageset*.

Definition 3. A user session S_i is the pageset containing all pages viewed by the user during one visit in the web site; $S_i \subseteq D$. S^S is the set of all user sessions gathered by the system. Each session must consist of at least two pages $\text{card}(S_i) \geq 2$. A session S_i contains the pageset X if and only if $X \subseteq S_i$.

Sessions correspond to transactions in typical data mining approach [3,20]. Note that pagesets and user sessions are unordered and without repetitions — we turn navigational sequences (paths) into sets. Additionally, user sessions may also be filtered to omit too short ones, which are not representative enough [12,15].

Definition 4. A *direct association rule* is the implication $X \rightarrow Y$, where $X \subseteq D$, $Y \subseteq D$ and $X \cap Y = \emptyset$. A direct association rule is described by two measures: *support* and *confidence*. The direct association rule $X \rightarrow Y$ has the support $sup(X \rightarrow Y) = sup(X \cup Y)/\,\text{card}(S^S)$; where $sup(X \cup Y)$ is the number of sessions S_i containing both X and Y; $X \cup Y \in S_i$. The confidence *con* for direct association rule $X \rightarrow Y$ is the probability that the session S_i containing X also contains Y: $con(X \rightarrow Y) = sup(X \cup Y)/\,sup(X)$; $sup(X)$

— the number of sessions that contain the pageset X. The pageset X is the *body* and Y is the *head* of the rule $X \rightarrow Y$.

Direct association rules represent regularities discovered from a large data set [2]. The problem of mining association rules is to extract all rules that are strong enough and have the support and confidence value greater than given thresholds: minimum direct support *supmin* and minimum direct confidence *conmin*.

In this paper we consider dependencies only between 1-pagesets — single web pages, in the web environment. For that reason, the 1-pageset X including d_i ($X = \{d_i\}$) will be denoted by d_i and a direct association rules from d_i to d_j is $d_i \rightarrow d_j$.

4 Indirect Association Rules

4.1 Partial Indirect Association Rules

Definition 5. *Partial indirect association rule* $d_i \rightarrow^{P\#} d_j, d_k$ is the *indirect* implication from d_i to d_j with respect to d_k, for which two direct association rules exist: $d_i \rightarrow d_k$ and $d_k \rightarrow d_j$ with $sup(d_i \rightarrow d_k) \geq supmin$, $con(d_i \rightarrow d_k) \geq conmin$ and $sup(d_k \rightarrow d_j) \geq supmin$, $con(d_k \rightarrow d_j) \geq conmin$, where d_i, d_j, $d_k \in D$; $d_i \neq d_j \neq d_k$. The page d_k, in the partial indirect association rule $d_i \rightarrow^{P\#} d_j, d_k$, is called *the transitive page* (Fig. 2).

Please note that for the chosen pair of pages d_i, d_j there may be many transitive pages d_k and as a result many partial indirect association rules $d_i \rightarrow^{P\#} d_j, d_k$.

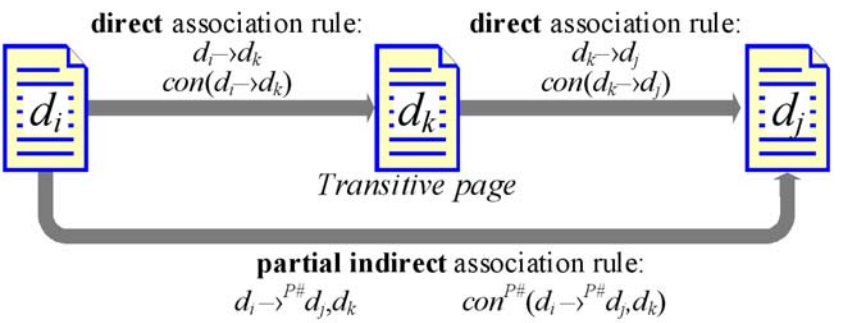


Fig. 2. Indirect association between two web pages

Each indirect association rule is described by *the partial indirect confidence* $con^{P\#}(d_i \rightarrow^{P\#} d_j, d_k)$, as follows:

$$con^{P\#}(d_i \rightarrow^{P\#} d_j,\ d_k) = con(d_i \rightarrow d_k) * con(d_k \rightarrow d_j) \tag{1}$$

The partial indirect confidence is calculated using direct confidence rather than source user session data. For that reason, the computational complexity of partial indirect rule mining is much less than for direct ones. Pages d_i, d_j do not need to have any common session, but rules $d_i \rightarrow d_k$ and $d_k \rightarrow d_j$ need to be "strong" enough, so that $con(d_i \rightarrow d_k)$ and $con(d_k \rightarrow d_j)$ exceed *conmin*.

Other functions instead of multiplication in (1) were considered like minimum and maximum, arithmetical mean and weighted mean [17]. Multiplication delivers the smallest values (in average even 1/10 compared to values of maximum function) but it has the best discrimination abilities at the same time — the standard deviation doubles the average while for other functions standard deviation is less than the average. Multiplication can be justified in terms of probabilities, if we remind that direct confidence (def. 4) is the conditional probability.

A partial indirect rule $d_i \rightarrow^{P\#} d_j, d_k$ reflects one indirect association existing between d_i and d_j so no direct association $d_i \rightarrow d_j$ is needed, although it may exist. The condition of non-existence of direct association is prior assumption in indirect rules proposed in [21–23].

The rule $d_i \rightarrow^{P\#} d_j, d_k$ also differs from two direct rules, which only look similarly: $\{d_i, d_k\} \rightarrow d_j$, and $d_i \rightarrow \{d_j, d_k\}$. Note that these direct rules respect only common user sessions that contain all three pages d_i, d_j, d_k. In opposite, the partial indirect rule $d_i \rightarrow^{P\#} d_j, d_k$ exploits common sessions of d_i, d_k and separately sessions with d_k, d_j. These two sets of sessions do not even need to overlap.

Definition 6. The set of all possible transitive pages d_k for which partial indirect association rules from d_i to d_j exists, is called T_{ij}.

Note that T_{ij} is not the same set as T_{ji}.

4.2 Complete Indirect Association Rules

Definition 7. *Complete indirect association rule* $d_i \rightarrow^{\#} d_j$ aggregates all partial indirect association rules from d_i to d_j with respect to all existing transitive pages $d_k \in T_{ij}$ and it is characterized by *complete indirect confidence* — $con^{\#}(d_i \rightarrow^{\#} d_j)$:

$$con^{\#}(d_i \rightarrow^{\#} d_j) = \frac{\sum_{k=1}^{\mathrm{card}(T_{ij})} con^{P\#}(d_i \rightarrow^{P\#} d_j,\ d_k)}{max_T}, d_k \in T_{ij}, \qquad (2)$$

where $max_T = \max_{d_i, d_j \in D} (\mathrm{card}\,(T_{ij}))$.

A complete indirect association rule from d_i to d_j exists if and only if it exists at least one partial indirect association rule from d_i to d_j.

Only indirect rules with complete indirect confidence greater than the given confidence threshold — *iconmin* are accepted. According to (1), there is no point in setting *iconmin* with the value less than the square of the appropriate threshold for direct rules divided by max_T: $iconmin \geq conmin^2 \,/\, max_T$.

The concept of partial indirect rules (1) enables the introduction of the threshold to partial indirect confidence — *piconmin* to exclude weak partial rules. However, *iconmin* is more general than *piconmin* so the former appears to be the better filtering factor.

Note, that complete indirect association rules are not symmetric: the rule $d_i \rightarrow^{\#} d_j$ may exists but the reverse one $d_j \rightarrow^{\#} d_i$ not necessarily. It results from features of partial indirect associations and direct associations, which also are not symmetric.

The normalization — the denominator max_T in (2) — ensures the range $[0, 1]$ to be the domain for complete indirect confidence. However, it also causes most complete confidence values to be less than equivalent direct ones. max_T represents "global" normalization, while using $\mathrm{card}(T_{ij})$ in the denominator we would obtain "local" normalization. Values of complete confidence are on average more than 10 times less at global normalization than at local one. According to experiments performed in the real e-commerce environment (4,242 web pages, 16,127 user sessions) typical value of max_T is about 250 while the average $\mathrm{card}(T_{ij})$ is about 10–20, depending on *minsup*.

5 Mining Indirect Association Rules, IDARM Algorithm

The discovery of indirect rules is performed in two main stages (Fig. 3): extraction of direct rules and mining indirect ones. The mining of direct association rules was considered in many papers [2–4,9,20,27]. There were distinguished two main approaches: horizontal and vertical [20]. In presented approach, we consider only simple direct rules: between 1-pagesets — single web pages and the choice between horizontal and vertical mining is not crucial. Nevertheless, we need to apply any algorithm for direct association rule mining at the first stage of the whole process. Taking into account the environment (sessions of web users) most suitable are incremental algorithms [6,7,26].

Due to frequent modifications of web pages, especially hyperlinks, typical user behavior and user sessions tend to change over time. The inclusion of time factor into direct rule mining was considered in [11]. Older sessions are smothered during confidence calculation, according to how much time went by between the beginning of session and the processing time.

The IDARM (*In-Direct Association Rules Miner*) algorithm was introduced to extract complete indirect association rules and their complete indirect confidence from direct ones. The IDARM algorithm consists of seven following steps, including 3 loops (Fig. 4):

1. Select all transitive pages d_k ($v_k^+ > 0$ and $v_k^- > 0$). For each transitive page d_k repeat steps 2–6.

2. Get all direct rules $d_i \rightarrow d_k$ and $d_k \rightarrow d_j$ for the given transitive page d_k. I_k is the set of pages d_i and J_k — the set of pages d_j, respectively.

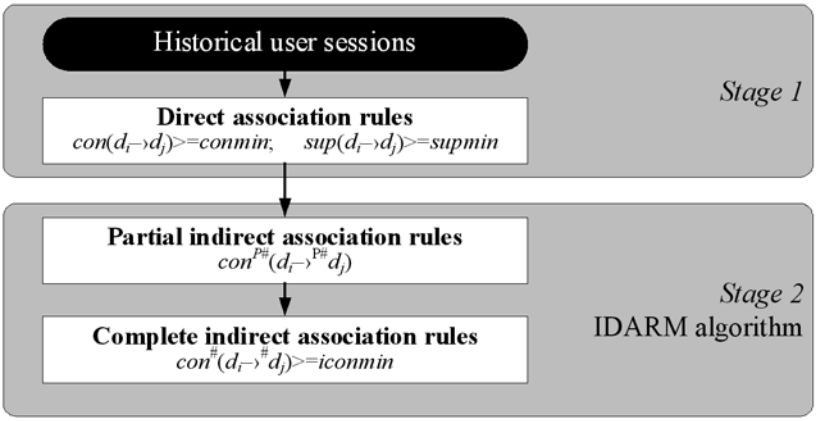


Fig. 3. Process of discovering association rules

3. For each "preceding" page d_i from the set I_k repeat steps 4–6.

4. For each "following" page d_j from the set J_k repeat steps 5–6. Create a complete association rule $d_i \rightarrow^{\#} d_j$ and assign $con^{\#}(d_i \rightarrow^{\#} d_j) = 0$, if such a rule does not yet exist.

5. Calculate partial confidence for the rule $con^{P\#}(d_i \rightarrow^{P\#} d_j,\ d_k)$ using (1).

6. Increase complete indirect confidence:

$$con^{\#}(d_i \rightarrow^{\#} d_j) = con^{\#}(d_i \rightarrow^{\#} d_j) + con^{P\#}(d_i \rightarrow^{P\#} d_j, d_k).$$

7. Estimate max_T. Divide all $con^{\#}(d_i \rightarrow^{\#} d_j)$ by max_T for normalization, according to (2).

The IDARM algorithm exploits the following property of direct association rules: to extract all partial indirect association rules, in which the page d_k is transitive, we only need to find all rules $d_i \rightarrow d_k$ (the set I_k) and $d_k \rightarrow d_j$ (the set J_k). Joining every direct rule from the set I_k with every rule from J_k we obtain all partial indirect rules with respect to d_k.

There are three loops in the IDARM algorithm, so we can estimate the primary complexity of the algorithm as $O(n^3)$, where n is the number of web pages visited during all user sessions. However, two internal loops operate on direct association rules and such a rule (we need its confidence value) is accessed only once. The complexity of the algorithm is then $O(mn)$, where m — the number of processed direct rules. Note, that the maximum value of m is $2n(n-1)$. Nevertheless, the reasonable value of m reached several dozen thousands according to the experiments performed on the set of 4,242 web pages (n^2 is about 18 billions). Thus m is nearly three orders of magnitude smaller than n^2 and we find out that $m \ll n^2$. Additionally, taking into account access to data, which is one of the most time consuming factor, the IDARM algorithm reaches out just for confidence values of direct rules and only once, so the complexity is then linear — $O(m)$.

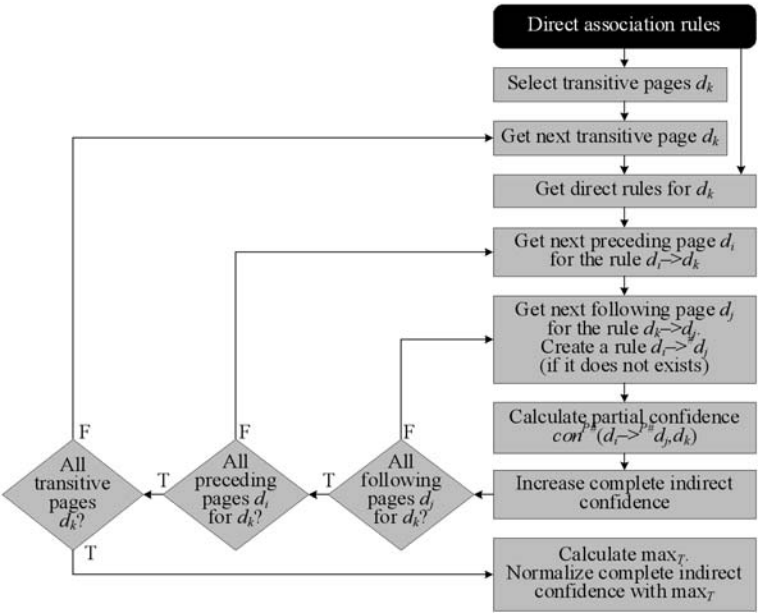


Fig. 4. Steps of the IDARM algorithm

6 Transitive Sets

The concept of partial indirect rules with single transitive page can be quite easily extended to indirect rules with the set of transitive elements. In such approach we need to replace d_k with D_k. Thus, we can modify definition 5.

Definition 8. *Partial indirect association rule with the set of transitive elements* $d_i \rightarrow^{P\#} d_j, D_k$ is the indirect implication from d_i to d_j with respect to the set D_k, for which two direct association rules exist: $d_i \rightarrow D_k$ and $D_k \rightarrow d_j$ with $sup(d_i \rightarrow D_k) \geq supmin$, $con(d_i \rightarrow D_k) \geq conmin$ and $sup(D_k \rightarrow d_j) \geq supmin$, $con(D_k \rightarrow d_j) \geq conmin$, where $d_i, d_j \in D$; $D_k \subset D$; $d_i, d_j \notin D_k$; $d_i \neq d_j$.

Note that no change is needed in (1). Nevertheless, the conversion of transitive pages into sets has significant consequences. The way of combination of all partial rules consistent with the definition 8 into complete indirect rules (def. 7) is not obvious due to the potential existence of many partial rules with transitive sets of different cardinalities. Naturally, these sets would often overlap each other and even they cover one another because for every set of cardinality K we have total $2^K - 2$ proper and non-empty subsets and the same number of different partial rules.

If we determine the way of calculation of complete rules we will only need to properly correct the sixth step of the IDARM algorithm. The remaining steps may be left without changes.

7 Conclusions and Future Work

Indirect association rules reflect relationships existing between transactions (user sessions). Owing to the presented IDARM algorithm, we obtain complete indirect rules with their complete indirect confidence. The algorithm exploits pre-calculated direct rules rather than raw user session data.

Indirect rules may not only confirm and strengthen direct relationships but they also often link objects not related with direct rules. In the web environment, they can help to go outside of typical user navigational paths coming form hyperlinks, so they reveal many associations out of reach for direct rule mining. For that reason indirect rules are useful in recommendation systems because they extend ranking lists and add to them non-trivial information.

Additionally, navigational patterns obtained from association rules can be combined with other data sources like relational databases and the textual content of web pages [14]. Such integration will be considered in future work.

References

1. Adomavicius G., Tuzhilin A. (2001) Using Data Mining Methods to Build Customer Profiles. IEEE Computer, **34** (2), 74–82.
2. Agrawal R., Imieliński T., Swami A. (1993) Mining association rules between sets of items in large databases. Proc. of the 1993 ACM SIGMOD Int. Conf. on Management of Data, Washington D.C., ACM Press, 207–216.
3. Agrawal R, Srikant R. (1994) Fast Algorithms for Mining Association Rules. Proc. of the 20^{th} International Conf. on Very Large Databases, 487–499.
4. Agrawal R., Shafer J.C. (1996) Parallel Mining of Association Rules. IEEE Transactions on Knowledge and Data Engineering, **8** (6), 962–969.
5. Boley D. et al. (1999) Document Categorization and Query Generation on the World Wide Web Using WebACE. Artificial Intelligence Review **13** (5–6), 365–391.
6. Cheung D. W. L. et al (1996) Maintenance of Discovered Association Rules in Large Databases: An Incremental Updating Technique. Proc. of the 12^{th} Int. Conf. on Data Engineering, New Orleans, IEEE Computer Society, 106–114.
7. Cheung D. W. L., Lee S. D., Kao B. (1997) A General Incremental Technique for Maintaining Discovered Association Rules. Fifth International Conference on Database Systems for Advanced Applications (DASFAA), Melbourne, Advanced Database Research and Development Series 6 World Scientific, 185–194.
8. Goodrum A. et al. (2001), Scholarly publishing in the Internet age: a citation analysis of computer science literature. Information Processing and Management **37** (5), 661–675.
9. Han J., Pei J., Yin Y. (2000) Mining Frequent Patterns without Candidate Generation. ACM SIGMOD Int. Conf. on Management of Data, 1–12.
10. Kazienko P. (2004) Multi-agent Web Recommendation Method Based on Indirect Association Rules. KES'2004, 8^{th} Int. Conf. on Knowledge-Based Intelligent Information & Engineering Systems, Wellington, New Zealand, Springer Verlag, LNAI **3214**, 1157–1164.

11. Kazienko P. (2004) Product Recommendation in E-Commerce Using Direct and Indirect Confidence for Historical User Sessions. DS'04. 7^{th} Int. Conference on Discovery Science, Padova, Italy, Springer Verlag, LNAI **3245**, 255–269.
12. Kazienko P., Kiewra M. (2003) Link Recommendation Method Based on Web Content and Usage Mining. Proc. of the International IIS: IIPWM'03 Conference, Zakopane, Advances in Soft Computing, Springer Verlag, 529–534.
13. Kazienko P., Kiewra M. (2003) ROSA — Multi-agent System for Web Services Personalization. AWIC 2003, First Atlantic Web Intelligence Conference Proceedings, Madrid, Spain, Springer Verlag, LNAI **2663**, 297–306.
14. Kazienko P., Kiewra M. (2004) Integration of Relational Databases and Web Site Content for Product and Page Recommendation. IDEAS'04, 8^{th} International Database Engineering & Applications Symposium, Coimbra, Portugal, IEEE Computer Society, 111–116.
15. Kazienko P., Kiewra M. (2004) Personalized Recommendation of Web Pages. Chapter 10 in: Nguyen T. (ed.) Intelligent Technologies for Inconsistent Knowledge Processing. Advanced Knowledge International, Adelaide, South Australia, 163–183.
16. Kazienko P., Kołodziejski M. (2005) WindOwls — Adaptive System for the Integration of Recommendation Methods in E-commerce. to appear.
17. Kazienko P., Matrejek M. (2005) Adjustment of Indirect Association Rules for the Web. SOFSEM 2005, Liptovsky Jan, Slovak Republic, Springer Verlag, LNCS **3381**, 211–220.
18. Lawrence S., Giles C. L., Bollacker K. (1999) Digital Libraries and Autonomous Citation Indexing. IEEE Computer **32** (6), 67–71.
19. Mobasher B., Cooley R., Srivastava J. (2000) Automatic Personalization Based on Web Usage Mining. Communications of the ACM, **43** (8), 142–151.
20. Morzy T., Zakrzewicz M. (2003) Data mining. Chapter 11 in Błażewicz J. et al. (eds): Handbook on Data Management in Information Systems. Springer Verlag, Berlin Heidelberg New York, 487–565.
21. Tan P.-N., Kumar V. (2002) Mining Indirect Associations in Web Data. WEBKDD 2001. Springer Verlag LNCS **2356**, 145–166.
22. Tan P.-N., Kumar V., Srivastava J. (2000) Indirect Association: Mining Higher Order Dependencies in Data. PKDD 2000, Springer Verlag, LNCS **1910**, 632–637.
23. Wan Q., An A. (2003) Efficient Mining of Indirect Associations Using HI-Mine. 16^{th} Conf. of the Canadian Society for Computational Studies of Intelligence, AI 2003, Halifax, Canada, Springer Verlag, LNCS **2671**, 206–221.
24. Weiss R. et al. (1996) HyPursuit: A Hierarchical Network Search Engine that Exploits Content-Link Hypertext Clustering. Hypertext'96. 7^{th} ACM Conf. on Hypertext, ACM Press, 180–193.
25. Yang H., Parthasarathy S. (2003) On the Use of Constrained Associations for Web Log Mining. WEBKDD 2002, Springer Verlag, LNCS **2703**, 100–118.
26. Yen S. J., Chen A. L. P. (1996) An Efficient Approach to Discovering Knowledge from Large Databases. 4^{th} Int. Conf. on Parallel and Distributed Information Systems, Miami Beach, Florida, IEEE Computer Society, 8–18.
27. Zaki M. J., Parathasarathy S., Li W. (1997) A Localized Algorithm for Parallel Association Mining. SPAA'97, 9^{th} Annual ACM Symposium on Parallel Algorithms and Architectures, Newport, Rhode Island, USA, 321–330.

Building a Concept Hierarchy from a Distance Matrix

Huang-Cheng Kuo[1] and Jen-Peng Huang[2]

[1] Department of Computer Science and Information Engineering
National Chiayi University, Taiwan 600
hckuo@mail.ncyu.edu.tw
[2] Department of Information Management
Southern Taiwan University of Technology, Taiwan 710
jehuang@mail.stut.edu.tw

Abstract. Concept hierarchies are important in many generalized data mining applications, such as multiple level association rule mining. In literature, concept hierarchy is usually given by domain experts. In this paper, we propose algorithms to automatically build a concept hierarchy from a provided distance matrix. Our approach is modifying the traditional hierarchical clustering algorithms. For the purpose of algorithm evaluation, a distance matrix is derived from the concept hierarchy built by our algorithm. Root mean squared error between the provided distant matrix and the derived distance matrix is used as evaluation criterion. We compare the traditional hierarchical clustering and our modified algorithm under three strategies of computing cluster distance, namely single link, average link, and complete link. Empirical results show that the traditional algorithm under complete link strategy performs better than the other strategies. Our modified algorithms perform almost the same under the three strategies; and our algorithms perform better than the traditional algorithms under various situations.

1 Introduction

Generalization on nominal data is frequently studied, such as mining multiple level association rules, by means of a concept hierarchy [8,5,3]. In a concept hierarchy of categories, the similarity between two categories is reflected by the length of the path that connecting the categories. The similarity between two concepts is not necessary unchanged all the time. Consider the scenario that lawyers and doctors have common habits in a certain period of time. However, these common habits may change in the next time period. So, with respect to habit, the similarity between lawyer and doctor is changing. Concept hierarchies used in generalized data mining applications are usually given by domain experts. However, it is difficult to maintain a concept hierarchy when the number of categories is huge or when the characteristics of the data are changing frequently. Therefore, there is a need for an algorithm to automatically build a concept hierarchy of a set of nominal values. In this paper, our proposed approach is to modify traditional hierarchical clustering algorithms for this purpose. The input to our method is a distance matrix which

can be computed by CACTUS [2] for tabular data, or by Jaccard coefficient for transactional data.

Our contribution in modifying the traditional agglomerative hierarchical clustering algorithm, called traditional clustering algorithm in the rest of this paper, is twofold. (1) The traditional clustering algorithm builds a binary tree. While, in a concept hierarchy, it is very likely that more than two concepts share a common general concept. In order to capture such characteristics of concept hierarchy, our modified agglomerative hierarchical clustering algorithm allows more than two clusters to merge into a cluster. (2) The leaves of the binary tree generated by the traditional clustering algorithm are not necessary at the same level. This may cause an inversion problem. Consider the merging of a deep subtree and a single-node subtree. The longest path between two leaves on the deep subtree is longer than the path from a leaf on the deep subtree and the leaf of the single-node subtree. We solve this inversion problem by keeping all the leaves at the same level.

In addition to the modified algorithm, we devise a novel measure metrics for the built concept hierarchy by deriving a distance matrix from the concept hierarchy. Root mean squared error between the input distance matrix and the derived distance matrix can then be computed. The paper is organized as follows. In section 2, we illustrate the need for a concept hierarchy. In section 3, the measurement for the algorithms is presented and the way to obtain the input distance matrix is discussed. Section 4 discusses the algorithms to build a concept hierarchy from a given distance matrix among the categories. Experiment description and the result are in section 5. The conclusion is in section 6.

2 The Need for a Concept Hierarchy

Concept hierarchies, represented by taxonomies or sets of mapping rules, can be provided by domain experts. Following are examples of mapping rules [4] for "Status" and "Income" attributes:

Status: {freshman, sophomore, junior, senior} → undergraduate
Status: {graduate, undergraduate} → student
Income: {1,000–25,000} → low_income
Income: {25,001–50,000} → mid_income

Multiple level association rule mining uses *"support"* for obtaining frequent itemsets [5,8]. By increasing the level of the concept hierarchy, support for an itemset increases with a possible increase in frequency. Other applications, such as data warehouse, require dimension tables for drill-down and roll-up operations [3]. Automatically constructing a concept hierarchy for a huge number of categories would relieve the burden of a domain expert.

3 Measurements for Concept Hierarchy

There are some metrics for the quality of clustering. For example, how well the intra-cluster distance is minimized, how well the inter-cluster distance is maximized, how high the accuracy is when the algorithm performs on a pre-classified test dataset. But, there are no existing metrics for measuring the quality of a concept hierarchy automatically built by an algorithm. The above mentioned metrics for clustering algorithms are not applicable to concept hierarchies, since the number of clusters is not concerned for a concept hierarchy building algorithm.

Input to the algorithms is a distance matrix, denoted as *provided* distance matrix. Output from the algorithm is a concept hierarchy. Since a correct concept hierarchy is usually not available, we propose an indirect measurement. In order to compare with the provided distance matrix, we convert the output concept hierarchy into a distance matrix, denoted as *derived* distance matrix. An element of the derived distance matrix is the length of path from a category to another. Min-max normalization is applied to the derived distance matrix so that it has the same scale with the provided distance matrix. Root mean squared error between the two distance matrices can be computed as the quality of the output concept hierarchy.

Definition: Concept Hierarchy Quality Index

Given a distance matrix, $M_{provided}$, over a set of categories $\{c_1, c_2, \ldots, c_n\}$, the quality index of a concept hierarchy with respect to $M_{provided}$ is the root mean squared error between $M_{provided}$ and $M_{derived}$, where $M_{derived}(c_i, c_j)$ is the normalized length of path from c_i to c_j in the concept hierarchy. Root mean squared error is defined as

$$\sqrt{\sum_{\forall ci,c_j} (M_{provided}(c_i, c_j) - M_{derived}(c_i, c_j))^2}$$

There are methods for different types of data to obtain distance matrix. For data in relational tables, we adopt the similarity definition from CACTUS [2] with simplification. After obtaining the similarities between pairs of categories, we would like to normalize the similarities into distances in the range of ϵ and 1. Since the distance between two different categories should be greater than zero, we denote ϵ as the expected distance between a category and its most similar category [7].

4 Algorithms

It is intuitive to use traditional agglomerative hierarchical clustering for building a concept hierarchy of categorical objects. We first describe the traditional algorithm and point out two drawbacks of the algorithms. Then, we propose a modified version of hierarchical clustering algorithm.

4.1 Traditional agglomerative hierarchical clustering

Hierarchical clustering treats each object as a singleton cluster, and then successively merges clusters until all objects have been merged into a single remaining cluster. The dendrogram built in this way is a binary tree. Leaf nodes in such a tree are likely at different levels of the tree. In this paper, we study the three strategies for computing the distance between a pair of clusters, namely, single link [9], average link, and complete link [6].

Agglomerative hierarchical clustering merges the pair of clusters with smallest distance into a cluster. The three strategies define the distance between a pair of clusters. The distances, $dist_{single}$, $dist_{average}$, and $dist_{complete}$, between cluster C_1 and C_2 are defined below.

$$dist_{single}(C_1, C_2) = \min_{x \in C_1, y \in C_2} dist(x, y)$$

$$dist_{average}(C_1, C_2) = \operatorname{avg}_{x \in C_1, y \in C_2} dist(x, y)$$

$$dist_{complete}(C_1, C_2) = \max_{x \in C_1, y \in C_2} dist(x, y)$$

With regard to building a concept hierarchy tree, there are two major drawbacks for traditional hierarchical clustering algorithms. First, the degree of a node can be larger than 2. For example, in figure 1, there are more than two kinds of juices, and they are all specific concepts of juice. However, traditional hierarchical clustering algorithm tends to build a binary tree concept hierarchy.

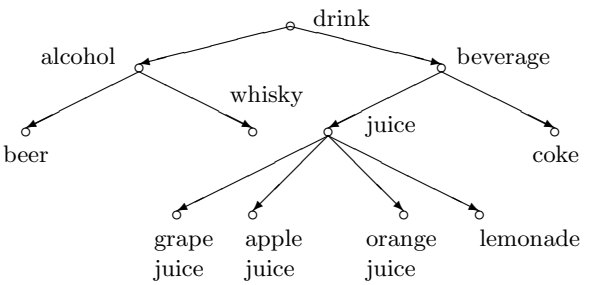


Fig. 1. A Concept Hierarchy

A possible way for perceiving the similarity between two categories in a concept hierarchy is the length of the path connecting the two categories. So, the second drawback is that the distance relationship among the categories might not be preserved with the traditional algorithm. In figure 2, the path from grape juice to orange juice is longer than the path from grape juice to coke. This is in contradiction with the intention specified by the users in figure 1. In order to solve or to improve the drawbacks, we propose modified

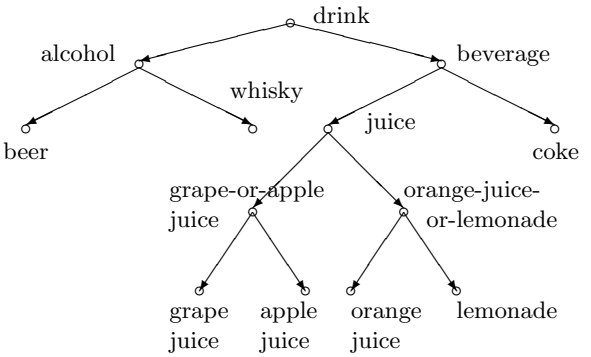


Fig. 2. Concept Hierarchy Built by Traditional Algorithm

hierarchical clustering algorithms that have two important features: (1) leaves of the tree are at the same levels, (2) the degree of an internal node can be larger than 2, i.e., a node *joins* another node in the upper level.

4.2 Multiple-way agglomerative hierarchical clustering

We propose a new hierarchical clustering algorithm to improve the traditional hierarchical clustering algorithm. Initially all items are singleton clusters, and all items are leaf nodes. In order to guarantee that all leaves are at the same level, the algorithm merges or joins clusters level by level. In other words, clusters of the upper level will not be merged until every cluster of the current level has a parent cluster. Two clusters of the same level can be merged and a new cluster is created as the parent of the two clusters. The newly created cluster is placed at the upper level of the tree. We propose a new operation that a cluster can *join* a cluster at the upper level, such that, the cluster of the upper level is the parent of the cluster of a current level. The process continues until the root of the tree is created.

Two clusters of the same level can be merged and a new cluster is created as the parent of the two clusters. The newly created cluster is placed at the upper level of the tree. We propose a new operation that a cluster can join a cluster at the upper level. Such that, cluster of the upper level is the parent of the cluster of current level. The process continues until the root of the tree is created.

In the following discussion, a node is a cluster that contains one or more categorical objects. First, we discuss the "join" operator for hierarchical clustering. Consider the four clusters, A, B, C, and D, in figure 3. Assume that *dist*(A, B) is the smallest among all pairs of clusters, and A and B are merged into cluster E. Assume that either *dist*(A, C) or *dist*(B, C) is less than *dist*(C, D). In other words, cluster C is better merged with A or B than merged with D. In traditional hierarchical clustering algorithm, C is either merged with D

or E. Merging with D is not good for C. Merging with E may be good. But, (1) if *dist*(A, B), *dist*(A, C) and dist(B, C) are about the same, the clustering result makes C quite different from A and B; (2) leaf nodes in the subtree rooted at C will not be at the same level of the whole tree.

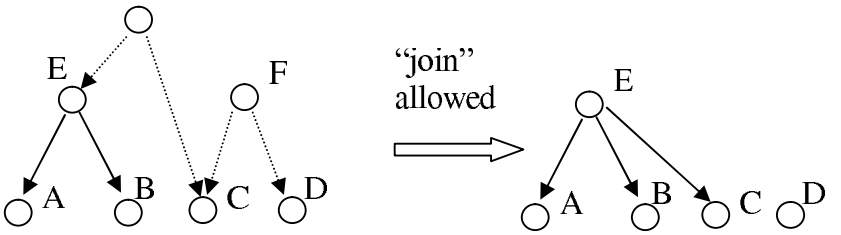


Fig. 3. Hierarchical Clustering with Join Operator

5 Experiment Results

In this paper, we evaluate algorithms with generated data. A provided distance matrix of n objects is generated with the assistance of a tree, which is built bottom up. The data generation is described in the following steps.

1. Let each object be a leaf node.
2. A number of nodes of the same level are grouped together and an internal node is created as the parent node of the nodes. This process continues until the number of internal nodes of a level is one. In other words, the root is created. Any internal node of the tree has at least two children. The degree of an internal node is uniformly distributed in the interval of two and span, a given parameter.
3. The distance between any pair of leaf nodes is prepositional to the length of their path in the tree. The distances are divided by the length of the longest path, i.e., are normalized to one.
4. Noise is applied on the distance matrix. Uniformly distributed numbers between $1 - noise$ and $1 + noise$ are multiplied to the distance values. In the experiment, we generate distance matrices where $noise = 0.1$ and $noise = 0.2$. The distance values, after the noise is applied, are truncated to the interval of zero and one.

The tree generated in the step 2 can be regarded as a perfect concept hierarchy. Since there is noise in the provided distance matrix, so the quality index of the perfect concept hierarchy with respect to the provided distance matrix is not zero.

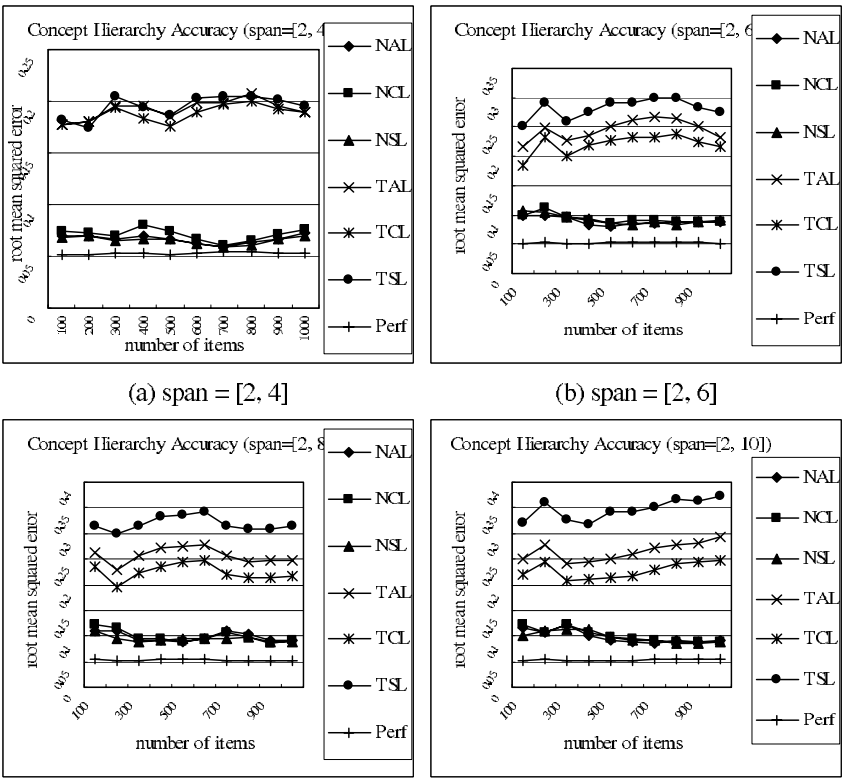


Fig. 4. Experiment Results for noise = 0.1

In the experiment, we illustrate the performance of the algorithms under three parameters, namely *noise*, *span*, and *number of items*. For each parameter combination, root mean squared error values of 30 tests are averaged. For generating provided distance matrices, we build trees with four intervals of spans: [2, 4], [2, 6], [2, 8], and [2, 10].

Figure 4 depicts the quality indices for the algorithms where noise for the input datasets is 0.1. The lines NAL, NCL, and NSL represent for the performance for our new modified algorithm under the strategies average link, complete link, and single link. The lines TAL, TCL, and TSL represent for the performance for traditional algorithm under the three strategies. The line *Perf* represents the quality index of the perfect concept hierarchy.

The results show that our proposed methods perform much better than the traditional agglomerative hierarchical clustering algorithm for all the input distance matrices. However, the strategy of cluster-to-cluster distance does not affect the result in our algorithms. Whereas, for the traditional algorithms, the single link strategy performs better than the other two strate-

gies. The reason might be that we generate the input distance matrices from trees. All the algorithms perform worse for wider spans. Comparing the performance for data with different noise levels, all the algorithms perform worse for noisier data.

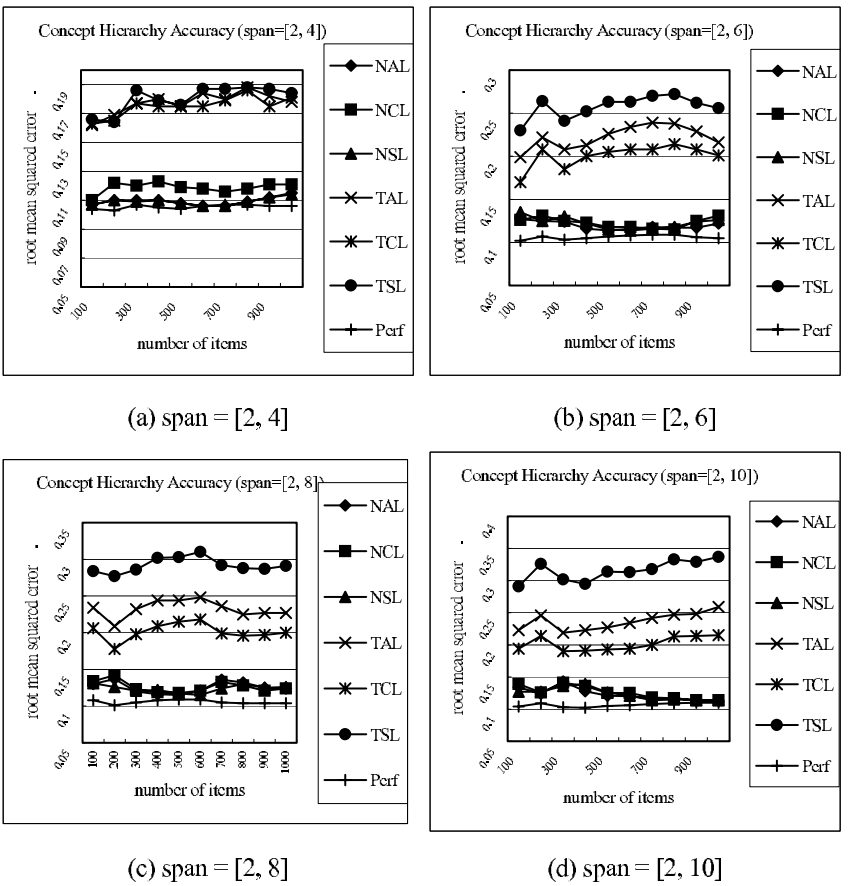


Fig. 5. Experiment Results for noise = 0.2

Figure 5 depicts the quality indices for the algorithms where noise for the input datasets is 0.2. Compare to the result from input data with noise 0.1, root mean squared error increases from 0.07 to 0.12 for our new algorithms where span = [2, 4]. Similar comparisons can be observed for different spans.

6 Conclusions and Future Works

Concept hierarchy is a useful mechanism for representing the generalization relationships among concepts. So that, multiple level association rule mining

can be conducted. In this paper, we build a concept hierarchy from a distance matrix with the goal that the distance between any pair of concepts is preserved as much as possible.

We adopt the traditional agglomerative hierarchical clustering with two major modifications: (1) not only a cluster merges with another cluster, but also a cluster joins another cluster, (2) leaf nodes are all at the same level of the concept hierarchy. Empirical results show that our modified algorithm performs much better than the original algorithm.

Some areas of this study warrant further research: (1) A frequently questioned drawback of hierarchical clustering algorithm is that it does not rollback the merge or division. If re-assignment of an object from a cluster to another is allowed in certain stages, the clustering result may be improved. (2) All the lengths, i.e. weights on edges of the concept hierarchy, are the same. If weights on the edges of the concept hierarchy can be trained, the distance relationship between concepts can be better preserved.

References

1. U. M. Fayyad, K. B. Irani, "Multi-interval Discretization of Continuous-Valued Attributes for Classification Learning," In Proceedings of the thirteenth International Joint Conference on Artificial Intelligence, 1993, pp. 1002-1027.
2. V. Ganti, J. Gehrke, and R. Ramakrishnan, "CACTUS-Clustering Categorical Data Using Summaries," ACM KDD, 1999, pp. 73-83.
3. J. Han and M. Kamber, Data Mining: Concepts and Techniques, Morgan Kaufmann Pub., 2001.
4. J. Han and Y. Fu, "Dynamic Generation and Refinement of Concept Hierarchies for Knowledge Discovery in Databases," Workshop on Knowledge Discovery in Databases, 1994, pp. 157-168.
5. J. Han and Y. Fu, "Discovery of Multiple-Level Association Rules from Large Databases," VLDB Conference, 1995, pp. 420-431.
6. A. K. Jain, R. C. Dubes, Algorithms for Clustering Data, Prentice-Hall, Inc., 1988.
7. Huang-Cheng Kuo, Yi-Sen Lin, Jen-Peng Huang, "Distance Preserving Mapping from Categories to Numbers for Indexing," International Conference on Knowledge-Based Intelligent Information Engineering Systems, Lecture Notes in Artificial Intelligence, Vol. 3214, 2004, pp. 1245-1251.
8. R. Srikant and R. Agrawal, "Mining Generalized Association Rules," VLDB Conference, 1995, pp. 407-419.
9. R. Sibson, "SLINK: An Optimally Efficient Algorithm for the Single-Link Cluster Method," Computer Journal, Vol. 16, No. 1, 1972, pp. 30-34.

Literal Trees and Resolution Technique

Alexander Lyaletski[1], Alexander Letichevsky[2], and Oleksandr Kalinovskyy[1]

[1] Faculty of Cybernetics, Kiev National Taras Shevchenko University,
2, Glushkov avenue, building 6, 03022 Kyiv, Ukraine
[2] Glushkov Institute of Cybernetics, NANU,
40, Glushkov avenue, 03022 Kyiv, Ukraine

Abstract. The problem of mechanized deduction requires carrying out research and comparison of different methods for inference search in first-order classical logic, such as resolution-type methods, the model elimination method, the SLD-resolution, and so on. In this connection, it is desired to give a way for investigating their common and distinct features in order to use them better in theory and practice. This paper is devoted to such an investigation. Interconnection between a complete extension of the SLD-type resolution having the form of literal trees calculus and a ertain resolution technique is established. The interconnection permits to obtain some results on soundness and completeness for different resolution-type methods in the case of the weakest requirements to the factorization. In addition, when classical logic with equality is considered, it gives a possibility to make an original way for complete incorporation of the paramodulation into resolution with weak factorization as well as into the model elimination method.
Key words: automated reasoning, calculus, first-order logic, resolution, tree

1 Introduction

Investigations in automated reasoning technologies gave rise to the appearance of various machine methods for inference search in first-order classical logic. Particularly, the resolution method was suggested [1]. A little later, the model elimination method appeared [2], which can be treated as a certain "linear projection" of trees calculus [3]. In spite of they practically use the same technique for refutation search, they operate with different objects: usual clauses and clauses containing framed literals, respectively. This results in misunderstanding of their common and distinct features.

Investigations presented in the paper are intended to fill a gap in this misunderstanding by means of using the notion of a so-called literal tree instead of a clause with framed literals. Note that literal trees has close relation to SLD-trees being used in Prolog-like systems [4], which are based on the SLD-resolution and play an important role in applications. The SLD-resolution is not complete itself, and considerations presented here require the complete extensions of the SLD-resolution, studied in [5]. In order to make the paper self-contained, all the necessary notions used in [5] are given below, in the section "Preliminaries". In addition, we refer to [6] for modern vision of the model elimination method and of its relation to other computer-oriented methods for inference search.

2 Preliminaries

First-order classical logic with equality is considered. Its language includes the universal and existential quantifiers symbols ($\forall$ and $\exists$), the propositional connectives of implication ($\supset$),disjunction ($\vee$), conjunction ($\wedge$), and negation ($\neg$) as well as the equality symbol ($=$).

It is known that by means of skolemization, the establishment of deducibility of any formula can be reduced to the establishment of deducibility of a formula with eliminated positive quantifiers and with free variables as constants [7]. That is why the examination of formulas with only negative quantifiers restricts us further. In this connection, we can eliminate all the quantifiers from a formula under consideration and can assume that any set of formulas contains quantifier-free formulas only, which pairwise have no common variable. Since the establishment of the deducibility (the validity) of a formula is equivalent to the establishment of unsatisfiability of the negation of this formula, we are interesting in the establishment of the unsatisfiability of a formula (of a set of formulas). Moreover, any formula (without quantifiers) can be transformed to the conjunctive normal form, which allows saying only about the examination of a set of formulas having the form of the disjunction of literals.

The notions of term, atomic formula, formula, and literal are considered to be known. If L is a literal, then $\tilde{L}$ denotes its complement.

A formula being the result of renaming of variables in some formula is called its *variant*.

A formula of the form $L_1 \vee \ldots \vee L_n$ is called a *clause*, where $L_1, \ldots, L_n$ are literals. (That is a clause is an ordered multi-set in terminology of [8].)

The empty formula is denoted by $\sharp$ and called the *empty clause*.

Remark 1. Presentation of a clause as a formula permits to distinguish different occurrences of the same literal in a clause. This property plays an important role later on. Also note, for a purpose of this paper, it is very important that any clause presents itself a "linear" expression when reading it "from left to right", and it determines a linear order over its literals in accordance with the way of its "reading".

In what follows, IS always denotes a set of clauses, which is investigated on unsatisfiability. Any clause belonging to IS is called *input*.

Let IS be a set of ordered clauses, which is examined on the unsatisfiability. Let C be $L_1 \vee \ldots \vee L_n$ and $C \in S$, where $L_1, \ldots, L_n$ are literals. The ordered pair (i, C) $(1 \leq i \leq n)$ is called an *index of occurrence* of L_i in C (w.r.t. IS).

We consider a reader to be familiar with the notions of substitution, unifier and most general unifier (mgu), which are treated like in [1]. Moreover, if E denotes an expression, and σ is a substitution, then the result of application σ to E is understood in the sense of [1] and is denoted by $E * \sigma$. For any set Ex of expressions, $Ex * \sigma$ denotes the result of application σ to every expression from Ex.

Resolution-type methods considered here are based on two rules: a certain modification of the resolution rule and a weak factorization rule introducing below for replacing the usual factorization.

Remark 2. The following condition is satisfied later: Indexes are assigned only to clauses from an initial set IS under consideration. Any application of an inference rule only preserves indexes given to literals from its premise(s). Because all possible inferences in any calculus use IS as a set of "axioms", any deduced clause consists only of indexed literals.

As usual, an *inference* in any calculus under consideration is defined as a sequence of objects, every of which is either a variant of an "input" (given) object or a variant of an object inferred from previous objects in accordance with some rule of the calculus.

Resolution rule (having two forms depending on whether at least one of its premises is an input clause or not.) Let clauses C_1 and C_2 have the form $C_1' \vee L$ and $C_2' \vee E$ (or, $C_2' \vee E \vee C_2''$ in the case of input clause), where L and E are literals; C_1', C_1', and C_2'' are (possibly, empty) clauses. If there exists the mgu σ of the set $\{E, \tilde{L}\}$, then the clause $(C_1' \vee C_2') * \sigma$ $((C_1' \vee C_2' \vee C_2'') * \sigma)$ is said to be inferred by the *resolution rule RR* from C_1 and C_2.

Weak Factorization rule. Let $C_1 \vee L_1 \vee \ldots \vee C_n \vee L_n \vee C_{n+1}$ be a clause, where $C_1 \ldots C_{n+1}$ are clauses, and $L_1, \ldots, L_n$ are literals that have the same index. If there exists the mgu σ of the set $\{L_1, \ldots, L_n\}$, then the clause $(C_1 \vee L_1 \vee C_2 \ldots \vee C_n \vee C_{n+1}) * \sigma$ is said to be inferred by the *weak factorization (WF) rule* from $C_1 \vee L_1 \vee \ldots \vee C_n \vee L_n \vee C_{n+1}$.

Remark 3. If the requirement about the same index is omitted in the definition of the WF rule, we get the definition of the usual factorization.

3 Calculus of Literal Trees for Logic without Equality

Well-formed expressions of the calculus of literal trees denoted by LC are trees with nodes labeled by literals. We identify a node with its label and suppose that any tree grows "from top to bottom".

A "degenerative" tree, not containing any nodes, is denoted by $\triangle$.

As in the case of formulas, some renaming of variables in a literal tree is called its *variant*.

Every input clause C (from a set IS under consideration) having the form $L_1 \vee \ldots \vee L_n$, where $L_1, \ldots, L_n$ are literals, induces the *initial tree* Tr^i w.r.t. C and IS in accordance with the following rule: Tr^i consists only of the root, labeled by IS, and of n its parentheses, labeled by $L_1, \ldots, L_n$ when looking through the parentheses from left to right.

Let $Tr_1, \ldots, Tr_m$ be a sequence of literal trees, where Tr_1 is an initial tree w.r.t. IS and some C, $C \in IS$, and Tr_{i+1} is a variant of a tree inferred with the help of some inference rule from literal trees preceding to Tr_{i+1}. Then the $Tr_1, \ldots, Tr_m$ is called an *inference* of Tr_m from Tr_1 in the calculus LC (w.r.t. S).

Input Extension (IE) rule. Let IS be a set of input clauses, containing C of the form $E_1 \vee \ldots \vee E_j \ldots \vee E_k$, where $E_1, \ldots, E_k$ are literals ($k \geq 1$). Let Tr be a literal tree not having common variables with C, and L be its the most right leaf differed from $\sharp$. Suppose that there exists the mgu σ of the set $\{E_j, \tilde{L}\}$, and that Tr' is constructed from Tr by adding $k-1$ nodes to L that are labeled by E_1, ..., E_{j-1}, E_{j+1}, ..., E_k, respectively (when k is equal to 1, the only node $\sharp$ is added to the node L). Then the tree $Tr * \sigma$ is said to be inferred by the *input extension (IE)* rule from Tr.

Contrary Closing (CC) rule. Let Tr be a literal tree and Br be the most right branch of Tr containing leaf Lf labeled by a non-empty literal L. Let Br contains such a node labeled by a literal E, that there exists the mgu σ of the set $\{E, \tilde{L}\}$. If Tr' is constructed from Tr by adding the only node $\sharp$ to the node L, then the tree $Tr * \sigma$ is said to be inferred by the *contrary closing (CC)* rule from Tr.

Chain Deleting (CD) rule. Let Tr be a literal tree, and Br be a branch of Tr containing leaf Lf labeled by $\sharp$. Let Ch denote the maximal part of Br, such that Ch "is ended" by $\sharp$, and every node of Ch with a label different from $\sharp$ has only one successor. If Tr' denotes the result of a deletion of Ch from Tr, Tr' is said to be inferred by *chain deleting (CD)* rule from Tr.

In what follows, we assume the *CD rule to be automatically applied* after any application of the IE or CC rule. Hence, we restrict us by consideration of literal trees, being inferred in LC and not contained nodes labeled by $\sharp$.

Proposition 1. *Let IS be a set of input clauses without equality literals, and C be such a clause from IS, that the set $IS \backslash \{C\}$ is satisfiable. The set IS is unsatisfiable if and only if there exists an inference of $\triangle$ from the initial tree w.r.t. IS and any $C \in IS$ in the calculus LC.*

Proof. The proposition may be easily obtained from the soundness and completeness of the calculus $LT^{\sharp}$ from [5] by taking into account that trees of literal sequents from [5] can be transformed into literal trees (and vice versa), and this transformation preserves the notion of inference both in LC and in $LT^{\sharp}$ and does not change the "degenerative" tree having the same definition for LC and $LT^{\sharp}$. Q.E.D.

4 Calculus of Literal Trees for Logic with Equality

Application of logic with equality for solving different tasks implies, as a rule, construction and/or use of the already known methods for equality handling. In this connection, we demonstrate some ways of incorporating paramodulation-type rules into the above-described literal tree calculus.

Below we use the notion of E-satisfibility of a set of formulas (in particular, of clauses) in the form of [9].

We remind that an expression of the form $f(x_1, ..., x_k) = f(x_1, ..., x_k)$, where f is a k-arity functional symbol and $x_1, ..., x_k$ are variables, is called functionally reflexive axiom.

If S is a set of clauses, then $Rf(S)$ denotes the set of functionally reflexive axioms for all the functional symbols from S.

The specific feature of the paramodulation extensions proposed is that the "directions" of the paramodulation rule applications as separate rules should be taken into account by the same way like in [3] and that to construct complete extensions, (in general) only three paramodulation-type rules from the four possible may be used (and this number of rules cannot be decreased).

Let us define four types of the paramodulation rule.

Paramodulation Extension rules P_i and P_o. Let IS be a set of clauses, Tr be a literal tree, and L be a label of the most right leaf of Tr different from $\sharp$. Suppose that IS contains such a clause D that the paramodulation rule [9] is applicable from C to L (from L to C), where L is considered as unit clauses, and C is such a variant of D, that it has no common variables with all the labels in Tr. Let σ be the mgu, and $L_1 \vee \ldots \vee L_m$ be a paramodulant of this paramodulation rule application from C to L (from L to C), where $L_1, ..., L_m$ are literals. Then the result of adding m immediate successors to the selected leaf $L * \sigma$ of $Tr * \sigma$ in such a way that for every i $(1 \leq i \leq m)$ the literal L_i is assigned to the ith successor, when looking through the successors from left to right, is said to be a tree deduced from Tr (w.r.t. D) by the *paramodulation extension rule* P_i (the *paramodulation extension rule* P_o).

Paramodulation Extension rules P_d and P_u. Let Tr be a tree, and w be the most right branch of Tr, whose leaf label is L different from $\sharp$. Suppose that the branch w contains a node labeled by such a literal E, that the paramodulation rule is applicable from E to L (from L to E) when L and E are considered as clauses. Let σ be the mgu, and L' be the paramodulant of this paramodulation application from E to L (from L to E). Then the result of adding the only successor L' to the selected leaf $L * \sigma$ of $Tr * \sigma$ is said to be a tree inferred from Tr by the *paramodulation extension rule* P_d (by the *paramodulation extension rule* P_u).

Let us construct calculus LC_d (LC_u) by adding the above-defined rules P_i, P_o, and P_d (P_i, P_o, and P_u) to the calculus LC.

Proposition 2. *Let IS is a set of clauses with equality literals, and C is such a clause from IS, that the set $IS\backslash\{C\}$ is E-satisfiable. The set IS is E-unsatisfiable if and only if there exists an inference of $\triangle$ from the initial tree w.r.t. C and $IS \cup Rf(IS) \cup \{x = x\}$ (x is a variable) in the calculus LC_d (LC_u).*

Proof. This proposition can be obtained as a corollary of the soundness and completeness of the calcali $LT_d^\sharp$ and $LT_u^\sharp$ from [5]. Q.E.D.

Remark 4. Axioms of functional reflexivity are necessary for completeness of LC_d and LC_u. To demonstrate this, we can use the example given in Section 7 of the paper [5].

5 Resolution-type Methods and Literal Trees Calculi

Below we develop a resolution–paramodulation technique both for usual clauses and for clauses containing so-called framed literals, which are used in the model elimination method in its original treatment. The peculiarity of a proposed approach is contained in that this technique constructs as different kinds of "projections" of a tree-like structure on the linear one.

5.1 Clause Images of Literal Trees

Let us look through all the nodes of a tree Tr "from top to bottom and from left to right", and let $N_1, \ldots, N_k$ be a sequence Q of labels of all its nodes, except the root, that is constructed by the following way: every label from Tr appears in Q only in the case of the first attainability of its node. Then the clause $N_1 \vee \ldots \vee N_k$ is called a *total clause image* of Tr and is denoted by $\tau(Tr)$. If $N_{i_1}, \ldots, N_{i_m}$ is a subsequence of Q, consisting of all the labels of leaves from Tr, then $N_{i_1} \vee \ldots \vee N_{i_m}$ is called *leaf-clause image* of Tr and is denoted by $\lambda(Tr)$. In addition, a labels of Q, the node of which is not leaf, is called framed literals in $\tau(Tr)$. Note that both $\tau(\triangle)$ and $\lambda(\triangle)$ are $\sharp$.

Remark 5. Clauses with framed literals, objects of the model elimination methods [2], can be viewed as a certain linear presentation of literal trees. The most complete similarity with these objects is achieved when $\tau(Tr)$ serves as a linear presentation of Tr. This approach to investigation of the model elimination method may give interesting results connected with the incorporation of the paramodulation in the model elimination method.

5.2 Resolution with Weak Factorization and Paramodulation

Let IS be a set of input clauses. We consider the following clauses calculi: RW, RW_l, and RW_r.

Their well-formed formulas are clauses.

The *calculus* RW contains: the rules RR and WF (RW is intended for refutation search in logic without equality).

The *calculus* RW_r *(*RW_l*)* (intended for refutation search in logic with equality), besides of RR and WF, contains restrictions ${P_i}^{\lambda}$, ${P_o}^{\lambda}$, and ${P_d}^{\lambda}$ (${P_u}^{\lambda}$) of the paramodulation rule, being defied as it follows:

– ${P_i}^{\lambda}$ permits to make the paramodulation from an input clause into the most right literal of any inferred clause;

– ${P_o}^{\lambda}$ permits to make the paramodulation from the most right literal being an equality literal of any inferred clause into any input clause;

– ${P_d}^{\lambda}$ (${P_u}^{\lambda}$) permits to make the paramodulation from the most right literal being an equality literal of any inferred clause into the most right literal of another inferred clause if the first (the second) clause was deduced earlier than the second (the first) clause.

Proposition 3. *Let IS be a set of input clauses (with or without equality literals), $C \in IS$, and Tr_1 be initial tree w.r.t. IS and C. Let $Tr_1, \ldots, Tr_m$ be an inference Tr_m in the calculus LC, LC_d, or LC_u. Then the sequence $\lambda(Tr_1), \ldots, \lambda(Tr_m)$ can be transformed into a linear inference [9] of the clause $\lambda(Tr_m)$ in the calculus RW, RW_r, or RW_l, respectively.*

Proof. Suppose, a tree Tr_{j+1} is inferred from Tr_j by means of the rule CC applied to a leaf L and a node E in Tr_j (and the subsequent application of CD). Let Tr_k denote the most right tree that contains E as its leaf ($k < j$). If the clause $\lambda(Tr_k)$ has the form $C_1 \vee E$, and the $\lambda(Tr_j)$ has the form $C_1 \vee C_2 \vee L$, then $\lambda(Tr_{j+1})$ has the form $(C_1 \vee C_2) * \sigma$, where C_1 and C_2 are clauses, and σ is the mgu of $\{L, \tilde{E}\}$. Therefore, RR is applied to $\lambda(Tr_k)$ and $\lambda(Tr_j)$, generating the clause $(C_1 \vee C_1 \vee C_2) * \sigma$. After applying the rule WF to $(C_1 \vee C_1 \vee C_2) * \sigma$, we get the clause $\lambda(Tr_{j+1})$.

Now, let a tree Tr_{j+1} be inferred from Tr_j by means of the rule P_d (P_u) applied to a leaf L and a node E in Tr_j (and the subsequent application of CD). Using the denotations from the above-given paragraph and making similar arguments, we get that the clause $\lambda(Tr_{j+1})$ is inferred by ${P_d}^{\lambda}$ (${P_u}^{\lambda}$) from some $\lambda(Tr_k)$ and $\lambda(Tr_j)$ ($k < j$).

In the case, if Tr_{j+1} is inferred from Tr_j by means of the rule IE, ${P_i}^{\lambda}$, or ${P_o}^{\lambda}$, it is obvious that $\lambda(Tr_{j+1})$ is the consequence of the input resolution or input paramodulation rule [9] applied to $\lambda(Tr_j)$ and some $A \in IS$.

Taking the above into account, it is easy to show by induction on m that after choosing an appropriate calculus from RW, RW_r, and RW_l, we can construct an inference $A_1, \ldots, A_p, B_1, \ldots, B_q$ in it, where the clauses $A_1, \ldots, A_p$ are variants of clauses from IS, and every clause B_i is either $\lambda(Tr_{j+1})$ or a consequence of the rule WF. Q.E.D.

Prop. 1, Prop. 2, and Prop. 3 give the following corollaries.

Corollary 1. *Let IS be a set of input clauses without equality literals, and C is such a clause from IS, that the set $IS \backslash \{C\}$ is satisfiable. The set IS be unsatisfiable if and only there exists a linear inference of $\sharp$ from IS in the calculus RW, and the inference contains C as its first clause.*

Corollary 2. *A set IS of input clauses without equality literals is unsatisfiable if and only if there exists an inference of $\sharp$ from IS in RW.*

Corollary 3. *Let IS be a set of input clauses with equality literals, and C be such a clause from IS, that the set $IS \backslash \{C\}$ is E-satisfiable. The set IS is E-unsatisfiable if and only if there exists a linear inference of $\sharp$ from $IS \cup Rf(IS) \cup \{x = x\}$, x is a variable, in the calculus RW_r (RW_l), and the inference contains C as its first clause.*

Corollary 4. *A set IS of input clauses with equality literals is E-unsatisfiable if and only if there exists an inference of $\sharp$ from $IS \cup Rf(IS) \cup \{x = x\}$, x is a variable, in RW_r (RW_l).*

Remark 6. Remind that a specific peculiarity of the above-described calculi is that they contain only "weakened" analog of the usual factorization used, as a rule, in all implementation of resolution-type. Note that this weakness of the factorization results in the appearance of some "strange" inference like the following one (it is impossible to avoid inferring clauses of type 3 (a tautology) for the given example). Note that L is an atomic formula, and ordered pairs of numbers signed to literals serve as their indexes.

1. $L^{(1,1)} \vee L^{(2,1)}$ (input clause)
2. $\neg L^{(1,2)} \vee \neg L^{(2,2)}$ (input clause)
3. $L^{(1,1)} \vee \neg L^{(1,2)}$ (by RR, from (1) and (2))
4. $L^{(1,1)} \vee L^{(1,1)}$ (by RR, from (1) and (3))
5. $L^{(1,1)}$ (by WF, from (4))
6. $\neg L^{(1,2)}$ (by RR, from (2) and (5))
7. $\sharp$ (by RR, from (5) and (6))

Also note that remark 3 holds for the calculi RW_r and RW_l.

5.3 Model Elimination Method and Paramodulation

The technique of inference search proposed for the above-defined calculi of literal trees can be easily "translated" in the language of the model elimination method treated like in [2,9].

The model elimination method deals with clauses, some of which are declared as *framed* literals, and has the following rules: the resolution (which coincides with RR extended by the "merging left" rule – a "linear form" CD^τ of CD – as its proper sub-rule), the reduction (being a "linear analog" CC^τ of the rule CC including CD as its proper sub-rule), and the factor (left).

Remark 7. Remind that all the literals of any clause from IS are determined as unframed literals. Framed literals appear after applications of the resolution and the reduction.

We define ME as such a calculus of clauses (with framed literals) that only includes the rules RR and CC^τ extended by CD^τ.

It is known that the model elimination method is sound and complete in the case of absence of the equality (see, for example, [2,9]). That is why is interesting to know some kinds of a way for building-in of the paramodulation into ME. Below we give a possible decision of this problem by using special forms of the paramodolution.

Let ME_r (ME_l) denote the calculus ME enriched by ${P_i}^\lambda$, ${P_o}^\lambda$, and the following rule ${P_d}^\tau$ (${P_u}^\tau$).

Paramodulation to the most right literal, ${P_d}^\tau$ rule (Paramodulation from the most right literal, ${P_u}^\tau$ rule). Let C be a clause of the form $L_1 \vee \ldots \vee t{=}t' \vee \ldots \vee L_k$ ($L_1 \vee \ldots \vee L_k \vee \ldots \vee L_r \vee t{=}t'$), where L_1, …, $t{=}t'$, …, L_k (L_1, …, L_k, …, L_r, $t{=}t'$) are literals, among which $t{=}t'$ is obligatory framed equality (unframed equality), and L_k is obligatory unframed literal (framed literal). Let there exists the paramodulation from $t{=}t'$ to L_k generating a paramodulant [9] L_{k+1} (L_{r+1}) and a mgu σ. Then

the clause $(L_1 \vee \ldots\ t{=}t' \ \ldots \vee L_k){*}\sigma \vee L_{k+1}$ $((L_1 \vee \ldots \vee L_k \vee \ldots \vee L_r \vee t{=}t'){*}\sigma \vee L_{r+1})$ is considered to be a conclusion of the rule ${P_d}^{\tau}$ $({P_u}^{\tau})$. The literal $L_k{*}\sigma$ is declared as a framed literal, and, at the same time, L_{k+1} is declared as an unframed literal; all the other literals preserve their property to be framed or unframed literals.

Proposition 4. *Let IS be a set of input clauses (with or without equality), $C \in IS$, and Tr_1 be initial tree w.r.t. IS and C. Let Tr_1, ..., Tr_m be an inference Tr_m in the calculus LC, LC_d, or LC_u. Then the sequence $\tau(Tr_1)$, ..., $\tau(Tr_m)$ is an inference of the clause $\lambda(Tr_m)$ in the calculus ME, ME_r, or ME_l, respestively.*

Proof. We can use the proof of Prop. 3 as an appropriate "scheme". Q.E.D.

Due to Prop. 1, Prop. 2, and Prop. 7, we easily obtain the following results.

Corollary 5. *Let IS be a set of input clauses without equality literals, and C be such a clause from IS, that the set $IS\backslash\{C\}$ is satisfiable. The set IS is unsatisfiable if and only there exists a linear inference of $\sharp$ from IS in the calculus ME, and the inference contains C as its first clause.*

Corollary 6. *Let IS be a set of input clauses with equality literals, and C be such a clause from IS, that the set $IS\backslash\{C\}$ is E-satisfiable. The set IS is E-unsatisfiable if and only if there exists a linear inference of $\sharp$ from $IS \cup Rf(IS) \cup \{x = x\}$, x is a variable, in the calculus ME_r (ME_l), and the inference contains C as its first clause.*

Remark 8. Using a modification of the example from Section 7 of [5], below we give inference of $\sharp$ belonging to both ME_r and ME_l. (Note that framed literals are underlined.) Besides of the necessity of using functional reflexivity axioms, it demonstrates some of peculiarities of ME_r and ME_l.

1. $a{=}b$ (input clause)
2. $R_1(h_1(z,z))$ (input clause)
3. $R_2(h_2(u,u))$ (input clause)
4. $g_1(f(a),f(b)){=}h_1(f(a),f(b))$ (input clause)
5. $g_2(f(a),f(b)){=}h_2(f(a),f(b))$ (input clause)
6. $f(v) = f(v)$ (functional reflexivity axiom)
7. $\neg R_1(g_1(x,x)) \vee \neg R_2(g_2(y,y))$ (input clause)

(a and b are constants, x, y, z, u are variables, R_1 and R_2 are predicate symbols, and f, g_1, g_2, h_1 and h_2 are functional symbols.)

8. $\neg R_1(g_1(x,x)) \vee \underline{\neg R_2(g_2(f(v),f(v)))} \vee \neg R_2(g_2(f(v),f(v)))$

(by ${P_i}^{\lambda}$, from (6) to (7))

9. $\neg R_1(g_1(x,x)) \vee \underline{\neg R_2(g_2(f(a),f(a)))} \vee \underline{\neg R_2(g_2(f(a),f(a)))} \vee \neg R_2(g_2(f(a),f(b)))$ (by ${P_i}^{\lambda}$, from (1) to (8))

10. $\neg R_1(g_1(x,x)) \vee \underline{\neg R_2(g_2(f(a),f(a)))} \vee \underline{\neg R_2(g_2(f(a),f(a)))} \vee \underline{\neg R_2(g_2(f(a),f(b)))} \vee \neg R_2(h_2(f(a),f(b)))$ (by ${P_i}^{\lambda}$, from (5) to (9))

11. $\neg R_1(g_1(x,x)) \vee \underline{\neg R_2(g_2(f(a),f(a)))} \vee \underline{\neg R_2(g_2(f(a),f(a)))} \vee \underline{\neg R_2(g_2(f(a),f(b)))} \vee \underline{\neg R_2(h_2(f(a),f(b)))} \vee \neg R_2(h_2(f(a),f(a)))$ (by $P_i{}^{\lambda}$, from (1) to (10))
12. $\neg R_1(g_1(x,x))$ (by RR extended by CD^{τ}, from (15) and (3))
13. $\underline{\neg R_1(g_1(f(v),f(v)))} \vee \neg R_1(g_1(f(v),f(v)))$ (by $P_i{}^{\lambda}$, from (6) to (12))
14. $\underline{\neg R_1(g_1(f(a),f(a)))} \vee \underline{\neg R_1(g_1(f(a),f(a)))} \vee \neg R_1(g_1(f(a),f(b)))$ (by $P_i{}^{\lambda}$, from (1) to (13))
15. $\underline{\neg R_1(g_1(f(a),f(a)))} \vee \underline{\neg R_1(g_1(f(a),f(a)))} \vee \underline{\neg R_1(g_1(f(a),f(b)))} \vee \neg R_1(h_1(f(a),f(b)))$ (by $P_i{}^{\lambda}$, from (5) to (14))
16. $\underline{\neg R_2(g_2(f(a),f(a)))} \vee \underline{\neg R_1(g_1(f(a),f(a)))} \vee \underline{\neg R_1(g_1(f(a),f(b)))} \vee \underline{\neg R_1(h_1(f(a),f(b)))} \vee \neg R_1(h_1(f(a),f(a)))$ (by $P_i{}^{\lambda}$, from (1) to (15))
17. $\sharp$ (by RR extended by CD^{τ}, from (15) and (2))

6 Conclusion

The paper shows that consideration of computer-oriented calculi using tree-like structures as their well-formed expression can produce new strategies of refutation search even for such well-known methods, as methods based on the resolution and paramodulation rules. Besides, they give different ways of building-in of the paramodulation into the model elimination method. This fact can be explained by the observation that tree-like approach gives more freedom for incorporation of various inference search technique in wide-spread methods and presents itself additional interest for automated reasoning.

References

1. Robinson, J. A. (1965) A machine-oriented logic based on resolution principle. J. of the ACM, **12**, 23–41
2. Loveland, D. V. (1968) Mechanical theorem proving by model elimination. J. of the ACM. **15**, 236–251
3. Lyaletski A. V., Yurchishin, V. V. (1990) On complete extension of input refutation (in Russian). Kibernetika. No 4, 127–128
4. Lloyd, J. V. (1987) Foundations of logic programming. Springer-Verlag, Berlin.
5. Lyaletski, A. (2004) Computer-oriented calculus of sequent tree. Lecture Notes in Computer Science. Springer-Verlag, Heidelberg. **2942**, 213–230
6. Letz, R., Stenz, G. (2001) Model elimination and connection tableaux procedures. Handbook of Automated Reasoning (Ed. by A.Robinson and A.Voronkov). Elsevier Science Pub. 2017–2116
7. Mints, G. (1967) Herbrand theorem (in Russian). Mathematical Theory of Logical Inference. Nauka, Moscow. 311–350
8. Bachmair, L., Ganzinger, H. (2001) Resolution theorem proving. Handbook of Automated Reasoning (Ed. by A.Robinson and A.Voronkov). Elsevier Science Pub. 19–99
9. Chang, C., Lee, R. (1997) Symbolic logic and mechanical theorem proving. Academic Press, Inc. Orlando, FL, USA.

Rough Classification Used for Learning Scenario Determination in Intelligent Learning System

Ngoc Thanh Nguyen[1] and Janusz Sobecki[2]

[1] Institute of Control and Systems Engineering, Wroclaw University of Technology, Poland, E-mail: thanh@pwr.wroc.pl
[2] Institute of Applied Informatics, Wroclaw University of Technology, Poland, E-mail: sobecki@pwr.wroc.pl

Abstract. Learning scenario determination is one of the key tasks of every Intelligent Learning Systems (ILS). This paper presents a method for learner classification in ILS based on rough classification methods proposed by Pawlak. The goal of rough learner classification is based on the selection of such a minimal set of learner profile attributes and their values that can be used for determination of optimal learning scenario. For this aim the problems of rough classification are defined and their solutions are presented.

1 Introduction

The times when knowledge acquired by people in the traditional education institutions were sufficient for the whole life, have gone for ever and there is a great need for application of new environments in this field. Therefore Intelligent Learning Systems are attracting nowadays increasing number of very differentiated users.

The conception of learning scenario determination in ILS using consensus methods have been presented in several recent works [4], [5], [11] and [12]. The different learning scenarios are usually designed because of differences in students' learning styles, which in turn are consequences of differences in cognitive styles. The learning scenario, more precisely described in the following section, is an ordered set of pairs of knowledge piece presentations and corresponding test questions.

The general architecture for learning scenario determination is a modification of that presented in [5]. One of the differences is replacing the classification of learners with the rough classification method. The quality of the classification will be measured with consistency measures. Then rough classification determines the minimal set of the learner profile attributes and their values that lead to the same partition of the set of users as that created by the interface profiles. The main advantage of this approach is the possibility for reduction of the number of attributes that are necessary to generate actual user classification what causes reduction of learner's effort in filling-in registration forms or determining their values by other means.

The paper is organized as follows: in the second section the overall ILS architecture containing learner's profile and learning scenario will be presented, in the Section 3 the rough classification of learners according their profiles will be described. Finally, Section 4 contains some conclusions describing current state of the implementation and the future work perspectives.

2 Architecture of ILS Using Rough Classification

Today ITS is build from three basic modules [8]: the domain, the student and the teacher modules. The overall ILS system architecture resembles that presented in works [4], [5] and [12]. Here the informal and formal description of learner profile, learning scenario as well as learning process in the ILS will be presented.

2.1 Learners profile

Learner profile (LP) contains all necessary data to model the learner in the ILS. In the field of adaptive systems the user profile contains two types of data: user data and usage data [3]. By the analogy the learner profile also may contain the user data that describes the learner such as name, age, sex, education, skills, interests, plans, etc; as well as usage data such as the learner's learning scenario with tests results.

In the recommender systems the user profile could be represented by many different forms: binary vectors, feature vectors, trees, decision trees, semantic networks. In our paper we propose to describe formally the learner profile in the following way. Let us assume that we have a finite set A of attributes describing each user and a set V^u of attribute *elementary values*, where $V = \bigcup_{a \in A} V_a$ (V_a is the domain of attribute a). A user profile is represented by a tuple that is a function

$$p: A \to V$$

where $(\forall a \in A)(p(a) \in V_a)$.

Following the works [4] and [5] we can distinguish two main attribute groups of the learner profile, personal data and individual learning style [1]:

Personal data:

- learner identifier — PI,
- name and forename — NF,
- date of birth — BD,
- age — $AG = \{YO, MI, AD\}$,
- sex — $SX = \{MA, FE\}$
- intelligence quotient — $IQ = \{LQ, MQ, HQ\}$,
- education — $ED = \{EE, SE, HE, PE\}$,

Individual learning style:

- perception — PER = {sensitive_low, sensitive_high, intuitive_low, intuitive_high}
- receiving — REC = {visual_low, visual_high, verbal_low, verbal_high},
- processing — PRO = {active_low, active_high, reflective_low, reflective_high}
- understanding — UND = {sequential_low, sequential_high, global_low, global_high}

2.2 Learning scenario

A learning scenario [4] (LS) represents the proceeding of the learning process of each individual student. The knowledge to acquire in the scope of the course may be divided into some concepts. Each concept may be presented in several ways, and each method of presentation may contain several hypermedia pages. It is assumed that they correspond to the teaching methods that could be applied to present the particular units of knowledge. These differences may result in language of presented definitions or media used for knowledge presentation. The main goal of application of different presentation methods is delivering learners a given knowledge according to his or her individual learning styles. For simplicity reasons, we assume that the order of learning concepts is given and could not be changed. In the ILS systems the individualization of learning process relies on distinguishing a learning scenario for a student according to learner's characteristics, knowledge structure and past learning experiences.

In works [4] and [5] the learning scenario was defined as the ordered set of pairs containing knowledge piece presentation and corresponding test. Where K was the finite set of knowledge units with a partial order, which represents the obligatory order in learning, α in K is defined, R_k denoted the set of all presentations of a knowledge unit $k \in K$, where each sequence of hypermedia pages corresponded to a teaching method suitable to the knowledge unit presentation. The T_k denoted the set of tests serving to verify learner competence referring to k. Let $R = \bigcup_{k \in K} R_k$ and $T = \bigcup_{k \in K} T_k$.

In the scope of this work we simplified the learning scenario of specified course as the tuple $l\colon B \to V'$. Let us assume that we have a finite set B of attributes describing each of the knowledge piece $k \in K$ and corresponding test result. The names of the attributes reflects the order of the knowledge piece in the learning scenario such as *piece_1*, *piece_2*, ..., *piece_n* (where $n = |K|$) with values from R_k ($k = 1, \ldots, n$) as well as corresponding test results *result_1*, *result_2,* ..., *result_n* with real numbers values from $[0, 1]$, which represent a factor of correct answers to all the questions in the test.

2.3 System Architecture

The general architecture for learning scenario determination is following [5]. First, the student that is taking up a course controlled by the ILS initializes the learner profile. The values of the personal data are initialized manually by the learner and the values of the individual learning style are also given by the learner or distinguished by the system according to the probability distribution. Second, the student is classified and third, corresponding learning scenario is determined according to the classification. Initially both learner classification and corresponding learning scenario is determined by the expert or experts. Fourth, the student learns the presented knowledge item according to the and then fifth the student answers the test question. If the student passes the test he can continue with the following knowledge piece from the learning scenario, if not he or she is offered the following presentation type of this knowledge piece and again answers the test. The operation is repeated until the student passes the test or the maximum number of repetitions is reached. In case the student accomplished the test in the second or later trial in the seventh step the learners profile is changed. This modification is based on the assumption that wrong recommendation of the type of presentation from the learning scenario may be caused by wrong values of the individual learning style from the learner profile. The modification should take into account which type of the presentation from the learning scenario finally enabled the learner to pass the test. Then, in the eight step, the classification is modified. Then the whole the algorithm is repeated from the step three.

In the modified architecture applied in the scope of this paper the following steps are identical, in this work however, we applied the rough classification to modify the classification of learners from the step eight. The rough classification is described in the following section.

3 Learner Rough Classification

The concept of rough classification was defined by Pawlak in work [10]. First we must define an information system that is an ordered quadruple $S = (L, A, V, \rho)$, where L is a set of objects; A is a set of attributes which describe the objects; V is the set of attribute values and $\rho: L \times A \rightarrow V$ is the information function, such that $\rho(x, a) \in V_a$ for each $x \in L$ and $a \in A$ [9]. The object is characterized by a tuple in the information system. Objects can be now classified into classes such that objects belonging to the same class have the identical tuples in the information system.

Pawlak worked out a methodology which enables to determine the minimal set of attributes (that is the *reduct*) which generates the same classification as the whole set of attributes. Having a classification C of set L Pawlak proposes a rough classification as its approximation. Assume that classification C is generated by set Q of attributes. The approximation of C is relied

on determining a set $P \subset Q$ such that the classification defined by P differs "a little" from C. The small difference between these classifications is illustrated by the difference of their accuracy measure, which should not be larger than a threshold ε, that is $\mu_Q(X) - \mu_{Q \setminus P}(X) \leq \varepsilon$, for each $X \subset U$, where $\mu_Q(X) = \text{card}(\underline{A}X)/\text{card}(\bar{A}X)$, where $\underline{A}X$ and $\bar{A}X$ are the upper approximation and lower approximation of set X, respectively, in information system S reduced to set R of attributes.

In this paper we consider another aspect of rough classification. This aspect is relied on the assumption that there is known a classification of set of learners L, and one should determine the set B of attributes from A such that the distance between the classification generated by attributes from B and the given classification is minimal. The given classification is that one which arises on the basis of learning scenario and the attributes come from learner profiles. If these attributes could be determined, they would enable to classify the new learners properly and as the consequence, to assign appropriate learning scenarios to them.

The classification is based on the learning scenario and performed by means of k-means algorithm [2]. We must only specify the number k of groups and the distance among the learning scenarios. The most important in the learning scenario is of course the type of presentation so the distance among learning scenarios may be calculate by means of the number of differences of presentation types. We may also consider other distance functions that distinguishes the types of presentations or takes also test results into account.

3.1 Basic Notions

In this section we define the following notions:

A. *Partitions:* By a partition of set L we call a set of nonempty and disjoint classes which are subsets of L, and whose union is equal L. By $\pi(L)$ we denote the set of all partitions of set U. Let P, $Q \in \pi(L)$, the product of P and Q (written as $P \cap Q$) is defined as: $P \cap Q = \{p \cap q : p \in P, q \in Q \text{ and } p \cap q \neq \emptyset\}$. Thus product $P \cap Q$ is also a partition of U.

B. *Distances between partitions*: We define two distance functions (σ and δ) between partitions:

Function σ:

In this function the distance between partitions P and Q of set L is equal to the minimal number of elementary transformations needed to transform partition P into partition Q. These elementary transformations are defined as follows:

(1) Removal of an element — transforms a partition P of set L into a partition Q of set $L \setminus \{x\}$ by deleting an element x from P.

(2) Augmentation of an element — transforms a partition P of set $L \cup \{x\}$ into a partition Q of set L by including an element x in one of existing classes from P or by creating a new class with only one element x.

Function δ:
For $P, Q \in \pi(L)$ let $M(P) = [p_{ij}]_{n\times n}$ be such a matrix that

$$p_{ij} = \begin{cases} 0, & \text{if } x_i, x_j \text{ are in different classes of } P, \\ 1, & \text{if } x_i, x_j \text{ are in the same class of } P \end{cases}$$

and matrix $M(Q) = [q_{ij}]_{n\times n}$ be defined in the similar way, where $n = \text{card}(U)$. The distance between partitions P and Q is defined as follows:

$$\delta(P,Q) = \frac{1}{2}\sum_{i,j=1}^{n} |p_{ij} - q_{ij}|.$$

Such defined distance function is also a metric. It is also easy to show that the algorithm for determining distance of this kind requires $O(n^2)$ time.

Let $P, Q \in \pi(L)$. We say partition P is included in Q (written as $P \subseteq Q$) if each class of P is included in a class of Q. As stated earlier, each user from L is represented by a user profile p which assigns each attribute from A with some value. Then each attribute a from set A determines a partition P_a of set U, as follows: two elements from U belong to the same class in P_a if and only if their user profiles assign attribute a with the same value. More general, a set B of attributes determines P_B of set L, as follows: 2 elements from L belong to the same class in P_B if and only if for each attribute $a \in B$ their user profiles assign with the same value. It is not hard to show that $P_B = \bigcap_{b\in B} P_b$.

3.2 Algorithms for Rough Classification of Learners

As we have assumed above, after registering each new learner is classified by the system into an appropriate class of learners, which is the basis for determining an learning scenario for him or her. As a result of the learning process we receive the actual learning scenario (as defined above) of that particular learner, which is stored for further recommendations.

In this concrete application let's denote by L as the set of lerners of the system and A as the set of all potential attributes which may be used in lerner profiles. By a learner strategy (LS) classification we mean that one which is generated on the basis of user interfaces, and by a user profile (LP) classification we call that one which is generated on the basis of profile attributes. We may notice that in the system, at present, uses LP classifications to assign to a new user a preliminary user interface. During using the system this interface may be personalized by the user (adjusted to his or her needs). As the consequence, we can the final interface should be the better for given user than the initial one.

Let P_B be the actual LP classification of L determined by means of attributes from set $B \subseteq A$ and let Q be the actual LS classification of L. If P_B differs from Q (or their distance is greater than a given threshold) then we

say that the system is weakly adaptive. The smaller is the difference between P_B and Q the more adaptive is the system. A system is then full adaptive if $P_B = Q$.

To make the adaptation more efficient, one should solve the problem of determining P_B on the basic of Q. Notice that P_B is dependent on B, this means that a set B of attributes determines exactly one partition P_B. Besides we know that different sets of attributes may determine the same partition, thus problem is relied on selection of a minimal set B (that is a set with minimal number of elements) such that the distance between P_B and Q is minimal.

Let $d \in \{\sigma, \delta\}$. The problems of rough classification (RC) can formally be defined as follows:

Problem RC-1: *For given partition Q of set L one should determine set $B \subseteq A$ such that $P_B \subseteq Q$ and $d(P_B, Q)$ is minimal.*

In this case set B represents such classification which is equal or minimally more exact than Q. This classification guarantees that two users for whom different interfaces are designed, should not belong to the same class in LP classification. In this case we have the "exact" classification.

Problem RC-2: *For given partition Q of set L one should determine set $B \subseteq A$ such that $Q \subseteq P_B$ and $d(P_B, Q)$ is minimal.*

In this problem set B represents such classification which is equal or minimally differs from Q. In this classification two users for whom different interfaces are designed, may belong to the same class in LP classification, but the number of users of this kind should be minimal. Thus we deal with a kind of rough classification. The only advantage of the solution of this problem is that the number of attributes needed for LP classification is often smaller than in the solutions of problem CR-1. Similarly to CR-1 this problem also may have no solutions.

Problem RC-3: *For given partition Q of set L one should determine set $B \subseteq A$ such that $d(P_B, Q)$ is minimal.*

Solutions of this problem may be needed if solutions of problems RC-1 and RC-2 do not exist. Thus set B represents a rough classification referring to Q. This problem should be solved in the majority of practical cases.

For the above defined problems on the basis of the analysis we propose the following algorithms:

For RC-1: Notice that if $P_B \subseteq Q$ then $P_A \subseteq Q$ because $P_A \subseteq P_B$ (since $P_A = P_B \cap P_{A \setminus B}$). Thus if $P_A \not\subset Q$ then the solution does not exist. If $P_A \subseteq Q$ then set B may be equal A, but for the condition that $d(P_B, Q)$ should be minimal some attribute eliminations from A should be done (see Theorem 1). The Theorem 1 and Theorem 2 proofs may be found in the work [7].

Theorem 1. *For P, $R \in \pi(U)$ and $d \in \{\sigma, \delta\}$ if $P \subseteq Q$ then*

$$d(P, Q) \leq d(P \cap R, Q).$$

According to Theorem 1 if for some attribute a that we have $P_{A\setminus\{a\}} \subseteq Q$ then after eliminating a the distance should be improved, that is

$$d(P_{A\setminus\{a\}}, Q) \leq d(P_A, Q)$$

because $P_A = P_{A\setminus\{a\}} \cap P_a$. The algorithm for RC-1 is following:

BEGIN

1. Calculate P_A;
2. If $P_A \not\subset Q$ then $B := \emptyset$ and GOTO END else $B := A$;
3. For each $a \in B$ do if $P_{B\setminus\{a\}} \subseteq Q$ then $B := B \setminus \{a\}$.

END

For RC-2: Notice that $Q \subseteq P_B$ if and only if $Q \subseteq P_a$ for each $a \in B$. From the Theorem 2 we can see that value $d(P_B, Q)$ is minimal if B contains all attributes $a \in A$ such that $Q \subseteq P_a$. Set B may be minimized by eliminating dependent attributes, that is if there exist attributes a, $b \in B$ such that $P_a \subseteq P_b$ then b should be eliminated.

Theorem 2. *For any information system (U, A), attributes a, $b \in A$ and partition $P \in \pi(U)$ the following properties are true:*

a) If $P \subseteq P_a \subseteq P_b$ then $d(P_a, P) \leq d(P_b, P)$,

b) If $P \subseteq P_a$ and $P \subseteq P_b$ then

$$d(P_{ab}, P) \leq d(P_a, P) \text{ and } d(P_{ab}, P) \leq d(P_b, P),$$

where $d \in \{\sigma, \delta\}$.

The algorithm for RC-2 should consist of the following steps:

BEGIN

1. Let $B := \emptyset$;
2. For each $a \in A$ if $Q \subseteq P_a$ then $B := B \cup \{a\}$;
3. For each a, $b \in A$ and $a \neq b$ do if $P_a \subseteq P_b$ then $B := B \setminus \{b\}$. It can be proved that $d(P_B, Q)$ is minimal for $d \in \{\sigma, \delta\}$.

END.

For RC-3: It was proven that this problem is NP-complete for both distance functions σ and δ [6]. A simple heuristic algorithm for RC-3 is following:

BEGIN

1. Choose $a \in A$ such that $d(P_a, Q) = \min_{b\in A} d(P_b, Q)$;
2. Let $B = \{a\}$;
3. Let $C = B$;
4. If there exists $c \in A\setminus B$ such that $d(P_{B+c}, Q) = \min_{b\in A\setminus B} d(P_{B\cup\{b\}}, Q)$, and $d(P_{B\cup\{c\}}, Q) < d(P_B, Q)$ then $B := B \cup \{c\}$;
5. If $B \neq C$ then GOTO Step 3.

END.

We can notice that the above algorithm is not complex.

4 Conclusions

In this paper a method for analysis of learner profile data in ILS system was. This method is based on user rough classification concept of Pawlak. The difference between our approach and Pawlak's approach to rough classification is that we do not use the upper and lower approximations of partitions, but we use the distance functions between partitions to determining the nearest partition to the given. The application of the rough classification enables as to find the minimal set of the learner profile attributes that the user must deliver to the system during the registration process, or determined in other way, in order to classify them and recommend them appropriate learning scenario. In case when it is impossible to find such set that generates the acceptable partition this gives the system designer the message that the set of the user profile attributes or their values are incomplete and there is a need for give the additional elements.

References

1. Felder, R. M., Silverman, L. K.: Learning and Teching Styles in Engineering Education. *Engineering Education* **78**(7) (1988) 674–681.
2. Kanungo, T. et al.: An Efficient k-means clustering algorithm: analysis and implementation. IEEE Tran. On Pattern Analysis And Machine Intelligence **24** (2002) 881–892.
3. Kobsa, A. et al.: Personalized Hypermedia Presentation Techniques for Improving Online Customer Relationships. The Knowledge Eng. Review **16**(2) (2001) 111–155.
4. Kukla E. et al.: A Model Conception for Optimal Scenario Determination in Intelligent Learning System. Interactive Technology & Smart Education **1** (2004) 3–10.
5. Kukla E. et al.: Determination of Learning Scenarios in Intelligent Web-based Learning Environment. In: Proceedings of IEA-AIE 2004, Ottawa, Canada, Lecture Notes in Artificial Intelligence **3029** (2004) 759–768.
6. Musial, K., Nguyen, N. T., On the nearest product of partitions. Bull. of Polish Academy of Sci. **36**(5-6) (1989) 333–338.
7. Nguyen, N. T., Sobecki, J., Determination of User Interfaces in Adaptive Multimodal Systems Using Rough Classification. Technical Report PRE 268 Department of Information Systems Wroclaw University of Technology. Wroclaw (2004).
8. Nwana, H. S.: Intelligent tutoring systems: an overview. *Artificial Intelligence Review*, **4** (1990) 251–277.
9. Pawlak, Z.: Information systems — Theoretical foundations. Information Systems **6** (1981) 205–218.
10. Pawlak, Z.: Rough classification. Int. J. Human-Computer Studies. **51** (1999) 369–383.
11. Sobecki, J., Morel, K., Bednarczuk, T.: Web-based intelligent tutoring system with strategy selection using consensus methods. In: Intelligent information processing and Web Mining. Proceedings of the International IIS:IIPWM'03

Conference. Mieczysław A. Kłopotek, Sławomir T. Wierzchoń, Krzysztof Trojanowski (eds). Zakopane, June 2–5, 2003. Berlin [i in.]: Springer, cop. (2003) 105–109.
12. Sobecki, J., Nguyen, N. T.: Using consensus methods to user classification in interactive systems. W: Advances in soft Computing. Soft methods in probability, statistics and data analysis. P. Grzegorzewski, O. Hryniewicz, M. A. Gil (eds). Heidelberg; New York: Physica-Verlag (Springer Group) (2002) 346–354.

Rough Ethograms: Study of Intelligent System Behavior

James F. Peters[1], Christopher Henry[1], and Sheela Ramanna[1,2]

[1] Department of Electrical and Computer Engineering, University of Manitoba, Winnipeg, Manitoba R3T 5V6 Canada
[2] Department of Applied Computer Science, University of Winnipeg, Winnipeg, MB R3B 2E9

Abstract. This article introduces a new form of ethogram that provides a basis for studying reinforcement learning in biologically inspired collective robotics systems. In general, an ethogram is a record of behavior patterns, which has grown out of ethology (ways to explain agent behavior). The rough set approach introduced by Zdzisław Pawlak in 1982 provides a ground for deriving pattern-based rewards in the context of an approximation space. The framework provided by an approximation space makes it possible to derive pattern-based reference rewards used to compute action rewards as well as action preferences. A brief description of a prototype of an ecosystem testbed used to record ethograms in a dynamically changing system of agents is presented. The contribution of this article is an introduction to an ethological approach to the study of action preferences and action rewards during reinforcement learning in intelligent systems considered in the context of approximation spaces.

1 Introduction

This paper introduces an approach to observing the collective behavior of cooperating agents in an ethogram, which is part of growing research concerning intelligent systems in the context of rough sets (see, e.g., [3,5,6]). Ethograms are commonly used in field biology (see, e.g., [2]). An *ethogram* is a catalogue of descriptions of separate and distinct, species-typical behavior patterns. The type of ethogram introduced in this article consists of a record of observations made concerning collective as well as individual agent behavior. The term *behavior* in this work refers to the way an agent responds to a stimulus. Action preferences and action rewards during reinforcement learning by collections of cooperating agents (swarms) are recorded in ethograms, which are considered in the context of approximation spaces. Considerable work has been done on approximation spaces (see, e.g., [11]) based on generalized approximation spaces introduced in [12]. This work on approximation spaces is an outgrowth of the original approximation space definition in [4]. Whenever a collection of cooperating agents engages in reinforcement learning, one can observe that actions followed by large rewards are more likely to be repeated. The magnitude of a reward is measured relative to a standard

or reference level. This paper introduces an approach to collecting observations useful in estimating reference rewards. One approach to obtaining a reference reward is to compute an incremental average of recently received rewards relative to actions taken by agents [13]. A pattern based approach to computing reference rewards is introduced in this paper. That is, reference rewards are derived in the context of an approximation space. The contribution of this article is an introduction to an ethological approach to the study of action-preferences and action rewards during reinforcement learning in intelligent systems considered in the context of rough sets.

2 Basic Concepts: Rough Sets

This section briefly presents some fundamental concepts in rough set theory that provide a foundation for projecting rewards for actions by collections of cooperating agents. The rough set approach introduced by Zdzisław Pawlak [4] provides a ground for concluding to what degree a set of equivalent behaviors are a part of a set of behaviors representing a standard. For computational reasons, a syntactic representation of knowledge is provided by rough sets in the form of data tables. Informally, a data table is represented as a collection of rows each labeled with some form of input, and each column is labeled with the name of an attribute (feature) that computes a value using the row input. Traditionally, row labels have been used to identify a sample element commonly called an object (see, e.g., [4]) belonging to some universe (e.g., states, processes, entities like animals, persons, and machines). In this work, the term *sample element* (i.e., member of a population) is used instead of *object* because we want to consider universes with elements that are behaviors observed in real-time. Formally, a data (information) table IS is represented by a pair $(U,\ A)$, where U is a non-empty, finite set of elements and A is a non-empty, finite set of attributes (features), where $a : U \longrightarrow V_a$ for every $a \in A$. For each $B \subseteq A$, there is associated an equivalence relation $Ind_{IS}(B)$ such that $Ind_{IS}(B) = \{(x, x') \in U^2 | \forall a \in B.a(x) = a(x')\}$. Let $U/Ind_{IS}(B)$ denote a partition of U, and let $B(x)$ denote a set of B-indiscernible elements containing x. $B(x)$ is called a block, which is in the partition $U/Ind_{IS}(B)$. For $X \subseteq U$, the sample X can be approximated from information contained in B by constructing a B-lower and B-upper approximation denoted by B_*X and B^*X, respectively, where $B_*X = \{x \in U | B(x) \subseteq X\}$ and $B^*X = \{x \in U | B(x) \cap X \neq \emptyset\}$. The B-lower approximation B_*X is a collection of sample elements that can be classified with full certainty as members of X using the knowledge represented by attributes in B. By contrast, the B-upper approximation B^*X is a collection of sample elements representing both certain and possible uncertain knowledge about X. Whenever B_*X is a proper subset of B^*X, i.e., $B_*X \subset B^*X$, the sample X has been classified imperfectly, and is considered a rough set.

3 Intelligent System Behavior: An Ethological Perspective

A number of features of the behavior of an agent in an intelligent system can be discovered with ethological methods. *Ethology* is the study of behavior and interactions of animals (see, e.g., [14]). The biological study of behavior provides a rich source of features useful in modeling and designing intelligent systems in general. It has been observed that animal behavior has patterns with a biological purpose and that these behavioral patterns have evolved. Similarly, patterns of behavior can be observed in various forms of intelligent systems, which respond to external stimuli and which evolve. In the search for features of intelligent system behavior, one might ask *Why does a system behave the way it does?* Tinbergen's four whys are helpful in the discovery of some of the features in the behavior of intelligent systems, namely, proximate cause (stimulus), response together with the survival value of a response to a stimulus, evolution, and behavior ontogeny (origin and development of a behavior) [14]. Only proximate cause and action taken in response are considered in this paper. Tinbergen's survival value of a behavior correlates with reward that results from an action made in response to a proximate cause. The assumption made here is that action-preference is influenced by a reference or standard reward.

4 Approximation Spaces

The classical definition of an approximation space given by Zdzisław Pawlak in [4] is represented as a pair (U, Ind), where the Indiscernibility relation Ind is defined on a universe of objects U. As a result, any subset X of U has an approximate characterization in an approximation space. A generalized approximation space was introduced by Skowron and Stepaniuk in [11,12]. A *generalized approximation space* is a system $GAS = (\ U,\ I,\ \nu\ )$ where

- U is a non-empty set of objects, and $\mathcal{P}(U)$ is the powerset of U.
- $I : U \rightarrow \mathcal{P}(U)$ is an uncertainty function.
- $\nu : \mathcal{P}(\mathrm{U})\ \mathrm{x}\ \mathcal{P}(\mathrm{U}) \rightarrow [0, 1]$ denotes rough inclusion

The uncertainty function I defines a neighborhood of every sample element x belonging to the universe U (see, e.g., [8]). For example, the sets computed with the uncertainty function $I(x)$ can be used to define a covering of U [10]. The rough inclusion function ν computes the degree of overlap between two subsets of U. Let $\mathcal{P}(U)$ denote the powerset of U. In general, rough inclusion $\nu : \mathcal{P}(\mathrm{U})\ \mathrm{x}\ \mathcal{P}(\mathrm{U}) \rightarrow [0, 1]$ can be defined in terms of the relationship between two sets where

$$\nu(X, Y) = \begin{cases} \frac{|X \cap Y|}{|Y|}, & \text{if Y } \neq \emptyset \\ 1 \quad , & \text{otherwise} \end{cases}$$

for any X, $Y \subseteq U$. In the case where $X \subseteq Y$, then $\nu(X, Y) = 1$. The minimum inclusion value $\nu(X, Y) = 0$ is obtained when $X \cap Y = \emptyset$ (i.e., X and Y have no elements in common). In a hierarchical model of an intelligent system, one or more approximation spaces would be associated with each layer [7].

4.1 Example: Approximation Space for a Swarmbot

To set up an approximation space for a swarmbot, let $DT_{sbot} = (U_{beh}, A, \{d\})$ be a decision system table, where U_{beh} is a non-empty set of behaviors, A is a set of swarmbot behavior features, and d is a distinguished attribute representing a decision. Assume that $I_B : U_{beh} \rightarrow \mathcal{P}(U_{beh})$ is an uncertainty function that computes a subset of U_{beh} relative to parameter B (subset of attributes in A). For example, $I_B(x)$ for $x \in U_{beh}$, $B \subseteq A$ can be used to compute B_*D of $D \subseteq U_{beh}$. Further, let B_*D represent a standard for swarmbot behaviors, and let $B_a(x)$ be a block in the partition of U_{beh} containing x relative to action a (i.e., $B_a(x)$ contains behaviors for a particular action a that are equivalent to x). The block $B_a(x)$ is defined in (1).

$$B_a(x) = \{y \in U_{beh} : xIND\,(B \cup \{a\})\, y\} \tag{1}$$

Then we can measure the closeness of $B_a(x)$ to B_*D as in (2):

$$\nu_B(B_a(x), B_*D) = \frac{|B_a(x) \cap B_*D|}{|B_*D|} \tag{2}$$

B_*D represents certain knowledge about the behaviors in D. For this reason, B_*D provides a useful behavior standard or behavior norm in gaining knowledge about the proximity of behaviors to what is considered normal. The term *normal* applied to a set of behaviors denotes forms of behavior that have been accepted. The introduction of some form of behavior standard makes it possible to measure the closeness of blocks of equivalent action-specific behaviors to those behaviors that are part of a standard.

5 Ethograms Reflecting Swarmbot Behavior

A prototype for a testbed that automates the production of ethograms that provide a record of observed swarmbot (sbot) behavior patterns, is briefly introduced in this section (see Fig. 1). Principal testbed symbols are explained in Fig. 2.

Multiple bots are shown in Fig. 1, where inspect-bots cooperate to navigate along a powerline with many obstacles to inspect power system equipment (e.g., insulators and towers). A sample snapshot of the individual and

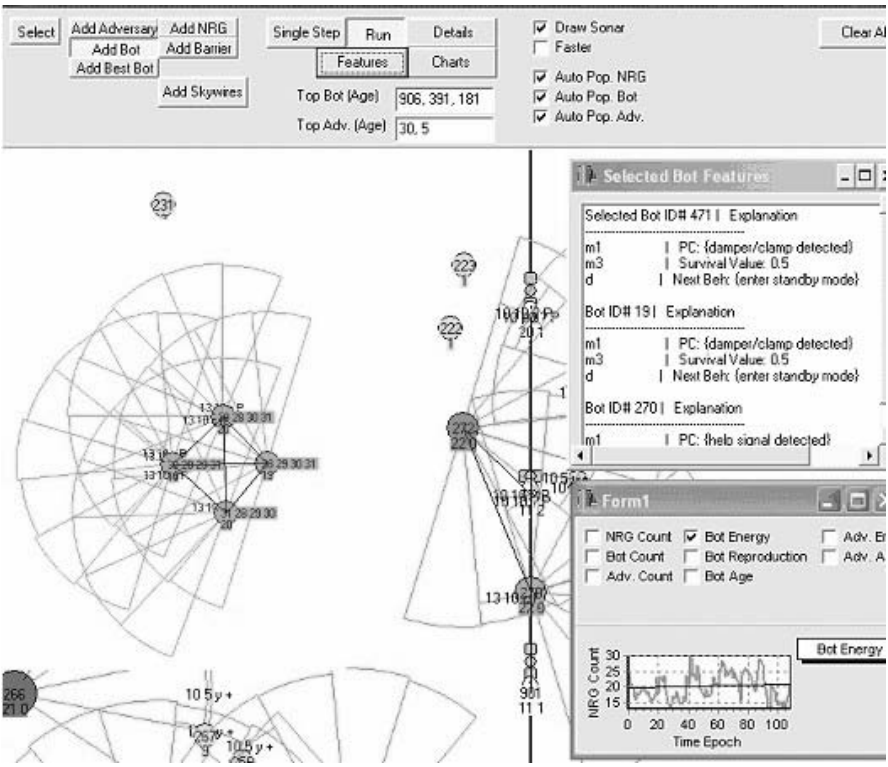


Fig. 1. Testbed for Swarmbot Behavior

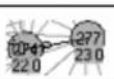
Symbol	Meaning	Symbol	Meaning
	Vibration damper.		Bot with ID 45, energy level 110.
	Tower clamp attached to wire.		Swarm (line shows wireless connection between bots).
	Energy source, ID 263, level 8		Energy radiating from bot proximity sensors.

Fig. 2. Swarmbot Testbed Symbols

collective behavior of inspect bots in a line-crawling swarmbot testbed that is part of the Manitoba Hydro Line-Crawling Robot research project (see Fig. 3).

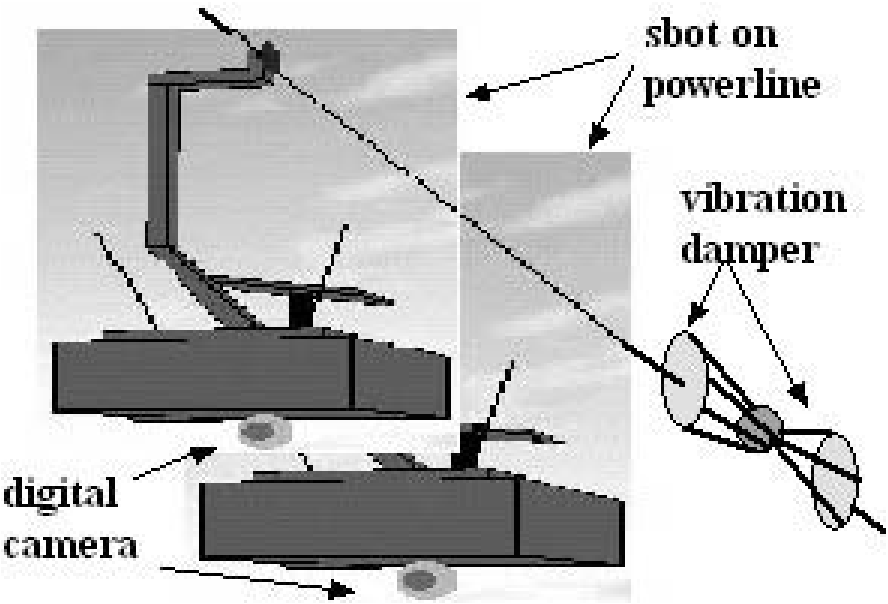


Fig. 3. Cooperating Bots Prototype

Briefly, this testbed makes it possible to experiment with various off- and on-line learning algorithms and various swarm behaviors such as cooperation between bots where one bot requires the help of another bot to crawl past an obstacle, message-passing between bots using wireless communication, and responding to various adversaries such as wind, rain, lightning, hunters and birds. Also included in the sbot testbed is a provision for tracking feature values for individual bots as well as swarms (see *Selected Bot Features* window in Fig. 1) over time (e.g., energy level) useful in behavior survivability estimates. For example, a bot that encounters an obstacle enters standby mode, and attempts to send a call for help to neighboring bots. The survivability of a bot in standby mode decreases over time (the longer it waits, the less chance it has to renew its energy). A bot in standby mode learns to cope with its decreasing energy problem and no response from another bot, by exiting standby mode and searching for a means to renew its energy (this usually means the bot adjusts its angle of rotation so that its solar panels can absorb the sun's rays). Cooperation between bots begins by one bot responding to a wireless communication from another bot. This marks the birth of a swarmbot. A responding bot docks or connects itself to a bot asking for help.

Then the responding bot provides stability and support while the other bot opens its calipers at the tips of its appendages (closed calipers hold onto a line). After this, the responding bot pushes the other bot past an obstacle. Such cooperation between bots is one of the hallmarks of swarmbot behavior (see, e.g., [1]), and in multiagent systems containing independent agents that exhibit some degree of coordination (see, e.g., [7]). Many of the details concerning this sbot have been already been reported (see, e.g., [9]). A more detailed consideration of the sbot testbed is outside the scope of this paper.

5.1 Sample Ethogram for a Swarmbot

An ethogram provides a record of observed swarmbot behavior that includes proximate cause (stimulus) and actions chosen and not chosen. The state of a swarmbot is either high or low (average charge on batteries of bots in a swarm). The form of ethogram automated in the testbed in Fig. 1 includes estimated action preference and reward as well as an action decision d. An overview of the codes for stimuli and actions is given in Fig. 4. For example, when the stimulus for an sbot is low battery (i.e., the average charge on the batteries of the bots in a swarm is below some preset threshold), then the action in response is either "get battery" (action code = 10) or "recharge" (action code = 11).

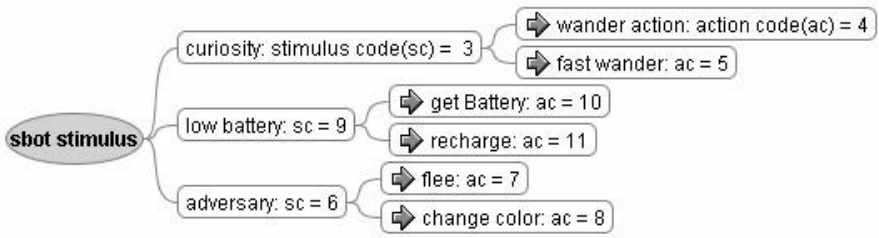


Fig. 4. Codes for Stimuli and Actions by an Sbot

As the energy for a swarm reduces, then survivability of the swarm also reduces. Battery charge (i.e., energy) for a swarmbot is computed using $V_{swarm} = average\,\{V_{bat}/V_{max} for\ each\ member\ of\ the\ swarm\}$ where V_{bat} is the current charge on a bot battery and V_{max} is the maximum voltage on a battery for a member of a swarm. Let th be a threshold on V_{swarm}. If $V_{swarm} < th$, then V_{swarm} is low, which is indicated by state $s = 0$. Otherwise, V_{swarm} is high and $s = 1$. Let a_t, r_t, $\overline{r}_t$, denote action, reward and reference reward at time t, and let p_{t+1} denote an action-preference at time $t + 1$. Preference and reference reward are computed in (3) and (4), respectively.

$$p_{t+1}(a_t) = p_t(a_t) + \beta(r_t - \bar{r}_t) \tag{3}$$

$$\overline{r}_{t+1} = \overline{r}_t + \alpha(r_t - \overline{r}_t) \tag{4}$$

where α, $\beta \in (0, 1]$ denote step size parameters. Formulas (3) and (4) are the same as in [13]. However, the method used to compute the reference reward in this article differs from the averaging method given in [13].

5.2 Pattern-Based Reward

An approach to deriving a pattern-based reward is given in this section. Let $B_a(x)$ denote a block in partition $U_{beh}/Ind_{DT}(B)$ with respect to action a. In the context of swarm behavior recorded in an ethogram, the lower approximation B_*D represents certain knowledge about blocks of equivalent behaviors. In what follows, the feature set B would always include feature a (action). The degree of inclusion of each of the blocks for a particular action yields useful information in reinforcement learning by a swarm. The rough inclusion values of the action blocks give us an indication of the closeness of the behaviors to the standard. We are interested in viewing this information collectively as a means of guiding the distribution of near-future rewards for particular behaviors. One way to do this is by averaging the closeness values for each of the action blocks represented, for example, in Table 1. In general, in the context of an approximation space, a reference reward $\bar{r}_t$ for each action a is computed as shown in (5).

$$\overline{r}_t = \frac{\sum_{i=1}^{n} \nu_B(B_a(x)_i, B_*D)}{n} \tag{5}$$

where n is the number of blocks $B_a(x)$ in $U_{beh}/Ind_{DT}(B)$.

A case study showing sample calculations of preference and reward relative to a particular action (namely, $a_t = 4$ (wander)) is given next based on Table 1, where

$$B_*D\,|_{d=1} = \left\{ \begin{matrix} \text{x1, x4, x6, x7, x9, x13, x15, x17, x20,} \\ \text{x22, x23, x26, x34, x36, x38, x40} \end{matrix} \right\}$$
$$r_t = 0.012, \bar{r}_t = 0.0475, \bar{r}_{t+1} = 0.044, p_{t+1}(a_t) = 0.028$$
$$B_*D\,|_{d=0} = \{\text{x3, x5, x19, x21, x25, x33, x35, x37, x39, x41}\}$$
$$r_t = 0.012, \bar{r}_t = 0.075, \bar{r}_{t+1} = 0.0687, p_{t+1}(a_t) = 0.025$$

The reference reward is smaller (small punishment) when d = 1 compared with the reference reward (higher punishment) when d = 0. This matches the intuition that when the agent is in a desirable state indicated by d = 1, the smaller the reference reward $\overline{r}_t$ at time t, the higher $\overline{r}_{t+1}$ at time t+1. The proximate causes and actions chosen by an sbot have been simplified in Fig. 4. Many other actions are possible, difficult to predict, and need to be learned in real-time (e.g., selecting a camera angle for inspection of a piece

Table 1. Sample Ethogram for Swarm Behavior ($\alpha, \beta = 0.1$).

$X\backslash A$	s_t	pc_t	a_t	p_t	r_t	d_t	$X\backslash A$	s_t	pc_t	a_t	p_t	r_t	d_t
x1	1	3	4	0.02	0.011	1	x22	1	6	8	0.02	0.011	1
x2	1	3	5	0.01	0.01	0	x23	0	9	10	0.02	0.011	1
x3	0	3	4	0.01	0.01	0	x24	0	9	11	0.01	0.01	0
x4	0	3	5	0.01	0.01	1	x25	0	3	4	0.01	0.01	0
x5	0	3	4	0.01	0.01	0	x26	0	3	5	0.01	0.01	1
x6	0	3	5	0.01	0.01	1	x27	1	3	4	0.01	0.01	0
x7	0	9	10	0.01	0.01	1	x28	1	3	5	0.01	0.01	1
x8	0	9	11	0.01	0.01	1	x29	1	3	4	0.01	0.01	0
x9	0	9	10	0.01	0.01	1	x30	1	3	5	0.01	0.01	1
x10	0	9	11	0.01	0.01	0	x31	1	3	4	0.01	0.01	1
x11	1	3	4	0.01	0.01	1	x32	1	3	5	0.01	0.01	0
x12	1	3	5	0.01	0.01	0	x33	1	6	7	0.01	0.01	0
x13	1	3	4	0.02	0.011	1	x34	1	6	8	0.02	0.011	1
x14	1	3	5	0.01	0.01	0	x35	0	6	7	0.01	0.01	0
x15	1	3	4	0.031	0.012	1	x36	0	6	8	0.02	0.011	1
x16	1	3	5	0.01	0.01	0	x37	0	6	7	0.01	0.01	0
x17	0	9	10	0.02	0.011	1	x38	0	6	8	0.02	0.011	1
x18	0	9	11	0.01	0.01	0	x39	0	6	7	0.01	0.01	0
x19	0	6	7	0.01	0.01	0	x40	0	6	8	0.02	0.011	1
x20	0	6	8	0.01	0.01	1	x41	0	6	7	0.01	0.01	0
x21	1	6	7	0.01	0.01	0							

of equipment). To cope with changing requirements and deteriorating performance by an s-bot, a reference reward needs to be periodically recomputed in terms of a new lower approximation for a new set of recent behaviors.

6 Conclusion

The studies of animal behavior by ethologists provide a number of features useful in the study of changing intelligent system behavior in response to environmental (sources of stimuli) as well as internal influences (e.g., image classification results, battery energy level, response to behavior pattern recognition, various forms of learning). This article introduces an approach to tabulating observations about the behavior of a swarmbot that learns to select actions resulting in rewards that are associated with survivability. In effect, a swarmbot learns to select actions so that cooperation between bots in a swarm serves to protect the sbot against extinction. An approach to computing reference rewards in the context of an approximation space is introduced in this article.

7 Acknowledgements

Many thanks are extended to the anonymous reviewers of this article as well as to Andrzej Skowron for their suggestions. Also, many thanks to Maciej Borkowski for creating the OpenGL view of a line-crawling bot, which provides the basis for Fig. 3. This research has been supported by Natural Sciences and Engineering Research Council of Canada (NSERC) grants 185986, 194376 and grant T247 from Manitoba Hydro.

References

1. Bonabeau, E., Dorigo, M., Theraulaz, G. (1999): *Swarm Intelligence. From Natural to Artificial Systems*, Oxford University Press, UK
2. Lehner, P. N. (1996): *Handbook of Ethological Methods*, 2nd Ed. Cambridge University Press, UK
3. Pal, S. K., Polkowski, L., Skowron, A.,Eds. (2004): *Rough-Neural Computing. Techniques for Computing with Words*. Springer-Verlag, Heidelberg
4. Pawlak, Z. (1991): *Rough Sets. Theoretical Reasoning about Data*. Kluwer, Dordrecht
5. Peters, J. F. (2004): Approximation space for intelligent system design patterns. *Engineering Applications of Artificial Intelligence* **17**, No. 4, 1–8
6. Peters, J. F., Ramanna, S. (2004): Measuring acceptance of intelligent system models. In: M. Gh. Negoita et al. (Eds.), Knowledge-Based Intelligent Information and Engineering Systems, *Lecture Notes in Artificial Intelligence* **3213**, Part I, 764–771
7. Peters, J. F. (2004): Approximation spaces for hierarchical intelligent behavioral system models. In: B.D.-Keplicz, A. Jankowski, A. Skowron, M. Szczuka (Eds.), Monitoring, Security and Rescue Techniques in Multiagent Systems. *Advances in Soft Computing*. Physica-Verlag, Heidelberg, 13–30
8. Peters, J. F., Skowron, A., Synak, P., Ramanna, S. (2003): Rough sets and information granulation. In: Bilgic, T., Baets, D., Kaynak, O. (Eds.), Tenth Int. Fuzzy Systems Assoc. World Congress IFSA, Instanbul, Turkey, *Lecture Notes in Artificial Intelligence* **2715**, Springer-Verlag, Heidelberg, 370–377
9. Peters, J. F., Ahn, T. C., Borkowski, M., Degtyaryov, V., Ramanna, S. (2003): Line-crawling robot navigation: A rough neurocomputing approach. In: C. Zhou, D. Maravall, D. Ruan (Eds.), Autonomous Robotic Systems. *Studies in Fuzziness and Soft Computing* **116**. Physica-Verlag, Heidelberg, 141–164
10. Polkowski, L. (2002): *Rough Sets. Mathematical Foundations*. Physica-Verlag, Heidelberg
11. Skowron, A., Stepaniuk, J. (1998): Information granules and approximation spaces. In: Proc. of the 7^{th} Int. Conf. on Information Processing and Management of Uncertainty in Knowledge-based Systems (IPMU98), Paris, 1354–1361
12. Skowron, A. Stepaniuk, J. (1995): Generalized approximation spaces. In: Lin, T.Y., Wildberger, A.M. (Eds.), Soft Computing, Simulation Councils, San Diego, 18–21
13. Sutton, R. S., Barto, A. G. (1998): *Reinforcement Learning: An Introduction*. The MIT Press, Cambridge, MA
14. Tinbergen N. (1963): On aims and methods of ethology, *Zeitschrift für Tierpsychologie* **20**, 410–433

Automatic Knowledge Retrieval from the Web

Marcin Skowron and Kenji Araki

Graduate School of Information Science and Technology, Hokkaido University, Kita-ku Kita 14-jo Nishi 8-chome, 060–0814 Sapporo, Japan

Abstract. This paper presents the method of automatic knowledge retrieval from the web. The aim of the system that implements it, is to automatically create entries to a knowledge database, similar to the ones that are being provided by the volunteer contributors. As only a small fraction of the statements accessible on the web can be treated as valid knowledge concepts we considered the method for their filtering and verification, based on the similarity measurements with the concepts found in the manually created knowledge database. The results demonstrate that the system can retrieve valid knowledge concepts both for topics that are described in the manually created database, as well as the ones that are not covered there.

1 Introduction

Despite the years of research in the field of Artificial Intelligence, the creation of a machine with the ability to think is still far from realization. Although computer systems are capable of performing several complicated tasks that require human beings to extensively use their thinking capabilities, machines still cannot engage into really meaningful conversation or understand what people talk about. One of the main unresolved problems is the lack of machine usable knowledge. Without it, machines cannot reason about the everyday world in a similar way to human beings. In the last decade we have witnessed a few attempts to create knowledge databases using various approaches: manual, machine learning and mass collaboration of volunteer contributors. So far, none of these approaches can be considered as fully reliable and/or efficient. To a various extent, each of the presented methods has some problem areas and limitations that are inherent to it. We discuss this issue in more detail in the next section.

Compared to the efforts taken to create knowledge databases manually, either as an effort of one entity or distributed among several parties, surprisingly little has been done to develop the methods for the automatic retrieval of knowledge concepts for such databases. This paper presents a language independent method, capable to retrieve general and commonsensical knowledge for the open-domain applications from the web and its implementation to the working system. The aim of the system that implements it, is to automatically create entries to a knowledge database, similar to the ones that are being provided by the volunteer contributors. The system uses Internet as a source of knowledge concepts candidates. Obviously, only a small fraction of the statements that are accessible on the web can be considered as

a valid entries to a knowledge database. Below, we describe the methods for "web knowledge concepts" filtering and verification, which are based on the similarity comparison with concepts found in a knowledge database created with the mass collaboration of Internet users. The paper also presents the implementation of these methods to the developed system and preliminary results obtained from it.

2 Approaches to Knowledge Database Construction

The most well known example of the manual approach to knowledge base construction is the CYC project, which contains 1.5 million assertions build over 15 years[4]. The aim of the project was to create a database that could provide knowledge from a large scope of domains, along with the means to effectively use this knowledge for systems that could engage in the reasoning of human affairs. We learn from the experiences of this and similar projects that building a database in this way was laborious, time-consuming and costly. We argue that in an open domain where new information appears and becomes obsolete on a daily basis, a complete knowledge-base build using this approach is out of reach.

The machine learning approach demonstrated that it was feasible to automatically induce rules from data and to overcome some problems characteristic for the manual approach presented above. However, as of yet the machine learning approach has not resulted in creation of a large, open-domain knowledge base. A typical learning program has only weak assumptions about the world; consequently, the learned rules are relatively shallow as they refer only to correlations between observable variables[7]. To address this problem the researchers attempted to incorporate pre-existing knowledge, combining the automatic and manual approaches. However, they soon faced the bottleneck related to collecting knowledge known from the manual approach.

Knowledge acquisition from Internet users is the approach used in the construction of the Open Mind Common Sense database (OMCS)[10], allowing mass collaborations of thousands of volunteer contributors. The ability to input knowledge concepts directly in the natural-language form simplified the process of database building compared to the attempts that used semi-programming languages. Thanks to this, theoretically every (English speaking) Internet user, without any special training can input knowledge concepts using plain sentences. As demonstrated by Borchardt[2] and Singh[10] such concepts can be used for reasoning. While significantly decreasing the amount of money spent, this approach still requires large investment of human labor distributed among thousands of Internet users. Compared to the manual approach, the time requirement for creation of the knowledge database is reduced; but, this factor cannot be overlooked. The challenges frequently faced in the projects that use the mass collaboration approach include the need to ensure[7]:

- High quality of contributed knowledge,
- Consistency between knowledge entered by different contributors and at different times,
- Relevance of inputted knowledge to a given task,
- Scalability of the project,
- Motivation of contributors to start and consistently work on the project.

So far the OMCS project has gathered more than 700,000 items from more than 15,000 users. The database was evaluated[10] by human judges using a sample of the knowledge concepts, and the following conclusions were presented: 75% of the items are largely true, 82% are largely objective, 85% were judged as largely making sense. Tables 1 and 2 show the examples of knowledge concepts related to the nouns "water" and "apple" from the OMCS database.

Table 1. Examples of knowledge concepts related to "water" from the OMCS.

No.	Knowledge concept related to "water"
1	The last thing you do when you take a shower is turn off the water
2	Human beings need water to survive
3	When animals need water, they feel thirsty
4	Clouds are made up of water vapor
5	People need to drink water verey day

Table 2. Examples of knowledge concepts related to "apple" from the OMCS.

No.	Knowledge concept related to "apple"
1	Yellow apples are soft and sweet
2	The first thing you do when you eat an apple is rub it
3	An apple contains seeds
4	The Michigan state flower is the apple blossom
5	When you drop an apple, it gets bruised

3 Automatic Knowledge Retrieval from the Web

Analyzing the content of knowledge database created by volunteer contributors, one can discover that several of the statements found there exist also on freely available web pages. Additionally, the number of "web knowledge concepts", using slightly different words and/or syntax, semantically provides equivalents for a large part of the entries from the manually constructed knowledge-bases. Other knowledge concepts accessible on the web describe

topics that are not covered yet in manually created databases or provide wider coverage and additional details for several of the concepts defined in the manually created database.

We argue that the web is a rich resource of commonsensical and general knowledge and that this resource is usable in the process of automatic creation of knowledge databases. For automatic knowledge retrieval the web has important advantages, including real-time updates of content, wide coverage of various domains, and diversity of presented opinions. At present, a popular search engine indexes more than $8*10^9$ web pages. Assuming only a small portion of them include statements that can be treated as valid entries to a knowledge database, the web still hosts an immense number of knowledge concepts that can be automatically retrieved. We think that in every moment, in various part of the world Internet users/WWW creators contribute the knowledge that can and ought to be used for the building of the knowledge databases and supporting several AI applications. Obviously, there is a need to filter out statements that cannot be considered as reliable and valid entries to a knowledge database. The main challenge in this approach is to ensure a high recall rate of knowledge concepts from various domains and precision of concepts filtering and selection.

4 Relevant Research

Some relevant research adapting different approaches, and to some extent having different aims, is included in the following works. Studies on knowledge representation and classification were done by Woods, followed by work on automatic construction of taxonomies by extracting concepts from texts [11]. Satoh [8] use connective markers to acquire casual knowledge similarly to the later work of Inui [3], where the method for classification of casual relations was presented. The research of Rzepka[9] described the methods for automatic word categorization and script generation using the statistic of words obtained from a Web-sized corpus and by exploring Japanese grammar features. The work of Alani [1] presented a method of automatic knowledge extraction from a selected domain (knowledge about artists) from web documents based on a predefined ontology. In the most closely related work [6], Moldovan described a system that gathers knowledge from a financial domain. However, this system was tailored to a specific domain and its processing could not be done automatically, as the system required an interaction with a user to verify the acquired concepts, relationships and extraction patterns.

5 System Processing

Below we present the "KnowY" system that implements the idea of automatic retrieval of open-domain knowledge concepts from the web. There are two aims of this system: to automatically create a knowledge database similar to

ones that are being built manually through the mass collaboration of Internet users, and to support various AI applications with the knowledge required in their processing. At present, the system utilizes the OMCS database to learn what constitutes a valid entry to a knowledge database. This information is necessary for filtering out statements that are unlikely to form such an entry and to rank discovered concepts depending on their similarity to ones found in the OMCS database. The adaptation to any other knowledge database written in natural language form is also feasible. The system is developed using Perl and C, and implemented on the single Pentium III-class Linux 2.4 workstation. For locating and accessing knowledge concepts on the web "KnowY" uses Yahoo search engine[1]. Figure 1 presents the system flowchart.

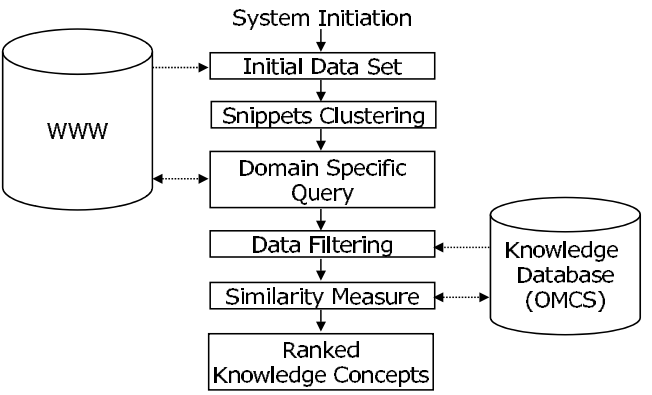


Fig. 1. System flowchart

"KnowY" is initiated by receiving a request for a noun or list of nouns for which knowledge concepts are to be discovered. To provide a wide representation of various topics, the system retrieves the 300 highest scored snippets for each of the 5 different domains (.gov, .edu, .net, .org, and .com) using a submitted noun as a query. A set of the unique snippets is clustered[5], and for each cluster the 3 most descriptive words are discovered. Initially the number of clusters is set to 4. For example for the noun "apple" the following most descriptive words characterizing the content of the clusters have been found (tree, fruit, apples), (juice, production, cider), (computer, mac, macintosh), (iTunes, music, server).

In the next step, "KnowY" executes the new search using a query consisting of the requested noun along with the 3 most descriptive words discovered for each of the discovered clusters. The system retrieves a set of 300 web pages, which provides a wide selection of information concerning the noun.

[1] http://www.yahoo.com

Extension of a used query with the 3 most descriptive words for a given cluster ensures that domain limitations are respected. After dividing the text into sentences, the ones that do not include the requested noun (either in singular or plural form) are excluded.

In the data filtering and similarity measure stages, the OMCS database was used as training data that revealed the most frequent ways of describing knowledge concepts, as well as commonly used words and grammatical constructions[2]. The comparison set is composed of the concepts that include the requested noun (either in singular or plural form). If the number of such concepts is less than 100, the system randomly adds additional concepts to obtain such a set. As our experiments show, for a well defined concept (high quality and wide coverage of a given noun in the OMCS in the database), the former method is likely to find a slightly better set of automatically retrieved "web knowledge concepts". On the other hand, the later approach provides the means to discover new concepts, and allows "KnowY" to generate a database for terms that are not covered by the OMCS at all. We think that this feature is of prime importance for many applications in the open domain. The average sentence length found in the OMCS database was 9.32 words long. For the similarity score ranking we decided to include only "web sentences" and OMCS concepts with the number of words between 3 and 20. In the filtering process "KnowY" also uses information on the proportion of alphanumeric and special characters. To exclude the sentences that are unlikely to be valid entries to the knowledge database, "KnowY" compares them to the OMCS knowledge database concepts and ranks using the highest similarity score obtained, calculated with the following formula:

$$Similarity = \frac{\sum_{n=1}^{N} Wn * Wn_{ITF}}{L1 + L2} + \log Np * \alpha, \tag{1}$$

where: Wn — matching word found both in a OMCS concept and a "web sentence", Wn_{ITF} — inverted term frequency for a matching word (OMCS), L1 — number of words in a "web sentence", L2 — number of words in a OMCS concept, Np — number of concepts from OMCS where a noun was found in a position from a "web sentence", α - parameter (the value of α was set to 0.2). The formula takes account of the number of matching words between a "web sentence" and an OMCS concept as well as the importance of the given word in a used set of OMCS terms by the means of the ITF. The Np value promotes "web sentences", where a noun appears on a position, which is frequent for many OMCS concepts.

[2] In our experiments we used the snapshot of the OMCS database consisting of 700,000 sentences, available at http://commonsense.media.mit.edu/cgi-bin/download.cgi. From this set the sentences describing pictures and stories were removed.

6 Experiment Results

The preliminary experiments with the system were performed using a set of nouns, including ones that are frequently described in OMCS, as well as ones that are not covered at all in this database. Tables 3 and 4 show the examples of the knowledge concepts automatically discovered from the web; only the first five highest ranked statements are presented.

Table 3. Highest ranked knowledge concepts discovered by "KnowY" related to "water", domain (drinking, epa, treatment).

No.	Discovered Knowledge Concept	Sim. score
1	Sanitation means not only clean water but also clean air and clean soil.	1.33
2	Clouds are made when water vapor condenses into tiny droplets.	1.29
3	Most lakes are filled with fresh water, but there are a few lakes that are filled with salt.	1.25
4	When animals need water, they should not have to stand and wait.	1.17
5	Overwatering your yard can also cause water to run into the streets and into storm drains.	1.12

Table 4. Highest ranked knowledge concepts discovered by "KnowY" related to "apple", domain (tree, fruit, apples).

No.	Discovered Knowledge Concept	Sim. score
1	Apples bruise easily and must be hand picked.	1.61
2	Apple blossom is the state flower of Michigan.	1.58
3	Provide small samples different types of apples.	1.28
4	Apples have 5 seeds.	1.22
5	Apples are a member of the rose family.	1.21

As the results demonstrate, "KnowY" is able to automatically retrieve the knowledge concepts respecting the domain limitations with relatively high accuracy. Furthermore, the experiments showed that the system is capable of finding and automatically retrieving from the web the semantic equivalents of several of the entries that were inputted manually to the OMCS database. An example of such a statement discovered for the noun "apple", which is present among first five concepts with the highest similarity score is "Apple blossom is the state flower of Michigan", and its counterpart from the OMCS, "The Michigan state flower is the apple blossom". Some of the knowledge concepts

obtained from the web provide more detail compared to the OMCS entries. The instances of such "web knowledge concepts" include "Clouds are made when water vapor condenses into tiny droplets", " Apples bruise easily and must be hand picked", " Apples have 5 seeds" and the corresponding OMCS database entries, " Clouds are made up of water vapor"," When you drop an apple, it gets bruised", and " An apple contains seeds". With the exception of the statement " Provide small samples different types of apples"(Table 4, pos 3.) all remaining, automatically retrieved statements can be treated as valid entries to a knowledge database.

Less reliable set of results was obtained for the noun "golf" in the domain "clubs, instruction, equipment". As shown in Table 5, the majority of the discovered sentences can not be considered as valid knowledge concept since they convey personal experience/opinion (pos. 2), or strictly commercial information (pos. 4 and 5).

Table 5. Highest ranked knowledge concepts discovered by "KnowY" related to "golf", domain (clubs, instruction, equipment).

No.	Discovered Knowledge Concept	Sim. score
1	Golf is a great game that you can play for a lifetime...enjoy it!	1.02
2	The men I usually play golf with refused to play golf with me until I stepped back.	0.99
3	Good golf techniques are simple to learn but must be reinforced to be effective.	0.94
4	Over 600 courses, 200 hotels, and 50 plus golf resorts to choose from!	0.93
5	Golf for Beginners will provide you with the necessary help you need to get started correctly.	0.93

Table 6 and 7 present the results obtained for the noun "wasabi" and "Kendo". These nouns are covered only to a very limited extend in the OMCS ("wasabi" — 7 entries) or do not appear in this database ("Kendo"). For the similarity score calculation the comparison set included 93 and 100 randomly selected statements from the OMCS database, respectively for the nouns "wasabi" and "Kendo". The average similarity score is considerably lower, compared with the score obtained for the automatically discovered knowledge concept related to "water" or "apple". However, although the comparison set was composed of mostly randomly selected concepts from OMCS, in our opinion it did not significantly compromise the quality of the discovered concepts. The majority of retrieved statements related to "wasabi" and "Kendo" could be included in a knowledge database. The exception is sentence " This being the modern day Kendo's source of philosophy"(Table 7, pos. 3), which according to the standards used in the evaluation of the OMCS database would not be classified as a complete and valid knowledge concept.

Table 6. Highest ranked knowledge concepts discovered by "KnowY" related to "wasabi", domain (sushi, japanese, horseradish).

No.	Discovered Knowledge Concept	Sim. score
1	That is one reason that wasabi is served with sushi and raw fish slices.	0.91
2	Wasabi is very very dangerous.	0.86
3	Wasabi A green pungent horseradish paste served with sushi and sashimi.	0.82
4	What sushi lovers are unaware of however, is that Wasabi served in America is seldom real.	0.72
5	In Japan, sushi and sashimi are served with a condiment of wasabi mixed with soy sauce.	0.70

Table 7. Highest ranked knowledge concepts discovered by "KnowY" related to "Kendo", domain (sword, arm, Japanese).

No.	Discovered Knowledge Concept	Sim. score
1	Anger and true aggression has nothing to do with Kendo.	0.73
2	The bokken is used in modern kendo for kata practice.	0.67
3	This being the modern day Kendo's source of philosophy.	0.65
4	There are eight striking points in Kendo used for scoring.	0.62
5	Shiai geiko is the most competitive part of Kendo.	0.61

7 Conclusions and Future Work

This paper presented the method of automatic knowledge retrieval from the web. We think the idea described here and implemented in "KnowY" contributes to address one of the most important challenges in AI that exists today. The experiments performed with the system demonstrated that it is capable of automatically discovering several knowledge concepts in a user-selected domain with relatively high accuracy. Some of the automatically retrieved knowledge concepts provided semantic equivalents of the statements that were manually inputted to the OMCS. Others, while including more details compared to the OMCS entries, could also become a part of a knowledge database. The results confirmed also that the system was able to retrieve high quality knowledge concepts, even for the terms that were not described in the knowledge database built by mass collaboration of Internet users.

In our future work we are focusing on evaluating other techniques proposed for the document similarity in the information retrieval and computational linguistics literature; especially those that could be effectively applied for the comparison of short, sentence long documents. Some shortcomings of the described method are related to the nature of the resource that it is based on; while the web provides wide coverage of various domains, very

frequently the amount of commercial information dominates over these that could be considered as valid knowledge concepts. To address this problem, we intend to extend queries used to access "web knowledge concepts" with a short list of negative words (frequently used on the pages that contain advertisements), which should not appear on the pages "KnowY" accesses, as well as to extend the similarity measurement formula with the means to penalize the statements that include strictly personal opinions or information of only a commercial nature. Given that the described method is language independent, we intend also to evaluate it using languages other than English.

References

1. Alani H., Kim S., Millard D., Weal M., Lewis P., Hall W. and Shadbolt N. Automatic Ontology-based Knowledge Extraction and Tailored Biography Generation from the Web. *IEEE Intelligent Systems, 18(1)*, pages 14-21, 2003.
2. Borchardt G. Understanding Casual Descriptions of Physical Systems. In Proceedings of the Tenth National Conference on Artificial Intelligence, pages 2-8, 1992.
3. Inui T., Inui K. and Matsumoto Y. What Kind and Amount of Casual Knowledge Can Be Acquired from Text by Using Connective Markers as Clues? In the 6th International Conference on Discovery Science, pages 179-192, 2003.
4. Lenat D. CYC: A large-scale investment in knowledge infrastructure. Communications of the ACM, 38(11), pages 33-38, 1995.
5. Karypis G. A Clustering Toolkit. http://www.cs.umn.edu/~karypis/cluto. 2003.
6. Moldovan D., Girju R. and Rus V. Domain-Specific Knowledge Acquisition from Text. Proceedings of the Applied Natural Language Processing Conference, pages 268-275, 2000.
7. Richardson M. Domingos P. Building Large Knowledge Bases by Mass Collaboration. Proceedings of the Second International Conference on Knowledge Capture, pages 129-137, 2003.
8. Satoh H. Retrieval of simplified casual knowledge in text and its applications.
9. Rzepka R., Itoh T. and Araki K. Rethinking Plans and Scripts Realization in the Age of Web-mining. IPSJ SIG Technical Report 2004-NL-162, pages 11-18, 2004.
10. Singh P. The public acquisition of commonsense knowledge. In Proceedings of AAAI Spring Symposium: Acquiring (and Using) Linguistic (and World) Knowledge for Information Access, 2002.
11. Woods W. A Better way to Organize Knowledge. Technical Report of Sun Microsystems Inc., 1997.

Knowledge Visualization Using Optimized General Logic Diagrams

Bartłomiej Śnieżyński[1], Robert Szymacha[1], and Ryszard S. Michalski[2,3]

[1] Institute of Computer Science, AGH University of Science and Technology, Kraków, Poland
[2] Machine Learning and Inference Laboratory, George Mason University, Fairfax, VA, USA
[3] Institute of Computer Science, Polish Academy of Science
e-mail: sniezyn@agh.edu.pl, robert@bystrze.org, michalski@gmu.edu

Abstract. Knowledge Visualizer (KV) uses a General Logic Diagram (GLD) to display examples and/or various forms of knowledge learned from them in a planar model of a multi-dimensional discrete space. Knowledge can be in different forms, for example, decision rules, decision trees, logical expressions, clusters, classifiers, and neural nets with discrete input variables. KV is implemented as a module of the inductive database system VINLEN, which integrates a conventional database system with a range of inductive inference and data mining capabilities. This paper describes briefly the KV module and then focuses on the problem of arranging attributes that span the diagram in a way that leads to the most readable rule visualization in the diagram. This problem has been solved by applying a simulated annealing.

Keywords: knowledge visualization, GLD diagrams, diagram optimization, machine learning.

1 Introduction

Data and knowledge visualization tools are components of many commercial data mining tools, such as Microsoft SQL Server 2000 Analysis Services, IBM DB2 Intelligent Miner, and Oracle Data Miner. They are also present in noncommercial software, e.g. Weka [11] and YALE [10]. The authors are not aware, however, of any tool with capabilities of Knowledge Visualizer (KV) described in this paper. KV is a unique knowledge visualization system that uses a General Logic Diagram (GLD) for representing examples and knowledge derived from them.

GLD, introduced by Michalski [6], is a planar model of a multidimensional space spanned over discrete variables (attributes). Each variable partitions the diagram into a set of disjoint areas corresponding to the values of the variable. The lines separating these areas for a single attribute are called *axes* of this attribute. An intersection of the areas corresponding to single values of each variable constitutes a cell of the diagram. Every cell of the diagram thus represents a unique combination of attribute values.

Decision rules and decision trees are represented by regular configurations of cells. Logical operations AND,OR, and NOT correspond to intersection, union and complement, respectively, of the groups of cells representing arguments of the operations (Fig. 2).

Depending on the way the attributes are assigned to the axes of the diagram, knowledge representation in a diagram may be less or more readable and visually attractive. Therefore, a problem arises as to how to optimize the attribute assignment for representing a given form of knowledge. In this paper, we will be concerned with the optimization of the diagram for representing attrbutional rules learned by AQ-type learning program [8].

The KV program has been implemented as a module of inductive database system VINLEN [3] that aims at integrating conventional databases and a wide range of inductive inference capabilities.

The following sections review the concept of GLD, describe the KV program for visualizing examples and/or decision rules using GLD, and present a method for solving the above-mentioned attribute assignment problem. The working of KV is illustrated by an example.

2 General Logic Diagrams

General Logic Diagrams were developed by Michalski and applied in many different areas, such as the design and optimization of switching circuits, the optimization of logic functions, conversion of decision tables into decision trees, and, most widely, for visualizing concept examples and decision rules induced from them [7]. A GLD can be viewed as a multivalued extension of the Marquand's binary diagram [5] with some additional properties. Specifically, the diagram is structured by drawing axes of different variables with different thickness. Such a structuring facilitates the readability of knowledge represented in the diagram even when it is spanned over a relatively large number of attributes.

The procedure for creating a GLD (Generalized Logic Diagram) for representing a space $E(x_1, x_2, \ldots, x_n)$ spanned over n discrete attributes $x_1, x_2, \ldots, x_n$, with domains $D_1, D_2, \ldots, D_n$, is as follows:

1. Divide attributes $x_1, x_2, \ldots, x_n$ into two subsets: *Row-att* and *Coll-att*.
2. Order attributes in these subsets.
3. Order values in the domains of nominal attributes.
4. Draw a rectangle, and divide it into the numer of rows equal the number of values of the first attribute in *Row-att*. Draw the separating lines with the greatest chosen thickness. These lines constitute the axes of the first attribute. Assign to rows values of the first attribute.
5. Divide each row into the number of subrows equal the number of values of the second attribute in *Row-att*. Separating lines, which are axes of the second attribute, are drawn with the lower thickness than the axes

of the previous (first) attribute. Continue such process for the remaining attributes in *Row-att*.

6. Repeat the same process for attributes in *Coll-att* by dividing the rectangle into columns.

Individual cells in the so-created diagram correspond to single events, that is, to distinct combinations of attribute values. To represent a function that maps the event space (the cartesian product of attribute domains) into the domain of output variable, the cells of the diagram are marked by function values. Thus, every such function can represented in the diagram. Because standard decision rules, attributional rules, decision trees, neural nets, and other forms of knowledge involving discrete input attributes (or discrtized continuous attributes) represent functions over an event space, they all can be represented using GLDs.

Single decision or attributional rules correspond to regular configurations of cells in the diagram, and thus can be easily recognized visually even in diagrams with a relatively large number of attributes (their number depends on the sizes of attribute domains).

When the event space is large, the diagram can be spanned only over the attributes that are present in the rules to be represented in the diagram (whose number if usually significantly smaller than the total number of attributes).

To illustrate a GLD rule presentation, let us use the attributional rules generated by a learning program for very simple robot domain. The rules are listed in Fig. 1.

```
A robot is classified as friendly, if:
  - its body is round, or
  - its head is triangular.
A robot is classified as unfriendly, if:
  - the shape of its head is round or square,
    and its body is square or triangular,
    and it is holding a balloon or a sword, or
  - its body is triangular,
    and the antenna is green.
A robots is unclassified, if:
  - its head is pentagonal or square,
    and it is holding a flag.
```

Fig. 1. Rules generated using AQ21 program for simple robot domain

These rules are visualized using GLD that is presented in Fig. 2. The following shortcuts are used in the diagram:

- attribute names: holding=$x0$, head=$x1$, antenna=$x2$, body=$x3$;

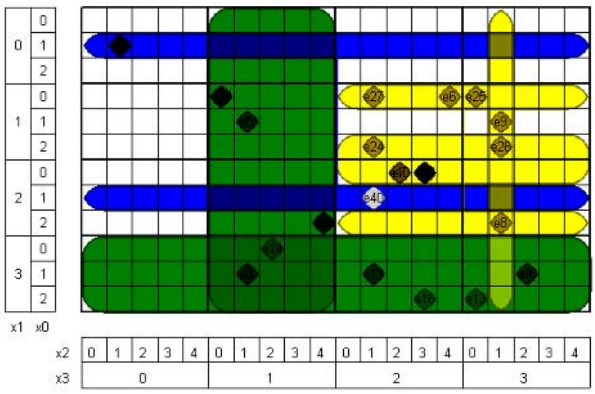


Fig. 2. GLD for rules generated from robots data

- holding values: balloon=0, flag=1, sword=2;
- body and head values: pentagon=0, round=1, square=2, triangle=3;
- antenna values: blue=0, green=1, red=2, white=3, yellow=4.

As we can see, *Coll-att*= $\{x3, x2\}$, *Row-att*= $\{x1, x0\}$. Big rectangles in the middle and on the bottom represent rules for class "good". Two long, horizontal rectangles represent rule for class "do_not_know". Rectangles on the right represent rules for class "bad". Circles represent events and their identifiers used to generate rules. They are darker if more then one event are printed is the same cell.

3 Current capabilities of the KV module

The KV program was implemented as a module of the VINLEN system. Input data (event space definition, examples, and rules) are taken from the VINLEN database. The module has following capabilities:

1. It can automatically draw a GLD for the given attributes.
2. The assignment of axes to attributes can be done by the user or automatically by the program.
3. It can represent training and testing examples for a given learning problem.
4. User can add events by choosing cells on a diagram.
5. It can automatically visualize attributional rules learned by AQ-type learning program and supplied to KV via VINLEN. Rules are represented by collections of linked rounded rectangles. Rectangles corresponding to rules of the same class are given the same color.
6. A diagram can be enlarged (to see details) or decreased (to fit the computer screen).

7. A diagram can be printed or saved as a graphic file. The user can choose the way the diagram is printed: in black-and-white or in color.
8. The assignment of axes of attributes can be automatically optimized to improve the rule visualization. Because the number of possible assignments can be very large, automatic optimization is very useful. An optimization method is described in the next Section.

4 Diagram optimization algorithm

To optimize the assignment of axes to attributes two features of the diagram are measured:

1. $C(d)$, the number of compact regions in the diagram, d, that represent given set of attributional rules.
2. $S(d)$, the closeness of the ratio of the height to width of the diagram to the golden ratio, $S(d) = |width(d)/height(d) - 1.62|$.

The following diagram cost function aggregating these features is used:

$$f(d) = C(d) + pS(d), \tag{1}$$

where p is a user defined parameter (integer number) representing the importance of shape.

In the current implementation, simulated annealing algorithm is used to minimize f. This is done in a similar way as in graph optimization [2]. Details of the algorithm can be found in [4]. Generally, it works as follows. Initial partitioning and ordering is randomized. Next, the algorithm makes random changes in partitioning and orderings. Probability to switch to the state with worse diagram (higher f value) depends on a "temperature," which is initially high, but reduced during simulation by multiplying by user-defined dT factor. After user-defined number of steps, the best diagram is presented to the user.

5 Experiments

The experiments were designed to test the performance of the KV module. In the experiment described here, the UCI's Mushroom dataset was used [1]. Two sets of rules were generated: RS1 using AQ21 program in strong patterns mode[1] for "cap-surface" target attribute, (see Fig. 3), and RS2 using C4.5 algorithm for "classes" target attribute.

Diagrams were optimized for displaying RS1 and RS2. Optimization was performed 10 times with the following parameters: 100 steps, initial temperature $T = 1$, temperature factor dT=0.9, and shape importance p=100. Diagrams are presented in Fig. 4, and 5. Axes descriptions are omitted because cells are very small. Average execution time on a Pentium 4 2.4GHz machine was 143 seconds for RS1 and 81 seconds for RS2.

[1] Rules generated using AQ21 in the strong pattern mode are simple but approximate.

```
cap-surface is FIBROUS, if:
  - gill-color is BLACK,BROWN,CHOCOLATE,GRAY,PINK,PURPLE;
  - classes is EDIBLE;
cap-surface is GROOVES, if:
  - stalk_surface_above_ring is SMOOTH;
cap-surface is SCALY, if:
  - cap-color is not WHITE;
cap-surface is SMOOTH, if:
  - cap-shape is BELL,KNOBBED;
```

Fig. 3. RS1 rules generated using AQ21 program, with "cap-surface" target attribute and strong patterns mode

6 Conclusion and Further Research

The KV module allows one to represent graphically rules and data from a VINLEN knowledge system. The visualization method uses the General Logic Diagram. KV allows one to check how rules cover given training examples, and to analyze the relationship between the rules and the examples.

The integration of the KV module in the VINLEN system facilitates experimental investigation of the rule learning process. The diagram optimization feature improves the readability of the rules in the diagram.

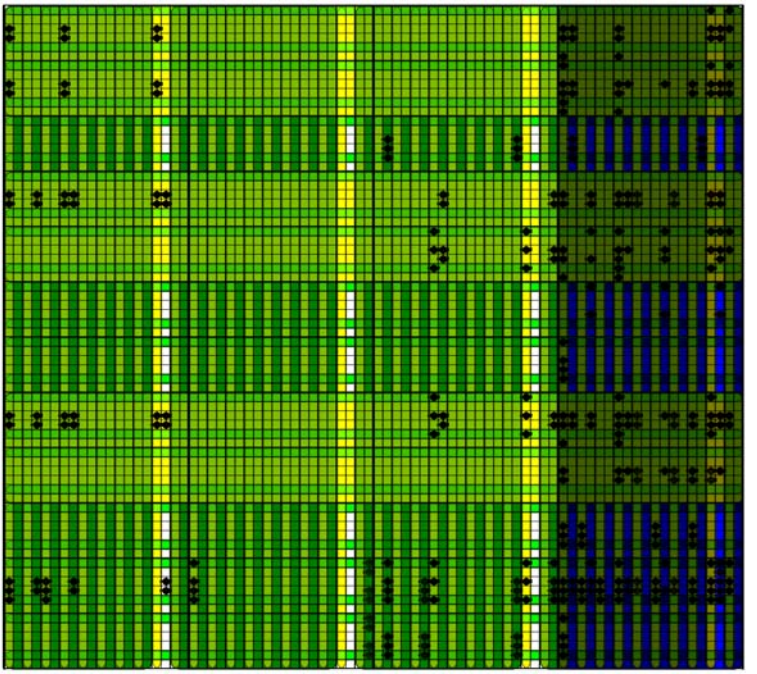
Further research will concern adding more features to the KV module, in particular, the ability to:

- display results of constructive induction;
- display results of abstraction and concretion operation on the representation space in the case of structured input attributes;
- place images in the cells, which will enable KV to visualize more complex knowledge;
- represent decision trees and results of clustering algorithms for unsupervised learning;

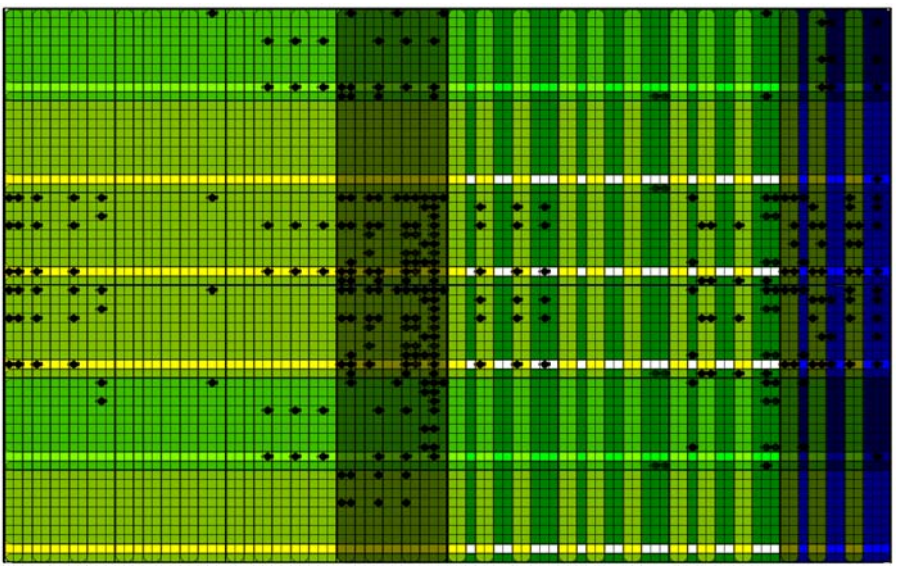
It is also planned to improve the current diagram optimization algorithm, for example, by applying the Learnable Evolution Model [9]. An associated problem is to develop and test other cost functions that take into consideration other diagram features. Another important problem is how to represent large representation spaces for which diagrams are too complex. A method of automatic generation of subsets of attributes to span different diagrams (views) needs to be developed.

References

1. C.L. Blake and C.J. Merz. UCI repository of machine learning databases, 1998.
2. R. Davidson and D. Harel. Drawing graphs nicely using simulated annealing. *ACM Transactions on Graphics*, 15(4):301–331, 1996.
3. K. A. Kaufman and R. S. Michalski. The development of the inductive database system VINLEN: A review of current research. In M. Kłopotek et al., editor, *Intelligent Information Processing and Web Mining*, Advances in Soft Computing, pages 393–398. Springer, 2003.
4. S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by genetic annealing. *Science*, (220):671–680, 1983.
5. A. Marquand. On logical diagrams for n terms. *The London, Edinburgh and Dublin Philosophical Magazine and Journal of Science*, XII(75):266–270, 1881.
6. R. S. Michalski. Graphical minimization of normal expressions of logic functions using tables of the veitch-karnaugh type. *Journal of the Institute of Automatic Control*, (52), 1978.
7. R. S. Michalski. A planar geometrical model for representing multi-dimensional discrete spaces and multiple-valued logic functions. Technical Report 897, Department of Computer Science, University of Illinois, Urbana, 1978.
8. R. S. Michalski. *Attributional Calculus: A Logic and Representation Language for Natural Induction*. Reports of the Machine Learning and Inference Laboratory, MLI 04-2. George Mason University, 2004.
9. R. S. Michalski, K. A. Kaufman, and G. Cervone. Speeding up evolution through learning: LEM. In M. Kłopotek et al., editor, *Proceedings of the Ninth International Symposium on Intelligent Information Systems*, 2000.
10. I. Mierswa, R. Klinkberg, S. Fischer, and O. Ritthoff. A Flexible Platform for Knowledge Discovery Experiments: YALE – Yet Another Learning Environment. In *LLWA 03 - Tagungsband der GI-Workshop-Woche Lernen - Lehren - Wissen - Adaptivitt*, 2003.
11. I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, 1999.

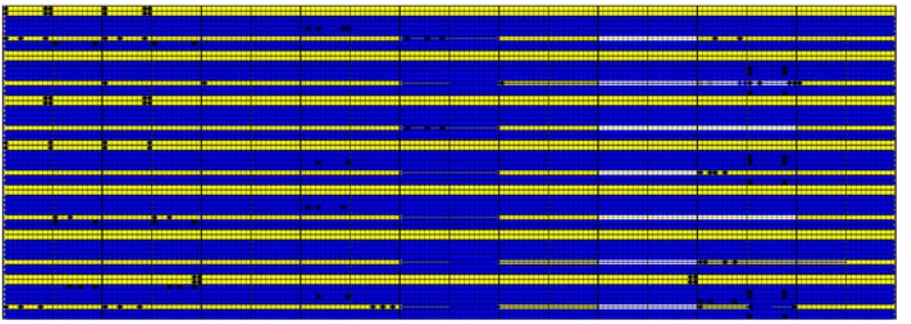


(a)

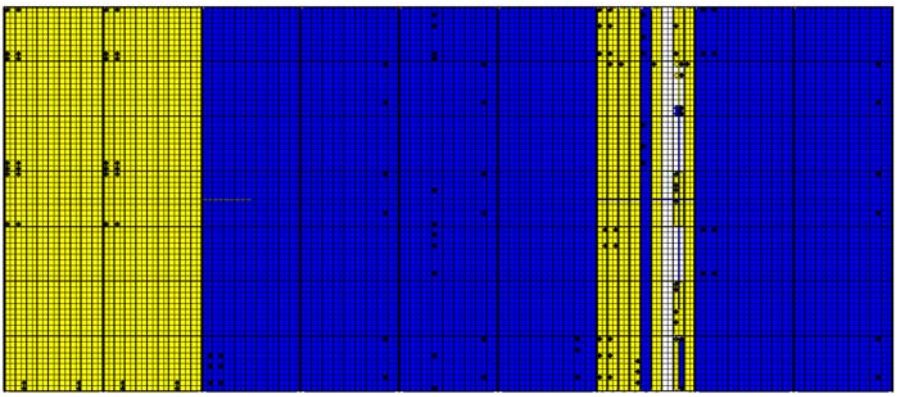


(b)

Fig. 4. GLD Diagrams for rules RS1: initial (a) and optimized (b)



(a)



(b)

Fig. 5. Diagrams for rules RS2: initial (a) and optimized (b)

Efficient Processing of Frequent Itemset Queries Using a Collection of Materialized Views*

Marek Wojciechowski and Maciej Zakrzewicz

Poznan University of Technology, ul. Piotrowo 3a, Poznań, Poland

Abstract. One of the classic data mining problems is discovery of frequent itemsets. Frequent itemset discovery tasks can be regarded as advanced database queries specifying the source dataset, the minimum support threshold, and optional constraints on itemsets. We consider a data mining system which supports storing of results of previous queries in the form of materialized data mining views. Previous work on materialized data mining views addressed the issue of reusing results of one of the previous frequent itemset queries to efficiently answer the new query. In this paper we present a new approach to frequent itemset query processing in which a collection of materialized views can be used for that purpose.

1 Introduction

Frequent itemset mining is one of the classic data mining problems, identified as the key step in association rule discovery [1]. Frequent itemsets and association rules capture the co-occurrence of items in the collection of sets, and find numerous applications including market basket analysis and web usage mining. Frequent itemset mining can be seen as advanced querying [6], where a user specifies the source dataset, the minimum support threshold, and optionally some constraints on itemsets, then the system chooses the appropriate data mining algorithm and returns the results to the user. Data mining query processing has recently become an important research area, focusing mainly on constraint handling and reusing results of previous queries.

We consider a data mining system which supports storing of results of previous queries in the form of materialized data mining views [7]. In our previous work [10] we addressed the issue of reusing results of one of the previous frequent itemset queries to efficiently answer the new query. In this paper we present a new approach to frequent itemset query processing in which a collection of materialized views can be used for that purpose. We propose a query execution method that uses partial results from a set of materialized views, and provide an algorithm that selects a set of materialized views that is optimal in terms of the I/O cost.

* This work was partially supported by the grant no. 4T11C01923 from the State Committee for Scientific Research (KBN), Poland.

1.1 Background

Frequent itemsets. Let $L = \{l_1, l_2, ..., l_m\}$ be a set of literals, called *items*. Let a non-empty set of items T be called an *itemset*. Let D be a set of variable length itemsets, where each itemset $T \subseteq L$. We say that an itemset T *supports* an item $x \in L$ if x is in T. We say that an itemset T *supports* an itemset $X \subseteq L$ if T supports every item in the set X. The *support* of the itemset X is the percentage of itemsets in D that support X. The problem of mining frequent itemsets in D consists in discovering all itemsets whose support is above a user-defined support threshold *minsup*.

Apriori algorithm. *Apriori* [2] is a classic algorithm for frequent itemset discovery. It makes multiple passes over the input data to determine all frequent itemsets. Let L_k denote the set of frequent itemsets of size k and let C_k denote the set of candidate itemsets of size k. Before making the k-th pass, *Apriori* generates C_k using L_{k-1}. Its candidate generation process ensures that all subsets of size $k-1$ of C_k are all members of the set L_{k-1}. In the k-th pass, it then counts the support for all the itemsets in C_k. At the end of the pass all itemsets in C_k with a support greater than *minsup* form the set of frequent itemsets L_k.

1.2 Related Work

Incremental mining in the context of frequent itemsets was first discussed in [4]. A novel algorithm called *FUP* was proposed to efficiently discover frequent itemsets in an incremented dataset, exploiting previously discovered frequent itemsets. *FUP* was based on the same generate-and-test paradigm as *Apriori* — it is bound to a particular mining methodology. Using our terminology, *FUP* exploits one materialized view, and cannot be easily extended to use more than one view.

In [8] the authors postulated to create a knowledge cache that would keep recently discovered frequent itemsets. Besides presenting the notion of knowledge cache the authors introduced several maintenance techniques for such cache, and discussed using the cache contents when answering new frequent set queries.

In [3] relationships between association rule queries were analyzed. They represented cases when results of one query can be used to efficiently answer the other. However, the relationships concerned association rules — not frequent itemsets.

The work on materialized views started in the 80s. The basic concept was to use materialized views as a tool to speed up queries. Since then, materialized views have become a key element of data warehousing technology (see [9] for an overview).

2 Frequent Itemset Query Execution Using a Single Materialized Data Mining View

In [10] we considered the problem of executing a data mining query using a single materialized data mining view. In this section we review the basic definitions and summarize the results of our previous study.

Definition 1 (Data mining query). *A data mining query for frequent itemset discovery* is a tuple $dmq = (\mathcal{R}, a, \Sigma, \Phi, \beta)$, where $\mathcal{R}$ is a database relation, a is a set-valued attribute of $\mathcal{R}$, Σ is a data selection predicate on $\mathcal{R}$, Φ is a selection predicate on frequent itemsets, β is the minimum support for the frequent itemsets. The data mining query dmq returns all frequent itemsets discovered in $\pi_a \sigma_\Sigma \mathcal{R}$, having support greater than β and satisfying the constraints Φ.

Example. Given is the database relation $\mathcal{R}_1(attr_1, attr_2)$. The data mining query $dmq_1 = (\mathcal{R}_1, "attr_2", "attr_1 > 5", "|itemset| < 4", 3)$ describes the problem of discovering frequent itemsets in the set-valued attribute $attr_2$ of the relation $\mathcal{R}_1$. The frequent itemsets with support above 3 and length less than 4 are discovered in records having $attr_1 > 5$.

Definition 2 (Materialized data mining view). *A materialized data mining view* $dmv = (\mathcal{R}, a, \Sigma, \Phi, \beta)$ is a data mining query, whose both the definition and the result are permanently stored (materialized) in a database. All frequent itemsets being the result of the data mining query are called *materialized data mining view contents.*

For a frequent itemset query $dmq = (\mathcal{R}, a, \Sigma_{dmq}, \Phi_{dmq}, \beta_{dmq})$, and a materialized view $dmv_1 = (\mathcal{R}, a, \Sigma_1, \Phi_1, \beta_1)$, we identified two general cases, presented below, in which dmq can be answered using dmv_1. The cases are described in terms of relationships between selection predicates. $\Sigma_1 \subset \Sigma_{dmq}$ means that the source dataset of dmv_1 is a subset of the source dataset of dmq (e.g., "$attr_1 > 5$" $\subset$ "$attr_1 > 2$"). $\Phi_1 \subseteq \Phi_{dmq}$ means that if an itemset satisfies Φ_{dmq} then it must also satisfy Φ_1 (e.g., "$|itemset| < 5$" $\subseteq$ "$|itemset| < 3$"). See [10] for details.

Verifying Mining (VM): $(\Sigma_1 = \Sigma_{dmq} \wedge \beta_1 \leq \beta_{dmq} \wedge \Phi_1 \subseteq \Phi_{dmq})$. Since the materialized data mining view dmv_1 contains a superset of the result of dmq, then the execution of dmq takes to read the contents of dmv_1 and filter the frequent itemsets with respect to β_{dmq} and Φ_{dmq}. (In a particular case, when $\beta_1 = \beta_{dmq}$ and $\Phi_1 = \Phi_{dmq}$, dmv_1 contains the exact result of dmq, and no filtering is needed.)

Incremental Mining (IM): $(\Sigma_1 \subset \Sigma_{dmq} \wedge \beta_1 \leq \beta_{dmq} \wedge \Phi_1 \subseteq \Phi_{dmq})$. The database has been logically divided into two partitions: (1) the records

covered by the view dmv_1, (2) the records covered by the query dmq, and not covered by dmv_1. The execution of dmq consists of three steps. In the first step, the contents of dmv_1 are filtered with respect to β_{dmq} and Φ_{dmq}. (In a particular case, when $\beta_1 = \beta_{dmq}$ and $\Phi_1 = \Phi_{dmq}$ this step is not needed.) In the second step, all itemsets frequent in the second partition are discovered using a complete mining algorithm (e.g., *Apriori*). Finally, locally frequent itemsets from both partitions are merged and then counted during the scan of the database in order to find globally frequent itemsets.

3 Frequent Itemset Query Execution Using a Set of Materialized Data Mining Views

Problem formulation. Given are: (1) a set of materialized data mining views $DMV = \{dmv_1, dmv_2, ..., dmv_n\}$, where $dmv_i = (\mathcal{R}, a, \Sigma_i, \Phi_i, \beta_i)$, Σ_i is of the form $(l^i_{1min} < a < l^i_{1max}) \vee (l^i_{2min} < a < l^i_{2max}) \vee ... \vee (l^i_{kmin} < a < l^i_{kmax})$, $l^i_* \in dom(a)$, and (2) a data mining query $dmq = (\mathcal{R}, a, \Sigma_{dmq}, \Phi_{dmq}, \beta_{dmq})$, where Σ_{dmq} is of the form $(l^{dmq}_{1min} < a < l^{dmq}_{1max}) \vee (l^{dmq}_{2min} < a < l^{dmq}_{2max}) \vee ... \vee (l^{dmq}_{mmin} < a < l^{dmq}_{mmax})$, $l^{dmq}_* \in dom(a)$. The problem of *materialized data mining view selection* consists in generating such an algorithm of dmq execution using views from DMV, that its I/O cost is minimal.

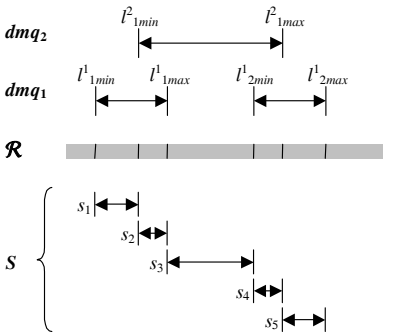


Fig. 1. Sample set of data mining queries and their distinct selection formulas

Definition 3 (Data sharing graph). Let $S = \{s_i, s_2, ..., s_k\}$ be a set of *distinct selection formulas* for $DMV \cup \{dmq\}$, i.e., a set of such selection formulas over the attribute a of $\mathcal{R}$ that for each i, j we have $\sigma_{s_i}\mathcal{R} \cap \sigma_{s_j}\mathcal{R} = \emptyset$, for each i there exist integers $a, b, ..., m$, such that $\sigma_{\Sigma_i}\mathcal{R} = \sigma_{s_a}\mathcal{R} \cup \sigma_{s_b}\mathcal{R} \cup ... \cup \sigma_{s_m}\mathcal{R}$, and there exist integers $a, b, ..., k$, such that $\sigma_{\Sigma_{dmq}}\mathcal{R} = \sigma_{s_a}\mathcal{R} \cup \sigma_{s_b}\mathcal{R} \cup ... \cup \sigma_{s_k}\mathcal{R}$ (Fig. 1). A graph $DSG = (V, E)$ will be called a *data sharing graph*

for DMV and a data mining query dmq if and only if $V = DMV \cup S \cup \{dmq\}$, $E = \{(v, s_j) | v \in DMV \cup \{dmq\},\ s_j \in S,\ \sigma_{\Sigma_v}\mathcal{R} \cap \sigma_{s_j}\mathcal{R} \neq \emptyset\}$.

Example. Let us consider a data sharing graph for a set of materialized data mining views and a data mining query. Given is a relation $\mathcal{R}_1 = (attr_1, attr_2)$, four materialized data mining views: $dmv_1 = (\mathcal{R}_1, "attr_2", "10 < attr_1 < 30", \emptyset, 10)$, $dmv_2 = (\mathcal{R}_1, "attr_2", "15 < attr_1 < 40", \emptyset, 4)$, $dmv_3 = (\mathcal{R}_1, "attr_2", "30 < attr_1 < 40", \emptyset, 8)$, $dmv_4 = (\mathcal{R}_1, "attr_2", "5 < attr_1 < 30", \emptyset, 15)$, and a data mining query $dmq = (\mathcal{R}_1, "attr_2", "5 < attr_1 < 40", \emptyset, 10)$. The set of distinct selection formulas consists of the following elements: $S = \{s_1 = "5 < attr_1 < 10", s_2 = "10 < attr_1 < 15", s_3 = "15 < attr_1 < 30", s_4 = "30 < attr_1 < 40"\}$. The data sharing graph for $DMV = \{dmv_1, dmv_2, dmv_3, dmv_4\}$ and dmq is shown in Fig. 2.

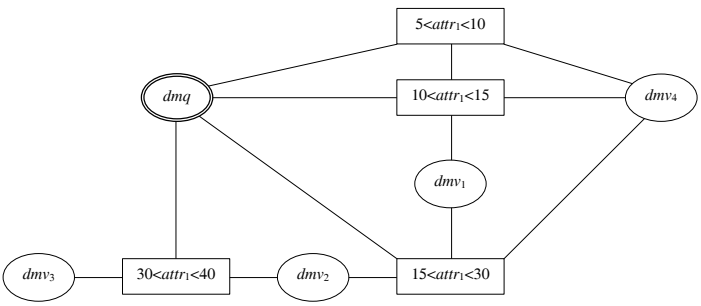


Fig. 2. Sample data sharing graph for a set of materialized data mining views and a data mining query

3.1 Materialized data mining view selection

Consider a data mining query dmq which can be executed using multiple data mining views from DMV. According to our previous analysis, dmq can use such views from DMV that: (1) are based on a subset of dmq's source dataset ($\Sigma_i \subseteq \Sigma_{dmq}$), (2) use $minsup$ which is not above the $minsup$ of dmq ($\beta_i \leq \beta_{dmq}$), and (3) their data selection conditions are identical to or relaxed with respect to dmq ($\Phi_i \subseteq \Phi_{dmq}$). Moreover, the source datasets of the materialized views must not overlap, since in such case, certain frequent itemsets might be lost if they were not frequent in any of the source datasets. Therefore, in order to efficiently execute the data mining query $dmq = (\mathcal{R}, a, \Sigma_{dmq}, \Phi_{dmq}, \beta_{dmq})$, we should consider the following subsets of materialized views:

$$DMV_{dmq} = \{dmv_i \in DMV | dmv_i = (\mathcal{R}, a, \Sigma_i, \Phi_i, \beta_i), \Sigma_i \subseteq \Sigma_{dmq}, \quad (1)$$
$$\beta_i \leq \beta_{dmq}, \Phi_i \subseteq \Phi_{dmq}, \forall dmv_k \in DMV_{dmq} - \{dmv_i\} : \Sigma_i \cap \Sigma_k = \emptyset\}.$$

Using DMV_{dmq} to execute dmq consists in: (1) selecting frequent itemsets from each $dmv_i \in DMV_{dmq}$ with supports above β_{dmq}, (2) discovering frequent itemsets in the portion of the database, which is not covered by any view from DMV_{dmq}, (3) merging all frequent itemsets from (1) and (2) to form a set of global candidates, and (4) scanning the database to evaluate their support and prune infrequent ones. The algorithm for executing a data mining query dmq using a set DMV_{dmq} of materialized data mining views is shown below.

Algorithm 1 (Data mining query execution using a set of materialized data mining views).
Input: A data mining query dmq and a set of materialized data mining views DMV_{dmq}
Output: The results of dmq.
Method:

1. **for each** $dmv_i \in DMV_{dmq}$ **do**
2. $\quad \mathcal{F}_i \leftarrow \{$*frequent itemsets from* dmv_i *having support* $\geq \beta_{dmq}$ *and satisfying* $\Phi_{dmq}\}$
3. $\mathcal{R}' \leftarrow \sigma_{\Sigma_{dmq}}\mathcal{R} - - - \bigcup_{dmv_i \in DMV_{dmq}} \sigma_{\Sigma_i}\mathcal{R})$
4. $\mathcal{F}' \leftarrow execute(\mathcal{R}', a, \text{"}true\text{"}, \Phi_{dmq}, \beta_{dmq})$
5. $\mathcal{C} \leftarrow \mathcal{F}' \cup \mathcal{F}_1 \cup \mathcal{F}_2 \cup ... \cup \mathcal{F}_{|DMV_{dmq}|}$
6. $count(\mathcal{C}, \sigma_{\Sigma_{dmq}}\mathcal{R})$
7. $Answers \leftarrow \{C \in \mathcal{C} | C.count \geq \beta_{dmq}\}$
8. **return** $Answers$

Since many applicable subsets of materialized data mining views may exist, we will look for such DMV_{dmq} that the I/O cost of dmq execution is minimal. The I/O cost of executing dmq using DMV_{dmq} is the following:

$$cost_{DMV_{dmq}} = \sum_{dmv_i \in DMV_{dmq}} ||dmv_i|| + k * ||\mathcal{R}'|| + ||\sigma_{\Sigma_{dmq}}\mathcal{R}||, \qquad (2)$$

where $\mathcal{R}'$ represents the portion of $\sigma_{\Sigma_{dmq}}\mathcal{R}$ which is not covered by any view from DMV_{dmq}, and k is the number of scans of $\mathcal{R}'$ required by a complete mining algorithm (e.g., *Apriori*) that has to be run to discover locally frequent itemsets in $\mathcal{R}'$.

In order to estimate the benefits from using multiple data mining views, let us compare the above cost with the cost of running a regular algorithm on the complete dataset. Assuming that the complete mining algorithm would need the same number k of dataset scans on the whole database and the portion of it not covered by materialized views, the materialized data mining views should be used if:

$$\sum_{dmv_i \in DMV_{dmq}} ||dmv_i|| + k * ||\mathcal{R}'|| < (k-1) * ||\sigma_{\Sigma_{dmq}}\mathcal{R}||. \qquad (3)$$

Since in practical applications we have $||dmv_i|| << ||\sigma_{\Sigma_i}\mathcal{R}||$, then the larger coverage of dmq's source dataset by the views from DMV_{dmq}, the better

performance of dmq's execution. Therefore we look for such a DMV_{dmq} that provides the largest coverage of dmq's source dataset.

Algorithm 2 (Materialized data mining view selection).
Input: A data sharing graph $DSG = (DMV \cup \{dmq\}, E)$, a set of distinct selection formulas S, a data mining query $dmq = (\mathcal{R}, a, \Sigma_{dmq}, \Phi_{dmq}, \beta_{dmq})$
Output: DMV_{dmq} providing the largest coverage of dmq's source dataset.
Method:

begin
1. $P \leftarrow \{s \in S | (dmq, s) \in E\}$
2. $SV \leftarrow \{dmv \in DMV | \forall s \in S, (dmv, s) \in E \Rightarrow s \in P\}$
3. $AV \leftarrow \{dmv \in SV | dmv = (\mathcal{R}, a, \Sigma, \Phi, \beta), \beta \leq \beta_{dmq}, \Phi \subseteq \Phi_{dmq}\}$
4. **global** $optCost \leftarrow +\infty$
5. **global** $OptViewSet \leftarrow \emptyset$
6. $scanViewSets(AV, \emptyset)$
7. **return** $OptViewSet$

end

8. **procedure *scanViewSets(V, Seed)*:**

begin
9. **for all** $dmv \in V$ **do begin**
10. $newCost \leftarrow cost(Seed \cup \{dmv\}, dmq)$
11. **if** $newCost < optCost$ **then**
12. $optCost \leftarrow newCost$
13. $optViewSet \leftarrow Seed \cup \{dmv\}$

end if
14. $V \leftarrow V - \{dmv\}$
15. $OL \leftarrow \{dmv_j | \exists s, (dmv, s) \in E \wedge (dmv_j, s) \in E\}$
16. $scanViewSets(V - OL, Seed \cup \{dmv\})$

end

The above algorithm constructs a DMV_{dmq} providing the largest coverage of dmq's source dataset. In step (1) we build the set P of distinct selection formulas for dmq. In step (2) we select materialized data mining views (SV) containing such data selection conditions that are relaxed with respect to dmq's data selection conditions. In step (3) we select such views from SV that use $minsup$ not above dmq's $minsup$ and use pattern selection conditions identical to or relaxed with respect to dmq. In step (4) we initialize a global variable to hold the best I/O cost found so far. In step (5) we initialize a global variable to hold the best set of materialized data mining views found so far. In step (6) we call the recursive procedure to generate all allowable subsets AV of the materialized data mining. In step (7) we return the optimal set of materialized data mining views for the execution of dmq. In step (8) the procedure called $scanViewSets$ begins; it scans all allowable subsets of V, and it appends $Seed$ to each of them. In step (9) a loop begins to iterate

over all materialized data mining views from V. In step (10) we evaluate the I/O cost of executing *dmq* using the set of materialized data mining views. If the cost is lower that the best one found so far, then in steps (12)-(13) we update the global variables. In step (14) we eliminate the current view from the set not to process it again during recursive calls. In step (15) we select all materialized data mining views which share at least one distinct data selection formula with the current view. In step (16) we recursively call *scanViewSets* to append more views to the current set.

4 Experimental Results

In order to evaluate performance of frequent itemset discovery using a collection of materialized views, we performed several experiments on a Pentium II 433MHz PC with 128 MB of RAM. The experiments were conducted on the MSWeb[1] (Microsoft Anonymous Web Data) dataset from the UCI KDD Archive [5]. The dataset contained 32710 transactions (user sessions), the average size of transaction was 3 URLs, and the total number of different URLs in the dataset was 285. As a complete data mining algorithm we used our implementation of *Apriori*.

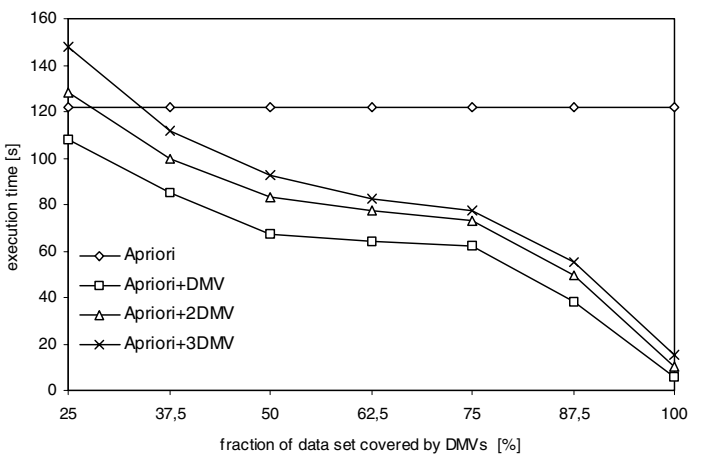


Fig. 3. Execution times for different numbers of used materialized views

Figure 3 presents the execution times for a different number of selected materialized views (1, 2, and 3) compared to the execution time of the complete *Apriori* algorithm. For the cases involving materialized views we varied the fraction of the dataset covered by them from 25% to 100%. The minimum

[1] http://kdd.ics.uci.edu/databases/msweb/msweb.html

support threshold of the query to be answered was by 0.1 higher than the average minimum support threshold of the selected materialized views.

The experiments show that using materialized views reduces processing time if the views cover a significant fraction of the original query's dataset. Already for materialized views covering 37.5% of the source dataset, using up to 3 views paid off. The best performance was achieved when the selected views covered the whole dataset — in such cases there was no need to run *Apriori* at all.

Regarding the number of selected materialized views, we observe that for the same level of coverage, the smaller the number of views the better. One reason for performance degradation due to using a higher number of views is that using more views means higher I/O costs as more views have to be read from disk. This effect is particularly noticeable for small datasets like MSWeb. Another issue that might affect performance even in case of large datasets is the distribution of values in the dataset. If the distribution is strongly skewed, each of the views will introduce many locally frequent patterns, which then have to be counted (verified) in the whole dataset.

5 Conclusions

In this paper we discussed answering a frequent itemset query using a collection of materialized data mining views. The proposed method is independent of a particular mining algorithm, and is a generalization of the method from our previous work that was capable of using only one view.

We have presented: (1) the conditions on a set of materialized views that have to be fulfilled for the set to be useful, (2) the algorithm that uses results from a set of materialized views to answer a new frequent itemset query, (3) the algorithm that finds the subset of applicable materialized views that is optimal in terms of I/O costs.

The experiments confirm our theoretical analysis and show that using a set of materialized views is an efficient solution, provided that the views cover a significant part of the dataset.

References

1. Agrawal, R., Imielinski, T., Swami, A. (1993) Mining Association Rules Between Sets of Items in Large Databases. Proceedings of the 1993 ACM SIGMOD Conference on Management of Data, Washington, D. C., 207–216
2. Agrawal, R., Srikant, R. (1994) Fast Algorithms for Mining Association Rules. Proceedings of the 20th International Conference on Very Large Data Bases, Santiago de Chile, Chile, 487–499
3. Baralis, E., Psaila, G. (1999) Incremental Refinement of Mining Queries. Proceedings of the 1st International Conference on Data Warehousing and Knowledge Discovery (DaWaK), Florence, Italy, 173–182

4. Cheung, D. W.-L., Han, J., Ng, V., Wong, C. Y. (1996) Maintenance of discovered association rules in large databases: An incremental updating technique. Proceedings of the 12th International Conference on Data Engineering, New Orleans, Louisiana, USA, 106–114
5. Hettich, S., Bay, S. D. (1999) The UCI KDD Archive [http://kdd.ics.uci.edu]. Irvine, CA: University of California, Department of Information and Computer Science
6. Imielinski, T., Mannila, H. (1996) A Database Perspective on Knowledge Discovery. Communications of the ACM **39**(11), 58–64
7. Morzy, T., Wojciechowski, M., Zakrzewicz, M. (2000) Materialized Data Mining Views. Proceedings of the 4th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD 2000), Lyon, France, 65–74
8. Nag, B., Deshpande, P. M., DeWitt, D. J. (1999) Using a Knowledge Cache for Interactive Discovery of Association Rules. Proceedings of the 5th International Conference on Knowledge Discovery and Data Mining, San Diego, California, 244–253
9. Roussopoulos, N. (1998) Materialized Views and Data Warehouses. SIGMOD Record, **27**(1), 21–26
10. Zakrzewicz, M., Morzy, M., Wojciechowski, M. (2004) A Study on Answering a Data Mining Query Using a Materialized View. Proceedings of the 19th International Symposium on Computer and Information Sciences (ISCIS'04), Kemer — Antalya, Turkey, 493–502

Part II

Regular Sessions: Computational Linguistics

GramCat and GramEsp: two grammars for chunking

Montserrat Civit and M. Antònia Martí

CLiC Centre de Llenguatge i Computació
C/ Adolf Florensa s/n 08028 Barcelona, Spain

Abstract. In this article[1] we present two grammars (GramCat and GramEsp) for chunking of unrestricted Catalan and Spanish texts. With these grammars we extend the classical notion of *chunk* as it is defined by Abney, taking advantage of Catalan and Spanish morphosyntactic features: Catalan and Spanish rich inflectional morphology and the high frequency of some prepositional patterns allow us to include both pre- and post-nominal modifiers in the noun phrase.

1 Introduction

Ideally speaking, Corpus Linguistics should work with complex grammar formalisms providing full-parsed sentences. However, up to now, parsers cannot provide the robustness nor the speed needed in the processing of great amounts of data. Moreover, the more complex the grammar is, the more ambiguous the output becomes, since there is not enough lexical or semantic knowledge to be added to those grammars[2].

Chunkers were designed for the robust processing of big collections of texts. The robustness is achieved by reducing the complexity of the analysis and by avoiding some attachment decisions. Chunking is based on formal information, mainly about grammatical categories, since the classical definition of chunk is

> *the typical chunk consists of a single content word surrounded by a constellation of function words, matching a fixed template* [1]

As stated in [1], there is psychological evidence for *chunks*, so that this notion not only might be applied to finite state parsing, but also it might be considered from a linguistic point of view[3].

It is clear that this definition was done bearing English in mind, in which the only morphological non-verbal mark is the *-s* for nouns in plural. But, if a language has a richer inflective system, it should be possible to extend the notion of chunk. This is the case for German, and this is the case for Catalan

[1] The work presented here was partially funded by the Xtract2 project (Platform of Linguistic Engineering resources BFF2002-04226-C03-03).

[2] See [11] for a detailed discussion about the pros and cons of full parsers.

[3] See section 3 for further discussion on this point.

and Spanish too. The German inflexional system contains case, as a nominal mark. Catalan and Spanish have not case but gender and number. These two marks appear not only in the noun, but also in determiners and adjectives accompanying a given noun. So, the definition of nominal chunk can be extended to include any element showing these agreement marks. Moreover, if we take into account the frequency of some syntactic patterns, the definition of *chunk* may be enlarged with the inclusion of other postponed elements. This conception of chunking has been used for Named Entity Recognition and Classification (NERC [3]) with good results. Furthermore, we think that it may be extended to the other Romance languages, since they show similarities in the morphological features.

The paper is organised as follows: section 2 is a state-of-the-art about chunkers for Catalan and Spanish; section 3 presents our definition of *chunk*; section 4 deals with GramCat and GramEsp; finally, sections 5 and 6 give some results and the conclusions.

2 Chunkers

In this section we describe the existing parser systems for Spanish and Catalan. There are very few analysers for Spanish and even less for Catalan. In addition, most of the existing systems are not freely available, like those developed by the UAM, by Connexor or by Xerox.

[12] describes a chunker for Spanish, built at the Universidad Autónoma de Madrid (UAM), that recognises basic NP, ADJP, VP, ADVP and PP. These phrases can be recursively identified, so that nesting of phrases is allowed up to a given maximum level. *Machinese Syntax for Spanish*, by Connexor, is based on the Constraint Grammar Formalism. According to the information in the website[4] it *is a syntactic parser which produces part-of-speech classes, inflectional tags, noun phrase markers and syntactic dependencies. Syntactic dependencies show functional relations between words and phrases in sentences.* Finally, Xerox company has set up a parser for Spanish [9], based on incremental finite-state parsing. The output of the system includes a first step of chunking and a further one of extraction of syntactic dependencies[5].

Freely available parsers are CATCG, the system developed at the National Polytechnic Institute in Mexico and TACAT. CATCG[6], by the Pompeu Fabra University in Barcelona, is a syntactic analyser for Catalan that uses the Constraint Grammar Formalism. In Mexico, a chart-based Spanish Parser has been developed [10] which uses a context free grammar with some

[4] See http://www.connexor.com/demos/syntax_es.html
[5] See http://www.xrce.xerox.com/competencies/content-analysis/robustparsing/home.en.html
[6] See http://mutis.upf.es/glicom/catcg/descript_es.htm

unification elements. Finally, TACAT[7] [4], the system we use, is a chart-based syntactic analyser that uses a bottom-up and left-right strategy to build up syntactic trees. As chart parsers produce a huge amount of trees, it uses some clues to choose the best tree: apart from conditions given by the grammar, its strategy here is a top-down one (it chooses the larger and the deeper tree). The grammar it uses is a context free grammar manually written and described in section 4. The parser is language-independent, while the grammars it uses are language-dependent (see [7]), so we have developed two grammars: one for Catalan and another for Spanish.

TACAT offers some advantages that we would like to point out: on the one hand, it is a well-documented system, both in what concerns the parser itself and the grammars it uses; on the other hand, it is freely available.

3 Chunks for Catalan and Spanish

The definition of chunk and the characteristics of the formalism are the same for Catalan and Spanish; therefore, we will only mention the difference, in this and the following sections, if it is relevant for the explanation.

Abney, in [1], defines chunks in terms of *major heads*: *major heads are all content words except those that appear between a function word f and the content word that f selects. For example,* proud *is a major head in* a man proud of his son*, but* proud *is not a major head in* the proud man*, because it appears between the function word* the *and the content word* man *selected by* the. According to this definition, *a proud man* would be one chunk, while *a man proud of his son* would consists of three chunks: *[a man] [proud] [of his son].*

In addition to Abney's first definition of chunk (see section 1), there is another one, given by the same author five years later: *islands of certainty* [2] via patterns' reliability. *Islands of certainty* being more general, it seems more appropriate to us, since it is a conceptual, general definition that can be applied in different ways for different languages (i.e. different patterns may be defined for different languages), because nothing is said about the concrete elements a chunk must contain. In fact, *islands of certainty* definition may apply to the definition of German chunk proposed in [11]:

> *a chunk is a continuous part of intra-clausal constituent including recursion and pre-head as well as post-head modifiers but not pp-attachment, or sentential elements*

But it applies too for our definition of chunk for Catalan and Spanish:

> *a chunk is a continuous part of intra-clausal constituent including pre-head as well as post-head modifiers, some restricted recursivity by means of pp-attachment, but without sentential elements*

[7] See http://www.lsi.upc.es/~ nlp/

So, the differences between our definition of chunk and first Abney's [1] are:

(i) that our chunks include post-head modifiers in the nominal chunk;
(ii) that there are (limited) recursive chunks, and
(iii) that we include prepositional phrases as post-head modifiers.

The difference between our chunks and the German ones is that we include some kind of PP-attachment.

4 GramCat and GramEsp

In this section we firstly describe the rule formalism and then we focus on the nominal chunk[8]. Both GramCat and GramEsp analyse prepositional phrases, complex verbal forms, adverbial and adjectival phrases. They also recognise starting elements of clauses. However, we will concentrate on the nominal chunk, as it is the larger one and it allows us to identify nominal chunks for NERC.

The parser, TACAT, takes the text with *pos* annotation as input and produces the bracketing of the text as well as the labelling of the constituents[9].

4.1 Rule formalism

TACAT works with a context free grammar. The rule formalism is as follows: an element in the left of the rule is rewritten as zero, one or more elements on the right[10]. Elements on the left are only non-terminal nodes, while elements on the right are both terminals (*pos*-tags) and non-terminals. The following rule, for instance, rewrites a nominal chunk (*sn*) as a masculine singular determiner (*espec-ms*) and a masculine singular nominal group (*grup-nom-ms*):

sn ==> espec-ms, grup-nom-ms.

There is the possibility to specify, in the right part of the rule, one given element. This specification may be done in two different ways: by writing either the *pos*-tag or a concrete word. Somehow, this is one way to make the grammar context dependent, since in these cases the rule only applies if the given *pos*-tag or word appears. An example of such specification appears in the following rule:

[8] All the examples in section 4 are given in Catalan.

[9] Theoretical support for the building of the two grammars was mainly obtained from two big works, each one of three volumes, published in recent years: [14] for Catalan and [5] for Spanish.

[10] The existing possibility of rewriting any element as a null one has not been used in any of our grammars, since it produced a lot of ambiguity and slowed down the process.

sn ==> vmip3s0(fa), sn(mesos).

in which it is said that a nominal chunk (*sn*) is rewritten as a concrete verbal form (*fa*) and a nominal chunk containing the word *mesos*[11]. This rule will be able to analyse different structures, like *fa mesos, fa vuit mesos, fa alguns mesos*[12].

As mentioned in section 2, TACAT allows some control of the output parser in the grammar. This is done by adding some lists at the end of the grammar: the *flat*-list was thought for recursive elements: the node appears only once in the output tree; the *prior*-list establishes some priorities in the analysis; finally, the *hidden*-list includes the tree nodes that do not appear in the output (it is especially interesting to hide preterminal nodes, so that the tree structure is not too hard to read but there is no loss of information). The effects of the *hidden*-list can be seen in the next two examples: in the first one, the list has been applied, while it has not in the second (where preterminal nodes such as *j-fs* or *w-fs*, which do not appear in the first analysis, can be seen)[13].

```
sn_[ espec-mp_[ els_da0mp0]
     grup-nom-mp_[ documents_ncmp000
           sp-de_[ prep_[ de_sps00]
                   sn_[ espec-fs_[ la_da0fs0]
                        grup-nom-fs_[ Generalitat_np00000
                              s-a-fs_[ republicana_aq0fs0]]]]]]
sa_[ s-a-mp_[ dipositats_aq0mpp]]
grup-sp_[ prep_[ a_sps00 ]
          sn_[ espec-fs_[ l'_da0cs0]
               grup-nom-fs_[ Arxiu_de_Salamanca_np00000]]]
----------------------------------------
[sn_[espec-mp_[j-mp_[els_da0mp0]]
     grup-nom-mp_[n-mp_[documents_ncmp000]
             sp-de_[prep_[de_sps00]
                    sn_[espec-fs_[j-fs_[la_da0fs0]]
                        grup-nom-fs_[w-fs_[Generalitat_np00000]
                              s-a-fs_[a-fs_[republicana_aq0fs0]]]]]]]
sa_[s-a-mp_[a-mp_[dipositats_aq0mpp]]]
grup-sp_[prep_[a_sps00]
         sn_[espec-fs_[j-fs_[l'_da0cs0]]
                  grup-nom-fs_[w-fs_[Arxiu_de_Salamanca_np00000]]]]
```

[11] Chunks like this one (*fa mesos*) are the Catalan way of saying *months ago.*

[12] *Months ago; eight months ago; some months ago.*

[13] The chunked fragment is *els documents de la Generalitat republicana dipositats a l'Arxiu_de_Salamanca*, that is *the Republican Catalan Government documents stored in the Archives of Salamanca.*

4.2 The nominal chunk

The nominal chunk has the following structure:

$$[(Determiner(s)) + (AdjP) + head + (AdjP|noun|PP_{de})]$$

There is an optional determiner at the beginning of the chunk; this determiner can have a complex form, i.e. in Catalan and Spanish there may appear zero, one, two or even three determiners. The second optional element before the head is an adjectival phrase that may include only one adjective or be a complex phrase, containing, for instance, two or more coordinated adjectives, or an adjective with an adverbial complement. After the head, only one of the three mentioned elements is allowed: an adjectival phrase, a noun (no matter whether it is a proper or a common noun) or a prepositional phrase introduced by the preposition *de*[14].

In the elements contained within the nominal chunk (*sn*), morphological features related to gender and number are taken into account, in order to ensure that chunks are correctly formed; i.e. in order to avoid the combination, for instance, of a masculine singular determiner with a feminine plural noun. However, as this information is not relevant for the nominal chunk, its tag does not contain such information.

Determiners. There are 108 rules for Catalan determiners and 115 for the Spanish ones. The difference is due to the different combinations of determiners allowed in each of the languages. The combinations were established taking into account not only the theory but also real cases found in corpora.

Head. Allowed elements in the head position of the nominal chunk are: common and proper nouns, pronouns, and two or more coordinated nouns. A case of proper noun, taken from the previous example, is *Generalitat*[15], with the *pos*-tag for proper nouns *np00000*:

```
sn_[espec-fs_[la_da0fs0]
    grup-nom-fs_[Generalitat_np00000
                 s-a-fs_[republicana_aq0fs0]]]
```

Pronouns (*pos*-tags beginning by *P*) also appear in this position: they are not only personal pronouns, but also demonstratives, possessives, etc., like in the following sentence fragment: *això ho permet*[16]

[14] *De* may take in Catalan another form (*d'*) and its meaning is, both for Catalan and Spanish, *of, from*.

[15] The name of the Catalan Government.

[16] Literally: *This [clitic] allows*;
Translation *this allows that*.

```
sn_[grup-nom-ms_[aixo pd0ns000]]
sn_[grup-nom-ms_[ho pp3ns000]]
grup-verb_[permet vmip3s0]]
```

Noun coordination has been dealt with in only one case: what we call *lexical coordination*, which is coordination between two (or more) isolated nouns, without any complement or determiner. Nouns can be both proper or common nouns[17]:

```
sn_[ espec-mp_[els_da0mp0]
     grup-nom-mp_[mesos_ncmp000
           sp-de_[prep_[de_sps00]
                  sn_[grup-nom-mp_[juliol_ncms000
                                   coord_[i_cc]
                                   agost_ncms000]]]]]
```

As for NERC, the identification of the head was extremely important, since in most of the cases the head establishes the kind of NE, so the classification could be done while parsing the text.

Head Complements. We defined three kinds of head complements: an adjectival phrase, a noun, and some prepositional phrases, even though only one element can precede the head, the first one.

Nouns being noun complements are those in apposition. Some examples are *raigs gamma; etapa reina; gas mostassa*[18]. As in Catalan and Spanish the complements appear almost always in a post-head position, for the above mentioned cases the first noun is the head and the second, the complement; moreover, the noun giving its morphological features to the whole phrase is the first. For instance, in the last case (*gas mostassa*), *gas* is masculine singular and *mostassa* feminine singular and the whole phrase is masculine singular, and consequently the determiner takes the masculine form as well.

Adjectival phrases may contain an adjective, a coordination of isolated adjectives, an adjective with an adverbial complement, or a series of adjectives without any punctuation mark, like in the following case[19]:

```
sn_[espec-fp_[les_da0fp0]
    grup-nom-fp_[previsions_ncfp000
              s-a-fp_[economiques_aq0fp0 s-a-fp_[inicials_aq0cp0]]]]
```

Finally, one type of PP is included in the nominal chunk: the prepositional phrase headed by the preposition *de*, but only if the preposition comes immediately after the noun. This decision was taken after carrying out different

[17] *els mesos de juliol i agost*: *the months of July and August.*

[18] *Gamma rays; top stage; mustard gas.*

[19] *Les previsions econòmiques inicials*; Lit: *the predictions economic initial*; Translation: *the initial economic predictions.*

experiments in order to find out the frequency of some syntactic patterns, both for Catalan and Spanish. The experiments and the results we show here were done for Spanish[20].

1. We extracted a set of 210 sentences from the corpus LexEsp [13] containing the pattern {common noun + preposition DE} in order to study whether the prepositional phrase depended on the previous noun or not. The results were as follows:
   - Total amount examples of the sequence: 237.
   - Number of PPs depending on the previous noun: 230 (97%).
   - Number of PPs with ambiguous attachment: 3 (1.26%).
   - Number of PPs modifying another element: 4 (1.68%).
2. We did the same for cases where there was an adjective between the noun and the preposition. The objective was the same, but the results were completely different, since in the 60% of the cases the PP was depending on the adjective and only the 40% left were prepositional phrases depending on the noun.
3. Finally, we decided to study what happened with other prepositions. The results were that in the 80% of the cases the prepositional phrases did not depend on the immedialety previous noun but on another element on the left.

All things considered, we decided to include only the prepositional phrases headed by the preposition *de* as a noun modifier. There will obviously be some errors in the attachment (at least 1.68%) but we thought that we could afford this error if, in contrast, we had more than 97% of suitable attachments.

5 Results

As there is not any corpus analysed in terms of chunks, we could not carry out any evaluation of the grammar. However, as the grammar was used to provide us with a first analysis for building the treebanks Cat3LB [8] and Cast3LB [6] we can present the most frequent errors. Firstly, there were some errors in noun coordination. Secondly, there were few errors in the prepositional phrase headed by *de*. And finally, there were some errors related to the inclusion in the noun phrase of a postponed adjective. In fact, this construction is sometimes wrongly built, even if there are no means to previously know that: these are the cases like *consideren la Maria intel·ligent*[21], in which the adjective should be taken apart from the noun, as it is a verbal complement.

[20] They were done for Catalan too with very similar results.
[21] Lit: *They consider Mary intelligent*; *They consider Mary to be intelligent.*

6 Conclusions

We have presented in this article two grammars for chunking. The main contribution has been, however, another definition of chunk, that applies for Catalan and Spanish but could also be applied for other Romance languages.

This chunking has been used for Named Entity Recognition and Classification with very good results and could also be used for terminology extraction.

References

1. Abney, S.: Parsing by Chunks. Principle-Based Parsing (1991)
2. Abney, S.: Partial Parsing via Finite-State Cascades. Proceedings of the ESSLI'96 Robust Parsing Workshop (1996)
3. Arévalo, M., Civit, M., Martí, M.A: MICE: A Module for Named Entity Recognition and Classification. International Journal of Corpus Linguistics Volume 9, Number 1, (2004). John Benjamins
4. Atserias, J., Rodríguez, H.: TACAT: TAgged Corpus Text Analyser. Technical Report, Software Department, UPC (1998)
5. Bosque, I. Demonte, V.: Gramática Descriptiva de la Lengua Española. Espasa-Calpe (1999)
6. Civit, M., Martí, M.A.: Design Principles for a Spanish Treebank. Proceedings of the First Workshop on Treebanks and Linguistics Theories (TLT2002). Sozopol, Bulgaria (2002), 61–77
7. Civit, M.: Criterios de etiquetación y desambiguación morfosintáctica de corpus en español. Sociedad Española para el Procesamiento del Lenguaje Natural. Colección monografías. 3. (2003)
8. Civit, M., Bufí, N., Valverde, M.P.: CAT3LB: a Treebank for Catalan with Word Sense Annotation. 3rd Workshop on Treebanks and Linguistic Theories. Tuebingen, Germany (2004)
9. Gala, N.: Using the Incremental Finite State Architecture to create a Spanish Shallow Parser. SEPLN, Proceedings of the 15th Conference of the SEPLN Lleida (1999), 75–82
10. Gelbukh, A., Sidorov, G. Galicia-Haro, S., Bolsharov, I.: Environment for Development of a Natural Language Syntactic Analyzer. Acta Academia (2002)
11. Kermes, H., Evert, S.: Text analysis meets corpus linguistics. Corpus Linguistics (2003) 402–411
12. Moreno, A., Grishman, R., López, S., Sánchez, F., Sekine, S.: A Treebank of Spanish and its Application to Parsing. Procedings of the Second Conference on Language Resources and Evaluation (LREC) (2000) 107–111
13. Sebastián, N., Martí, M.A., Carreiras, M.F., Cuetos, F.: LEXESP: Léxico Informatizado del Español. Edicions de la Universitat de Barcelona, (2000)
14. Solà, J., Lloret, M.R., Mascaró, J., Pérez, M.: Gramàtica del català contemporani. Empúries, (2002)

Dynamic Perfect Hashing with Finite-State Automata

Jan Daciuk[1], Denis Maurel[2], and Agata Savary[2]

[1] Gdańsk University of Technology, jandac@eti.pg.gda.pl. Invited professor at the University of Tours, October through November 2004
[2] LI-University of Tours, France, (denis.maurel+agata.savary)@univ-tours.fr

Abstract. Minimal perfect hashing provides a mapping between a set of n unique words and n consecutive numbers. When implemented with minimal finite-state automata, the mapping is determined only by the (usually alphabetical) order of words in the set. Addition of new words would change the order of words already in the language of the automaton, changing the whole mapping, and making it useless in many domains. Therefore, we call it static. Dynamic minimal perfect hashing assigns consecutive numbers to consecutive words as they are added to the language of the automaton. Dynamic perfect hashing is important in many domains, including text retrieval and databases. We investigate three methods for its implementation.

1 Introduction

Finite-state machines are widely used in computational morphology for representing natural language dictionaries ([3], [5], [2]). However, their space efficiency decreases in applications where a different output has to be associated to each entry via the finite-state dictionary access (e.g. creation of an inverted index file). In such cases, the required space of a finite-state machine approaches that of a tree. A possible solution of this problem is compression [6].

Another solution seems to be perfect hashing, implementing a function from a word to a number from the range 0 to $n-1$, where n is the number of words. It has been proposed independently by [3] and [5]. However, the number associated with a word is not arbitrary; it is determined by the position of the word among other words stored in the dictionary. Adding a new word can change numbers associated with all other words. This may be acceptable if the data structures that use those numbers are rebuilt every time a new word is added. In text retrieval, e.g. in a search engine for a newspaper, we maintain a vocabulary with links to inverted files. New words are found continually, but we do not want those links to change with time. In databases, we can replace family names in records with corresponding numbers. As there can be many people with the same family name, this saves the space. If the database is that of living people, new family names will continually appear. However, we still want the same numbers refer to the same names.

We propose three dynamic hashing methods to solve this problem. The first one uses perfect hashing while allowing the perfect hashing function to change inside the automaton. We then use an additional data structure, a translation vector, to compensate for the change. In another solution, we include weights introduced by perfect hashing into the labels, i.e. into the right language of states. As the right language is used for determining equivalence of states, we obtain an automaton that is not minimal. The third solution is also based on an automaton that is not minimal – it is pseudo-minimal.

2 Mathematical Preliminaries

A deterministic finite-state automaton is a quintuple $M = (Q, \Sigma, \delta, q_0, F)$, where Q is a finite set of states, Σ is a finite set of symbols called the *alphabet*, $\delta : Q \times \Sigma \to Q$ is a (partial) *transition function* (if $\delta(q, a)$ is not defined, we write $\delta(q, a) = \perp$), $q_0 \in Q$ is the *initial state*, and $F \subseteq Q$ is a set of final states. The transition function δ can be extended to $\delta^* : Q \times \Sigma^* \to Q$:

$$\forall_{q \in Q}\, \delta^*(q, \epsilon) = q$$
$$\forall_{q \in Q, a \in \Sigma, \delta(q,a) \neq \perp, w \in \Sigma^*}\, \delta^*(q, aw) = \delta^*(\delta(q, a), w)$$

The language of an automaton is a set of words (strings) spelled out on all paths leading from q_0 to any of the final states:

$$\mathcal{L}(M) = \{w \in \Sigma^* : \delta^*(q_0, w) \in F\}$$

The size of an automaton is defined as the number of its states:

$$|M| = |Q|$$

The minimal automaton M_{min} is the unique (up to isomorphisms) automaton that has the smallest size among all deterministic automata that recognize the same language:

$$\forall_{M : \mathcal{L}(M) = \mathcal{L}(M_{min})}\, |M_{min}| < |M|$$

The right language of a state q is a set of words (strings) spelled out on all paths leading from q to any of the final states:

$$\overrightarrow{\mathcal{L}}(q) = \{w \in \Sigma^* : \delta^*(q, w) \in F\}$$

Obviously, $\overrightarrow{\mathcal{L}}(q_0) = \mathcal{L}(M)$. The equality of right languages is an equivalence relation that partitions the set of states into equivalence classes. In the minimal automaton, each class has only one representative.

3 Static Perfect Hashing

The number of words in the automaton is the number of words in the right language of the initial state. For each state, we can calculate:

$$|\overrightarrow{\mathcal{L}}(q)| = (\Sigma_{a:\delta(q,a)\neq\perp} |\overrightarrow{\mathcal{L}}(\delta(q,a))|) + \begin{cases} 0 & q \notin F \\ 1 & q \in F \end{cases}$$

We can assume that the alphabet is ordered. That order induces an ordering of words[1], and an ordering of transitions of a state. In that ordering and, in the case of acyclic automata, a word $w \in \mathcal{L}(M)$ gets number $\rho(w)$ which is the number of words that precede it in the language of the automaton (supposing that words are counted from 0).

This number may be calculated from weights included either in states or in transition labels of the automaton. If the latter case the weight $\omega : Q \times \Sigma \to \mathcal{I}$ stored on a particular transition is the ordinal number of the first word recognized by traversing this transition minus the sum of weights on the preceding transitions (see Figure 1 for an example).

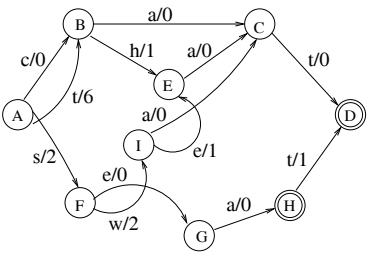


Fig. 1. Static perfect hashing on words {*cat, chat, sea, seat, swat, sweat, tat, that*} using weights included in labels. To calculate the number associated with a word, we add all weights on its transitions. For example, to calculate the number associated with *seat* we get $2 + 0 + 0 + 1 = 3$

An arbitrary word w can precede a number of words that are already in the language of an automaton. Therefore, adding such a word has to change ordinal numbers of words that follow it in lexicographical order. While adding the new word, we follow the path that recognizes the longest prefix of w that can be found in the automaton. At each state along that path, we add 1 to the weight on every transition that has a label that lexicographically follows the label on the transition we traverse (see Figure 2).

4 Perfect Hashing with Translation Vector

To counteract the changes introduced by adding new words, a *translation vector* can be used. The purpose of the vector is to translate from the num-

[1] various compression techniques may introduce different ordering

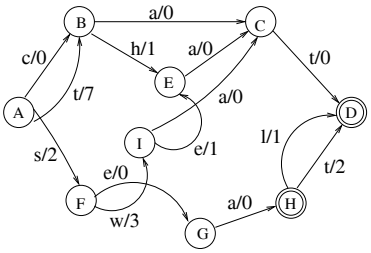


Fig. 2. Weighted automaton resulting from the addition of a new word *seal* to the language of the previous example. The new word gets number 3, the words *seat, swat, sweat, tat* and *that* are shifted to numbers 4 through 8, respectively.

bers resulting from the internal ordering of words in the automaton (in the simplest case – ordinal numbers of words in lexicographical order), to the order in which they appear in the input. To obtain the ordinal number of a word, we first look for it in the automaton and obtain its number in the minimal automaton using weights on transitions. Then that number serves as an index in the translation vector. The value held in the place indexed with the word's number in the automaton gives the ordinal number of the word.

The length of the translation vector is the number of words recognized by the minimal automaton. If a mapping from numbers to words is required, it can be implemented with a second translation vector. A translation vector for the automaton on Figure 1, implementing the order {*cat, seat, swat, chat, sea, sweat, tat, that*} is $\{0, 3, 4, 1, 2, 5, 6, 7\}$.

Suppose that we want to add a new word w to an automaton already re-cognizing n words, as described in the preceding section. Suppose that there are exactly i words belonging to the language of the automaton that lexicographically precede w. The automaton modification algorithm runs in $\mathcal{O}(|w||\Sigma|)$. Then, we need to update the translation vector. This is done by shifting values starting at position i on forward by one position. It means moving $|\mathcal{L}(M)|/2$ on average, and $|\mathcal{L}(M)|$ at most values. For example, the translation vector for the automaton on Figure 2 is $\{0, 3, 4, 8, 1, 2, 5, 6, 7\}$, where the 5 last values have been shifted after the addition of word *seal*.

5 Perfect Hashing with Weights Included in Labels

If we don't want to use a translation vector, which might be huge, it is possible to arrange the weights in transition labels in such a way that their sums on appropriate paths would give respective word numbers direcly. However, such an automaton may be bigger then the minimal automaton (note that this is not the case while using a translation vector). For example, given $\mathcal{L} = \{aa, ba, bb, ab\}$, the minimal automaton (Figure 3) has 3 states.

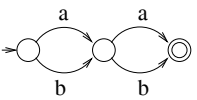


Fig. 3. Minimal automaton accepting $\{aa, ab, ba, bb\}$

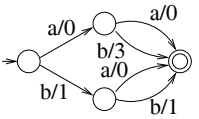


Fig. 4. Automaton with weights assigning consecutive weights (starting from 0) to words $\{aa, ba, bb, ab\}$

However, if we assign 0 to aa, 1 to ba, 2 to bb, and 3 to ab, we get two states with the right language $\{a, b\}$ that have different weights on their transitions (Figure 4). Those weights would be lost if we were to merge the states.

In order to prevent merging of states, we incorporate weights into the right language of a state. Evidently, if words come lexicographically sorted, the automaton we get has the same number of states as the minimal automaton for the given language.

Handling the new automaton is almost identical to handling an ordinary acyclic deterministic automaton. There are, however, a few modifications. To avoid problems posed by negative weights that are necessary when handling words that are prefixes of other words, we use a special end-of-word symbol that does not belong to the alphabet. The symbol is appended to each word. In the automaton that is built from such data, there is only one final state, and it has no outgoing transitions. All states that would be final are non-final, and they have an additional transition labeled with the special symbol, and leading to the final state.

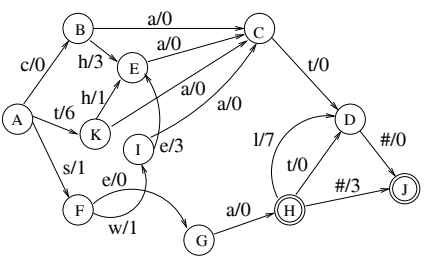


Fig. 5. Weighted automaton (with no translation vector) implementing words in the following order {*cat, seat, swat, chat, sea, sweat, tat, that, seal*}. The addition of word *seal* numbered 8 resulted in the creation and minimization of the suffix transitions *l/7* and *#/0*.

While adding a new word, we traverse the path recognizing the longest prefix of the word that can be found in the automaton, calculating the sum of the weights on transitions. Then we create a chain of states and transitions recognizing the suffix of the word. On the first transition we create, we put the difference between the word number and the sum of weights on transi-

tions recognizing the prefix. Because the words on input receive consecutive numbers, and the sum of weights on the path recognizing the prefix of the words to be added is a sum of weights on a path recognizing a prefix of a word w' already in the automaton, that sum is not greater than the number assigned to w'. Therefore, the weight to be put on the first transition is always positive. Figure 5 shows the weighted automaton with no translation vector for the language from the previous section.

Note that with this solution, it is no longer possible to easily compute the mapping from numbers to words. The weights on transitions no longer partition words into sets with word numbers staying below/above certain value.

6 Pseudo-minimal Automata

The third solution for dynamic perfect hashing is a pseudo-minimal automaton [4], i.e. an automaton having a proper element (a transition or a state) for each word. That proper element is not shared with any other word. Therefore, to achieve perfect hashing (including dynamic perfect hashing), it is sufficient to put the word number on its proper element. To avoid putting the word numbers on both transitions and states, we introduce the end-of-word symbol appended to each word.

We call a state *convergent* if it has more than one incoming transition. We call a state *divergent* if it has more than one outgoing transition. Note that these definitions are simplified versions of those found in [4]. In a pseudo-minimal automaton, there is no path on which a divergent state follows a convergent state. For example, the automaton on Figure 2 is not pseudo-minimal as state B is both convergent and divergent. Figure 6 shows a pseudo-minimal automaton ordering the same words in the same order as in the preceding section. Note that the two automata on Figures 5 and 6 are identical up to weights. Our tests described in section 8 show that the two automata are identical for languages of up to 100 words while for larger languages their size ratio is very close to 1.

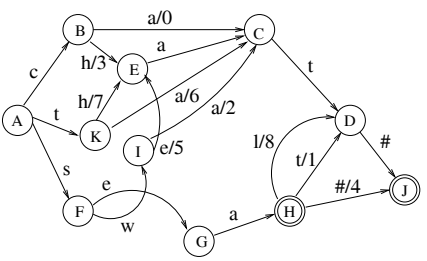


Fig. 6. Pseudo-minimal automaton implementing words in the following order {*cat, seat, swat, chat, sea, sweat, tat, that, seal*}.

It is possible to obtain a pseudo-minimal automaton from a minimal one by first marking states q for which $|\overrightarrow{\mathcal{L}}(q)| > 1$, and then cloning every convergent state marked by this procedure. Modifications for incremental construction algorithms for sorted data and unsorted data [1], and for a semi-incremental Watson's construction algorithm [7] exist that directly construct pseudo-minimal automata. Of those, only the incremental algorithm for unsorted data is of interest for dynamic hashing. Those modifications are out of scope of this paper. The Revuz's semi-incremental construction algorithm [5] also builds a pseudo-minimal automaton before minimizing it. However, that algorithm requires data to be sorted on reversed words.

Note that with this solution, it is no longer possible to easily compute the mapping from numbers to words, as the word number is stored only on the proper transition.

7 Theoretical Limits

It is interesting to find what are theoretical limits for sizes of automata with weights included in labels and pseudo-minimal automata, as compared to standard minimal automata for respective languages.

It is easy to show that the worst case for pseudo-minimal automata is that of $\mathcal{L} = \Sigma^n$. The minimal automaton for that case has $n+1$ states and $n|\Sigma|$ transitions, with $|\Sigma|$ transitions linking each consecutive pair of states. An example of such automaton is shown on Figure 3. Because prefixes of length $n-1$ are common for $|\Sigma|$ words, only the transitions recognizing the last symbol in words can be their proper elements. Because there are $|\Sigma|^n$ words, there are as many proper transitions. They all lead to a state that has a single outgoing transition, and it is labeled with the end-of-word marker. The automaton is shown on Figure 7.

The pseudo-minimal automaton has $2+\Sigma_{i=0}^{n-1}|\Sigma|^i = 2+\frac{|\Sigma|^n-1}{|\Sigma|-1}$ states, and $\Sigma_{i=0}^{n}|\Sigma|^i = \frac{|\Sigma|^{(n+1)}-1}{|\Sigma|-1}$ transitions.It is exponentially bigger than the minimal automaton. However, it is the same order of magnitude as the length of the translation vector.

It turns out that the same case is also the most difficult one for perfect hashing with weights included in labels. However, to prevent minimization, a particular order of words must be used. The automaton forms a tree identical to the one for pseudo-minimal automaton (with transitions recognizing the last letter of words converging to a single state with a single transition labeled with the end-of-word symbol). To insure that, the weights on outgoing transitions of all states reached with Σ^{n-1} should form pairwise different sets.

One way to achieve that is to put all words ending with the first symbol of the alphabet first, and then put all remaining words in reverse lexicographical order. In the resulting automaton (see Figure 8 for an example with $\Sigma =$

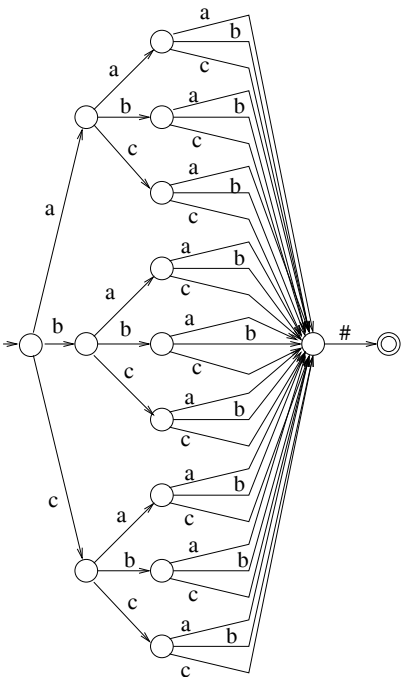


Fig. 7. Pseudo-minimal automaton recognizing the language $\mathcal{L} = \Sigma^n$

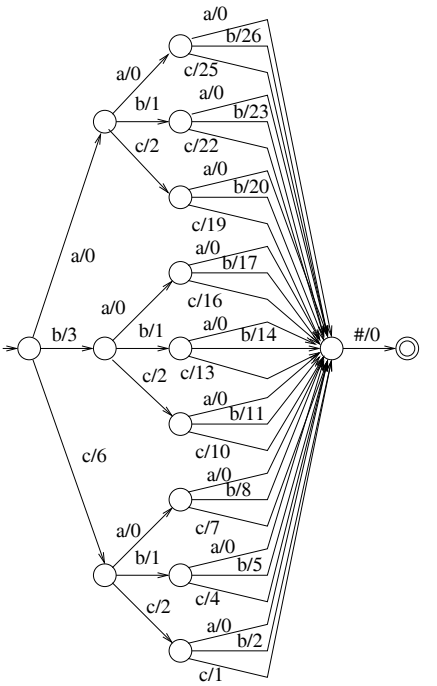


Fig. 8. The worst case for an automaton with weights included in labels

$\{a, b, c\}$ and $n = 3$), the sum of weights on transitions recognizing the prefixes Σ^{n-1} will be the ordinal numbers of those prefixes in the set (starting from 0). Let $\Sigma = \{\sigma_1, \sigma_2, \ldots \sigma_m\}$, where $m = |\Sigma|$, be an ordered set of symbols. The transition labeled with the first symbol of the alphabet σ_1 recognizing the last symbol of the word will always have weight 0. Because consecutive prefixes Σ^{n-1} will have the sum of weights equal to consecutive integers, all words $\Sigma^{n-1}\sigma_1$ ending with the first symbol of the alphabet will get consecutive numbers corresponding to their position in the input.

The remaining $m^{n-1}(m-1)$ words come in reverse order. The first of them gets number m^{n-1}, which is 1 greater than the sum of weights on the lexicographically last prefix σ_m^{n-1}. The transition recognizing the last symbol σ_m of the alphabet as the last symbol of a word that has the prefix being lexicographically the last among prefixes of length $n-1$ will get weight 1. Last transitions for $m-2$ subsequent words $\sigma_m^{n-1}\sigma_{m-1}, \sigma_m^{n-1}\sigma_{m-2}, \ldots \sigma_m^{n-1}\sigma_2$ ending with preceding symbols of the alphabet will get consecutive integers starting from 2 as their weights. The outgoing transitions for the last symbols of words $\sigma_m^{n-2}\sigma_{m-1}\sigma_m, \ldots, \sigma_m^{n-2}\sigma_{m-1}\sigma_2$ will get weights $m+1, \ldots, 2m-1$. The weights start with $m+1$ and not with m because the sum of weights on σ_m^{n-1} is 1 greater than the sum of weights on $\sigma_m^{n-2}\sigma_{m-1}$. In this way

each state reachable with different prefix Σ^{n-1} will get a different suite of transition weights. Thus the states cannot be merged, although their right languages without weights are identical.

8 Experiments

We performed experiments simulating the application of the 3 hashing methods described above to indexing of a natural language corpus. Three corpora of different sizes, languages and types were used: the English corpus was a several-thousand-word technical text[2], the Polish one was a 1.5-million-word literary corpus[3], and the French one was a 44-million-word press corpus (*Le Monde* journal's full 1998-99 edition). Each corpus was first reduced by a set of filters so that only the first occurence of each word form was retained (non-alphabetical tokens were not taken into account). Thus, the unique word forms appeared in the resulting list in a "natural" order determined by the corpus. That list was given as the input to the construction algorithms for the minimal automaton (MA), the weighted perfect-hashing automaton (WA), and the pseudo-minimal automaton (PA). The results are shown in Table 1.

		English corpus	Polish corpus	French corpus
# of words		276,268	1,500,536	44,406,805
# of unique words		11,168	146,279	373,860
Ave. word length		7.4	8.5	8.1
# states / # transitions	MA	$9,564/18,102$	$73,584/170,173$	$184,859/465,873$
	WA	$13,318/24,390$	$119,007/265,128$	$307,525/679,714$
	PA	$13,437/24,603$	$119,188/265,465$	$309,884/683,741$
	$\lvert WA\rvert/\lvert MA\rvert$	1.39/1.35	1.62/1.56	1.66/1.46
	$\lvert PA\rvert/\lvert MA\rvert$	1.40/1.36	1.62/1.56	1.68/1.47
	$\lvert PA\rvert/\lvert WA\rvert$	1.009/1.009	1.002/1.001	1.0077/1.0059

Table 1. Experimental results for various corpus types and sizes

The size of the translation vector is not taken into account in the data for the minimal automaton. The results show that WA and PA are almost identical as far as both the number of states and the number of transitions are concerned, PA being very slightly bigger than WA. A more detailed test for different prefixes of the biggest corpus showed that the three automata are identical for small corpora of up to 100 words. With a growing corpus size, the PA/MA (WA/MA) ratio grows continually up to 1.75. However, the ratio decreases slightly (down to 1.66) when the corpus size rises above 30-million words.

[2] downloaded from `www.research.ibm.com/journal` in August 2000
[3] downloaded from `monika.univ.gda.pl/~literat/books.htm` on Nov 1, 2004

Note that for morphological dictionaries the $|PA|/|MA|$ ratio may grow up to 8 [5] because each entry contains a full morphological paradigm.

9 Conclusions

Unsurprisingly, the results show that the minimal automaton (MA) is up to 50% smaller than both the weighted perfect hashing automaton (WA) and the pseudo-minimal automaton (PA), in terms of the number of states and transitions. However, the real memory size ratios of the three automata depend on the implementation details. Suppose we need 4 bytes for a translation vector cell (the length of the vector is equal to the number of unique words), 8 bytes for a state, and 16 bytes for a transition. WA is then 28% bigger than MA. The difference between WA and PA is negligible.

Note that MA, contrary to WA and PA, allows for an easy implementation of the number-to-word mapping (by doubling the translation vector size).

On the other hand, in terms of speed, the WA and PA have an advantage over MA. Updating WA and PA is only slightly faster than updating MA. However, it is the cost of updating the translation vector that counts. One has to move half of the vector on average to readjust the hashing function, while the cost of updating the automaton (whatever kind) is proportional to the length of the word being added.

References

1. Jan Daciuk, Stoyan Mihov, Bruce Watson, and Richard Watson. Incremental construction of minimal acyclic finite state automata. *Computational Linguistics*, 26(1):3–16, April 2000.
2. Lauri Karttunen. Constructing lexical transducers. In *In Proc. of the 15th International Conference on Computational Linguistics, COLING'94*, Kyoto,Japan, 1994.
3. Claudio Lucchiesi and Tomasz Kowaltowski. Applications of finite automata representing large vocabularies. *Software Practice and Experience*, 23(1):15–30, Jan. 1993.
4. Denis Maurel. Pseudo-minimal transducer. *Theoretical Computer Science*, (231):129–139, 2000.
5. Dominique Revuz. *Dictionnaires et lexiques: méthodes et algorithmes*. PhD thesis, Institut Blaise Pascal, Paris, France, 1991. LITP 91.44.
6. Strahil Ristov and Eric Laporte. Ziv Lempel compression of huge natural language data tries using suffix arrays. *Journal of Discrete Algorithms*, pages 241–256, 1999.
7. Bruce Watson. A fast new (semi-incremental) algorithm for the construction of minimal acyclic DFAs. In *Third Workshop on Implementing Automata*, pages 91–98, Rouen, France, September 1998. Lecture Notes in Computer Science.

Dictionary-Based Part-of-Speech Tagging of Polish

Stanisław Galus

Wyższa Szkoła Finansów i Administracji, ul. Władysława IV 22, Sopot, Poland

Abstract. A set of 70 lexical symbols is defined which covers written Polish. Under the assumption of the second order Markov process, a dictionary-based method of tagging parts of speech is presented. Instead of having to be trained on earlier tagged text, parameters of the method are estimated on frequencies of unambiguously classifiable unigrams, bigrams and trigrams of lexical symbols found in untagged literary text. The method is being proved to tag correctly 88% of all words in text.

1 Introduction

Part-of-speech tagging is an act of assigning part of speech to each word in a text. It is one of the tasks of natural language processing.

There are two major applications of automatic tagging. It can be used in an early phase of parsing, that is stating the grammatical functions of all the words in a sentence. It can also be used as a research tool for searching texts for sequences of words of given parts of speech. Automatic tagging has numerous practical applications as well.

Models being used to formulate algorithms of tagging may be classified as probabilistic and non-probabilistic.[1] The most extensively used probabilistic models, namely Markov models and hidden Markov models, are based on Markov processes. The most common non-probabilistic approach is transformation-based tagging.

To estimate parameters of a model, supervised and unsupervised estimation can be used. Supervised estimation requires the correct tagging of a large sample of text, while unsupervised learning does not. A kind of unsupervised estimation is based on a dictionary providing information on parts of speech admissible for each word.

Accuracy of tagging is easy to evaluate. A common measure of accuracy is the fraction of correctly tagged words. The percentage of correctly tagged ambiguous words is also used.

The most simple Markov models are n-gram models which assume that the n-th word depends only on the previous $n-1$ words. We shall be proving that **with probability 0.99, the accuracy of dictionary-based trigram part-of-speech tagging of Polish exceeds 88%**.

[1] See [4, chapter 10].

As far as we know, there has already been an attempt to build an automatic part-of-speech tagger of Polish. Namely, a trigram morphosyntactic tagger presented in [2], disambiguates a set of tags including inflexional categories. Trained on an annotated corpus, it achieves accuracy of 90.6%. Several attempts have also been made to build a lexical analyser, for example LEXAN, described in [10, p. 50–56][2].

2 Methods

Let Σ be a finite alphabet. A language L is a certain set of finite strings over the alphabet Σ. A text over the language L is a finite sequence of strings from L. The set of all texts over the language L will be denoted by L^*.

Let us assume that the language L is a sum of $m > 1$ classes of lexical symbols $L_1, L_2, \ldots, L_m$. Given a text

$$\boldsymbol{a} = (a_i)_{i=1}^n \in L^*, \quad n \geq 1, \tag{1}$$

we define its tagging as any sequence $\boldsymbol{t} = (t_i)_{i=1}^n$ such that $t_i \in \{1, 2, \ldots, m\}$ and $a_i \in L_{t_i}$ for $i = 1, 2, \ldots, n$. We assume that among all taggings of the text $\boldsymbol{a}$, there exists exactly one tagging $\boldsymbol{t}$ which may be called a true tagging. In context of tagging, the numbers $1, 2, \ldots, m$ are called tags.

2.1 The alphabet and lexical symbols

Written Polish language uses letters, punctuation marks, a hyphen, digits and graphical signs. The Polish alphabet consists of twenty four letters of the Latin alphabet, two non-Latin letters and nine letters of the Latin alphabet with diacritical marks: A, Ą, B, C, Ć, D, E, Ę, F, G, H, I, J, K, L, Ł, M, N, Ń, O, Ó, P, Q, R, S, Ś, T, U, V, W, X, Y, Z, Ź, Ż, a, ą, b, c, ć, d, e, ę, f, g, h, i, j, k, l, ł, m, n, ń, o, ó, p, q, r, s, ś, t, u, v, w, x, y, z, ź, ż. Since numerous foreign proper names and borrowings are written without a transcription, the set of letters is often extended. There are ten punctuation marks: ., ;, „ :, –, . . . , ?, !, brackets and quotation marks. [5, p. 1656] names five kinds of brackets: ( ), [ ], / /, { }, < > and three kinds of quotation mark: „ ”, ≪ ≫, ‘ ’. Typewriter quotation mark: " " may also sometimes occur [7, p. CXXII]. The hyphen: - is used for spelling compounds and dividing words at ends of lines. Ten digits are: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Graphic signs include basic mathematical symbols, section sign and computer symbols: =, +, %, §, #, $, &, *, @, \, ˆ, _, |, ˜.

We distinguish six classes of lexical symbols: words, punctuation marks, hyphen, numbers, graphical signs and unknown characters. These classes may be further subdivided. The most detailed division used in this paper, possessing $m = 70$ classes, is presented in table 1.

[2] For a more comprehensive description of morphological analysers for Polish see [3].

Table 1. The list of lexical symbols

Parts of speech.		Punctuation marks.		Hyphen, number, graphical signs, unknown character.	
Class	Symbol	Class	Symbol	Class	Symbol
L_1	unknown word	L_{30}	.	L_{54}	-
L_2	unknown part of speech	L_{31}	;	L_{55}	number
L_3	noun	L_{32}	,	L_{56}	=
L_4	noun, masculine-personal	L_{33}	:	L_{57}	+
L_5	noun, masculine-animal	L_{34}	–	L_{58}	%
L_6	noun, masculine-inanimate	L_{35}	...	L_{59}	§
L_7	noun, feminine	L_{36}	?	L_{60}	#
L_8	noun, neuter	L_{37}	!	L_{61}	$
L_9	adjective	L_{38}	(	L_{62}	&
L_{10}	adjective, positive	L_{39}	)	L_{63}	*
L_{11}	adjective, comparative	L_{40}	[	L_{64}	@
L_{12}	adjective, superlative	L_{41}	]	L_{65}	\
L_{13}	pronoun	L_{42}	/	L_{66}	ˆ
L_{14}	numeral	L_{43}	{	L_{67}	_
L_{15}	verb	L_{44}	}	L_{68}	|
L_{16}	imperfect verb	L_{45}	<	L_{69}	˜
L_{17}	perfect verb	L_{46}	>	L_{70}	unknown
L_{18}	adjectival passive participle	L_{47}	„		character
L_{19}	adjectival active participle	L_{48}	”		
L_{20}	verbal noun	L_{49}	≪		
L_{21}	adjectival past participle	L_{50}	≫		
L_{22}	adverb	L_{51}	‘		
L_{23}	adverb, positive	L_{52}	’		
L_{24}	adverb, comparative	L_{53}	"		
L_{25}	adverb, superlative				
L_{26}	preposition				
L_{27}	conjunction				
L_{28}	particle				
L_{29}	interjection				

The class of words, which may be otherwise defined as non-empty finite sequences of letters, is equal to the sum $L_1 \cup L_2 \cup \ldots \cup L_{29}$. The sum $L_{30} \cup L_{31} \cup \ldots \cup L_{53}$ forms the class of punctuation marks. The class of numbers L_{55} consists of non-empty finite sequences of digits. The sum $L_{56} \cup L_{57} \cup \ldots \cup L_{69}$ forms the class of graphical signs.

Because of linguistic complexity of the matter, several classifications of parts of speech have been presented.[3] The list shown in table 1, although arbitrary, seems to be sufficient for assigning parts of speech. It extends traditional school grammar which discriminates ten parts of speech:[4] L_3, L_9, L_{13}, L_{14}, L_{15}, L_{22}, L_{26}, L_{27}, L_{28}, L_{29}. In order to make it possible to assign the most basic selective categories, that is gender of a noun, degree of an adjective, aspect of a verb and degree of an adverb, the next thirteen classes have been introduced: L_4, L_5, L_6, L_7, L_8, L_{10}, L_{11}, L_{12}, L_{16}, L_{17}, L_{23}, L_{24}, L_{25}. To avoid difficulties of classification, we arbitrary put $L_9 = L_{10} \cup L_{11} \cup L_{12}$, $L_{15} = L_{16} \cup L_{17}$ and $L_{22} = L_{23} \cup L_{24} \cup L_{25}$.[5] As there are many genderless nouns, $L_3 \setminus (L_4 \cup L_5 \cup L_6 \cup L_7 \cup L_8) \neq \emptyset$. The most controversial issue is separation of four declensional forms of a verb: L_{18}, L_{19}, L_{20}, L_{21} from conjugation.[6] The remaining two classes have only technical importance: a correct word which cannot be clearly assigned to any part of speech should be considered as an unknown part of speech L_2, while words from outside the dictionary should be counted as unknown words L_1.

2.2 The model

Let us assume that the text $\boldsymbol{a}$ defined by (1) is generated in such a way that its tagging $T = (T_1, T_2, \ldots, T_n)$ follows the second order Markov process

$$\begin{aligned} &\Pr(T_1 = t_1) && = p^1_{t_1}, \\ &\Pr(T_2 = t_2 \,|\, T_1 = t_1) && = p^2_{t_1 t_2}, \\ &\Pr(T_i = t_i \,|\, T_{i-2} = t_{i-2}, T_{i-1} = t_{i-1}) && = p^3_{t_{i-2} t_{i-1} t_i}, \quad i = 3, 4, \ldots, n, \end{aligned} \tag{2}$$

where the probabilities

$$[p^1_{t_1}]_{t_1=1,2,\ldots,m},\ [p^2_{t_1 t_2}]_{t_1,t_2=1,2,\ldots,m},\ [p^3_{t_1 t_2 t_3}]_{t_1,t_2,t_3=1,2,\ldots,m}$$

are given beforehand. The model can be realized by drawing a sequence of tags $t_1, t_2, \ldots, t_n$ according to (2), followed by drawing elements $a_i \in L_{t_i}$, $i = 1, 2, \ldots, n$, assuming equal probabilities inside each class.

[3] See [10, p. 53–56] and [11] for examples.

[4] See [6, p. 101].

[5] The reason for existence of L_9, L_{15} and L_{22} is future development of the dictionary. For example, even if it were well-founded, it would be impossible to move incomparable adjectives from L_{10} to L_9 at this moment in time.

[6] See [9, chapter III.6], as well as [6, p. 112–115] and [9, chapter III.1] for discussions.

Let us define a dictionary function $L \ni a \to d(a) \subset \{1, 2, \ldots, m\}$, such that $t \in d(a)$ if and only if $a \in L_t$. Given the text (1), the likelihood function of the sequence $\boldsymbol{t} = (t_i)_{i=1}^n$ is defined as[7]

$$L(\boldsymbol{t}; \boldsymbol{a}) = p^1_{t_1} \cdot p^2_{t_1 t_2} \cdot \prod_{i=3}^{n} p^3_{t_{i-2} t_{i-1} t_i} \tag{3}$$

for those values of $\boldsymbol{t}$ which belong to the set

$$d(a_1) \times d(a_2) \times \ldots \times d(a_n). \tag{4}$$

A sequence $\hat{\boldsymbol{t}} = (\hat{t}_1, \hat{t}_2, \ldots, \hat{t}_n)$ is a maximum likelihood estimator of the true sequence $\boldsymbol{t}$ if

$$L(\hat{\boldsymbol{t}}; \boldsymbol{a}) = \max_{\boldsymbol{t}} L(\boldsymbol{t}; \boldsymbol{a}), \tag{5}$$

where $\boldsymbol{t}$ runs over (4).

Let us make two obvious remarks that much simplify the maximization (5). Firstly, if $d(a_j) = \{t_j\}$ for a certain $1 \le j \le n$, then $\hat{t}_j = t_j$. Secondly, every two consecutive unambiguously classifiable lexical symbols may be used to factorize the maximization (5). Namely, let $(i_j)_{j=1}^r$, $r \ge 1$, be the sequence of all indexes of the text (1) such that $i_{j+1} \ge i_j + 2$ for $j = 1, 2, \ldots, r-1$ and $|d(a_{i_j})| = |d(a_{i_j+1})| = 1$ for $j = 1, 2, \ldots, r$. The sequence $1, 2, \ldots, n$ may be divided into r or $r+1$ (according as $i_r + 1 = n$ or $i_r + 1 < n$) segments:

$$\begin{aligned} S_1 &= \{1, 2, \ldots, i_1, i_1 + 1\}, \\ S_j &= \{i_{j-1} + 2, i_{j-1} + 3, \ldots, i_j, i_j + 1\}, \quad j = 2, 3, \ldots, r, \\ S_{r+1} &= \{i_r + 2, i_r + 3, \ldots, n-1, n\}. \end{aligned}$$

Owing to independence induced by uniqueness, the maximization (5) may be carried out on each segment separately:

$$\max_{\boldsymbol{t}} L(\boldsymbol{t}; \boldsymbol{a}) = Q_1 Q_2 \ldots Q_r Q_{r+1}, \tag{6}$$

where

$$Q_1 = \max \left\{ p^1_{t_1} p^2_{t_1 t_2} \prod_{k=3}^{i_1+1} p^3_{t_{k-2} t_{k-1} t_k} : t_s \in d(a_s), s \in S_1 \right\},$$

$$Q_j = \max \left\{ \prod_{k \in S_j} p^3_{t_{k-2} t_{k-1} t_k} : t_s \in d(a_s), s \in S_j \right\}, \quad j = 2, 3, \ldots, r, r+1.$$

[7] In contrast to the equation (10.7) of [4, section 10.2] for determining the optimal tags, the formula (3) does not contain the terms corresponding to $\Pr(w_i \,|\, t_i)$. This is due to limited information provided by the dictionary d: it can only answer the question whether or not $\Pr(w_i \,|\, t_i) > 0$.

High relative frequency of unambiguously classifiable lexical symbols breaks text up into many short runs of symbols, making the process of maximization fast.

In other words, the algorithm of tagging is such that if a word cannot be found in the dictionary, it is classified to the class L_1. Otherwise, if the word belongs to exactly one of the classes L_2, L_3, ..., L_{29}, it is classified into this class. If the word belongs to more than one of the classes, the maximization (5) is performed on a sequence of surrounding words.

For example, when tagging the text defined by

$$\begin{array}{cccccccc} a_1 = & a_2 = & & a_3 = & a_4 = & & a_5 = & a_6 = \\ \textit{Paweł} & \textit{mieszka} & \textit{w} & & \textit{małym} & \textit{mieście} & . & \end{array}$$

we have $d(a_1) = \{4\}$, $d(a_3) = \{26\}$, $d(a_4) = \{10\}$, $d(a_6) = \{30\}$. The remaining words are ambiguous: $d(a_2) = \{6, 16\}$ (as *mieszek* or *mieszkać*), $d(a_5) = \{16, 8\}$ (as *miesić* or *miasto*). The four possible taggings have the following likelihoods (3):

$$\begin{array}{l} p_4^1 \cdot p_{4,6}^2 \cdot p_{4,6,26}^3 \cdot p_{6,26,10}^3 \cdot p_{26,10,16}^3 \cdot p_{10,16,30}^3, \\ p_4^1 \cdot p_{4,16}^2 \cdot p_{4,16,26}^3 \cdot p_{16,26,10}^3 \cdot p_{26,10,16}^3 \cdot p_{10,16,30}^3, \\ p_4^1 \cdot p_{4,6}^2 \cdot p_{4,6,26}^3 \cdot p_{6,26,10}^3 \cdot p_{26,10,8}^3 \cdot p_{10,8,30}^3, \\ p_4^1 \cdot p_{4,16}^2 \cdot p_{4,16,26}^3 \cdot p_{16,26,10}^3 \cdot p_{26,10,8}^3 \cdot p_{10,8,30}^3, \end{array}$$

with

$$\begin{aligned} Q_1 &= \max\left\{p_4^1 p_{4,6}^2 p_{4,6,26}^3 p_{6,26,10}^3,\ p_4^1 p_{4,16}^2 p_{4,16,26}^3 p_{16,26,10}^3\right\}, \\ Q_2 &= \max\left\{p_{26,10,16}^3 p_{10,16,30}^3,\ p_{26,10,8}^3 p_{10,8,30}^3, \right\}. \end{aligned}$$

In the absence of tagged text, the point of estimating the probabilities $p_{t_1 t_2 \ldots t_k}^k$ can be based on frequencies of classes of unambiguously classifiable k-grams of lexical symbols. Namely, let $f_{t_1 t_2 \ldots t_k}^k$ denote the frequency of such k-grams in a text (1), that is

$$f_{t_1 t_2 \ldots t_k}^k = |\{k \le i \le n : (\forall j = 0, 1, \ldots, k-1)\ d(a_{i-j}) = \{t_{k-j}\}\}|.$$

The conditional probability $p_{t_1 t_2 \ldots t_k}^k$ may be estimated according to Laplace's law[8] [4, section 6.2.2]:

$$\hat{p}_{t_1 t_2 \ldots t_k}^k = \frac{1 + f_{t_1 t_2 \ldots t_k}^k}{m + \sum_{t_k=1}^m f_{t_1 t_2 \ldots t_k}^k}. \tag{7}$$

[8] The original factor behind the Laplace law was to avoid too frequent zero-valued likelihoods (3) which might appear if the maximum likelihood estimator were used. No sensitivity analysis of results in respect of the estimator of $p_{t_1 t_2 \ldots t_k}^k$ has been carried out.

2.3 Data

There are three separate sets of lexical data used to compile dictionary, calculate estimates (7) and control tagging accuracy.

The main source of words in the dictionary is the dictionary [8]. The source of proper names is a list of geographical and ethnical names from [5, p. 1391–1473] and a list of surnames and names from [5, p. 1475–1554]. The set of words for the program Ispell[9], compiled by Piotr Gackiewicz, Włodzimierz Macewicz and Mirosław Prywata, was helpful in an early stage of preparing the dictionary. The number of entries in the dictionary totals 133749 and comprises 60095 nouns, 16999 adjectives, 15853 verbs and 5352 adverbs. An entry is a single word for uninflected parts of speech and a set of words for inflected ones. A deeper explanation of inflexional paradigms as well as algorithms which make it possible that significantly more words can be recognized is outside the scope of the study.

The training data is fourteen Polish novels edited in machine-readable form by Marek Adamiec in the Institute of Polish Philology at the University of Gdańsk.[10] The novels are mentioned in table 2. The chi-square test for homogeneity[11] strongly rejects the hypothesis of one common distribution of parts of speech in the novels.

The test data set consists of eleven first fragments of contemporary novels, tagged by the author. Each fragment is about a thousand words long. The novels are mentioned in table 3.

3 Results

Using the training data set, we estimated conditional probabilities (2) according to (7). We used the obtained estimators instead of true probabilities in (3) to maximize (5) in order to find a tagging of each text of the test data set. Tagging accuracy, that is the fraction of correctly tagged words, was calculated on a random subset of 300 words sampled from the test data set. The accuracy proved to be 87.3%. Finding it to be unsatisfying, we investigated fractions of erroneous classifications of L_i as L_j, $i, j = 1, 2, \ldots, 29$. Throughout all the test data set, these fractions exceeded 10% in the following cases: (a) L_3, L_4 and L_5 were tagged as L_1 in 22%, 23% and 16%, respectively, (b) L_5 was tagged as L_6 in 11%, (c) L_{20} was tagged as L_8 in 62%, (d) L_{21} was tagged as L_{17} in 16%, (e) L_{27} was tagged as L_{28} or L_{29} in 22% and 28%, respectively, (f) L_{28} was tagged as L_2 or L_{13} in 18% and 22%, respectively, (g) L_{29} was tagged as L_{26} in 38%.

Tagging nouns as unknown words is the evidence of limited contents of the dictionary. Using nouns like *papieros*, *banan* as masculine-animal or

[9] Downloaded from ftp://ftp.icm.edu.pl/pub/unix/polish-ispell/ in July 2000.
[10] Downloaded from http://monika.univ.gda.pl/~literat/ in December 2003.
[11] See [1, §30.6].

Table 2. Distribution of parts of speech in fourteen Polish novels[a]

Author, title	Nouns	Adj.	Verbs	Adv.	Others
T. Dołęga-Mostowicz, Kariera...	16940	4288	15617	2098	23788
W. Gomulicki, Wspomnienia...	10669	3618	8335	1186	13936
J. I. Kraszewski, Stara baśń	22256	5515	23402	2165	36282
A. Mickiewicz, Pan Tadeusz	17713	4311	11778	1257	20614
E. Orzeszkowa, Nad Niemnem	33372	11967	26780	4198	49095
B. Prus, Faraon	45942	11206	34491	2903	55860
B. Prus, Lalka	51015	11774	44346	5014	73722
W. St. Reymont, Chłopi	54504	13270	55120	6939	98487
W. St. Reymont, Ziemia...	34656	10399	33219	5213	48302
H. Sienkiewicz, Krzyżacy	40576	11953	38204	4146	73471
H. Sienkiewicz, Potop	76173	20129	65189	7552	114968
H. Sienkiewicz, Quo vadis	30307	7889	30008	2540	53048
S. Żeromski, Popioły	54659	16698	36716	3988	67035
S. Żeromski, Przedwiośnie	19231	6541	12760	1669	23472

[a] $\chi^2 = 15094.22$ (52 degrees of freedom), $\Pr(\chi^2 \geq 15094.22) = 0.00$.

masculine-inanimate is a matter of style as long as two different forms of accusative are permissible. The correct tagging of nouns like *stwór*, *rak* seems to be impossible without realizing what they reflect or may reflect. The most common mistake, that is tagging verbal nouns as neuter nouns, is much more difficult to overcome: tagging *życie* in the phrase *ostatnia miłość życia* depends on the way of interpreting it. Tagging adjectival past participles, like *roztajały*, *posiwiały*, *zżółkły*, *wynędzniały*, as verbs is caused by identity of the most often used forms of nominative singular and several forms of the third person of the past tense. True confusion is made when classifying prepositions, conjunctions, particles, interjections, pronouns and words of unknown parts of speech, like *aby*, *ale*, *bo*, *i*, *nie*. Even reflecting the one and only sense, these, mostly functional words, having varying shades of meaning, may be classified into several classes. Unquestionable tagging of such words seems to be impossible without additional suppositions.

Considering the above observations, we included the class of verbal nouns in the class of neuter nouns and united the classes of prepositions, conjunctions, particles, interjections, pronouns and words of unknown part of speech in a class of functional words. This step referred only to the phase of tagging, not calculating probabilities (7). After this manipulation, the tagging accuracy figured out 92.3%. Using the central limit theorem approx-

imation $\Pr\left(a \leq (\hat{p}-p)/\sqrt{\hat{p}(1-\hat{p})/300} \leq b\right) \approx (1/\sqrt{2\pi}) \int_a^b e^{-t^2/2} dt$, where p and $\hat{p}$ are, respectively, true and calculated accuracies, we may say that $\Pr(p \geq 0.887) \approx 0.99$. When calculated only for words that are present in the dictionary, the accuracy is as high as 95.8%. Last of all, if calculated only for ambiguous words, the accuracy is 80.4%. Variability across the texts is illustrated in table 3.

A comparison of the method of tagging described above with the tagger presented in [2] is pointless since substantially different sets of tags are used by them. From the foregoing discussion it would seem, however, that a simple technique applied to existing resources may, in certain cases, be sufficient for tagging text.

Table 3. Tagging accuracy on the test data set

Author, title	All words	Known words	Ambiguous words
K. Brandys, Rondo	94.4	95.4	82.7
S. Chwin, Hanemann	91.7	95.4	79.5
G. Grass, Blaszany bębenek	93.6	95.5	78.8
J. Joyce, Ulisses	91.1	96.2	80.1
J. Kosiński, Przechodząc obok	91.9	92.8	75.4
T. Konwicki, Kalendarz i klepsydra	94.2	95.3	79.9
St. Lem, Solaris	92.5	96.2	80.3
F. Molnar, Chłopcy z Placu Broni	93.2	95.8	79.5
J. Pilch, Pod mocnym aniołem	92.9	95.1	82.8
M. Szołochow, Los człowieka	94.3	95.8	80.1
K. Truchanowski, Totenhorn	93.6	95.6	83.3
Random subset of 300 words	92.3	95.8	80.4

References

1. Cramér H. (1958) Metody matematyczne w statystyce. Państwowe Wydawnictwo Naukowe, Warszawa
2. Dębowski Ł. (2004) Trigram morphosyntactic tagger for polish. Retrieved from http://www.ipipan.waw.pl/~ldebowsk/

3. Hajnicz E., Kupść A. (2001) Przegląd analizatorów morfologicznych dla języka polskiego. Prace IPI PAN 937, Instytut Podstaw Informatyki Polskiej Akademii Nauk, Warszawa
4. Manning Ch. D., Schütze H. (1999) Foundations of Statistical Natural Language Processing. The MIT Press, Cambridge, Massachusetts, London, England
5. Markowski A. (ed.) (2000) Nowy słownik poprawnej polszczyzny PWN. Wydawnictwo Naukowe PWN, Warszawa
6. Nagórko A. (2000) Zarys gramatyki polskiej (ze słowotwórstwem). Wydawnictwo Naukowe PWN, Warszawa
7. Polański E. (ed.) (1996) Nowy słownik ortograficzny PWN. Wydawnictwo Naukowe PWN, Warszawa
8. Szymczak M. (ed.) (1999) Słownik języka polskiego PWN. Wydawnictwo Naukowe PWN, Warszawa
9. Tokarski J. (2001) Fleksja polska. Wydawnictwo Naukowe PWN, Warszawa
10. Vetulani Z., Martinek J., Obrębski T., Vetulani G. (1998) Dictionary based methods and tools for language engineering. Wydawnictwo Naukowe Uniwersytetu im. Adama Mickiewicza, Poznań
11. Woliński M., Przepiórkowski A. (2001) Projekt anotacji morfosyntaktycznej korpusu języka polskiego. Prace IPI PAN 938, Instytut Podstaw Informatyki Polskiej Akademii Nauk, Warszawa

Enhancing a Portuguese Text Classifier Using Part-of-Speech Tags

Teresa Gonçalves and Paulo Quaresma

Departamento de Informática, Universidade de Évora
7000-671 Évora, Portugal
tcg | pq@di.uevora.pt

Abstract. Support Vector Machines have been applied to text classification with great success. In this paper, we apply and evaluate the impact of using part-of-speech tags (nouns, proper nouns, adjectives and verbs) as a feature selection procedure in a European Portuguese written dataset – the Portuguese Attorney General's Office documents.

From the results, we can conclude that verbs alone don't have enough information to produce good learners. On the other hand, we obtain learners with equivalent performance and a reduced number of features (at least half) if we use specific part-of-speech tags instead of all words.

1 Introduction

The learning problem can be described as finding a general rule that explains data given a sample of limited size. In supervised learning, we have a sample of input-output pairs (the *training sample*) and the task is to find a deterministic function that maps any input to an output such that the disagreement with future input-output observations is minimised. If the output space has no structure except whether two elements are equal or not, we have a *classification* task. Each element of the output space is called a *class*.

Our problem can be viewed as a supervised classification task of natural language texts, also known as *text classification*. Research interest in this field has been growing in the last years. Several learning algorithms were applied such as decision trees [14], linear discriminant analysis and logistic regression [10], naïve Bayes algorithm [7] and Support Vector Machines – SVM [6].

In [12], linguistic information is applied on the preprocessing phase of text mining tasks. This work applies a linear SVM to the Portuguese Attorney General's Office dataset – PAGOD [8], addressing the impact of using part-of-speech (POS) tags to build the learner.

In previous work, we evaluated SVM performance compared with other Machine Learning algorithms [5] and performed a thourough study on some preprocessing techniques (feature reduction, feature subset selection and term weighting) and on the performance achieved when balancing the dataset [4].

In Section 2, a brief description of the Support Vector Machines theory is presented, while in Section 3 the PAGOD dataset is characterised. Section

4 describes our experimental setup, Section 5 our previous experiments and Section 6 our present work. Conclusions and future work are pointed out in Section 7.

2 Support Vector Machines

Motivated by theoretical results from statistical learning theory, Vapnik and coworkers [2] introduced the Support Vector Machines learning algorithm. It joins a kernel technique with the structural risk minimisation framework.

Kernel techniques comprise two parts: a module that performs a mapping into a suitable feature space and a learning algorithm designed to discover linear patterns in that space. The *kernel function*, that implicitly performs the mapping, depends on the specific data type and domain knowledge of the particular data source. The *learning algorithm* is general purpose and robust. It's also efficient, since the amount of computational resources required is polynomial with the size and number of data items, even when the dimension of the embedding space grows exponentially [11]. Four key aspects of the approach can be highlighted as follows:

- Data items are embedded into a vector space called the feature space.
- Linear relations are discovered among the images of the data items in the feature space.
- The algorithm is implemented in a way that the coordinates of the embedded points are not needed; only their pairwise inner products.
- The pairwise inner products can be computed efficiently directly from the original data using the kernel function.

These stages are illustrated in Figure 2.

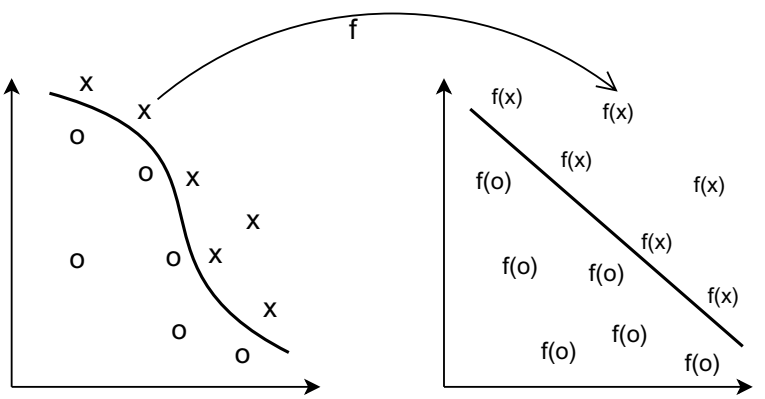


Fig. 1. Kernel function: The nonlinear pattern of the data is transformed into a linear feature space.

The *structural risk minimisation* (SRM) framework creates a model with a minimised VC dimension. This developed theory [15] shows that when the VC dimension of a model is low, the expected probability of error is low as well, which means good performance on unseen data (good generalisation).

SVM can also be derived in the framework of regularisation theory instead of the SRM theory. The idea of regularisation, introduced by Tickonov and Arsenin [13] for solving inverse problems, is a technique to restrict the (commonly) large original space of solutions into compact subsets.

3 Dataset Description

Each PAGOD document is classified into multiple categories so, we have a multi-label classification task. Normally, this task is solved by splitting it into a set of binary classification tasks and considering each one independently.

This dataset has 8151 documents and represents the decisions of the Portuguese Attorney General's Office since 1940. It is written in the European Portuguese language, and delivers 96 MBytes of characters. All documents were manually classified by juridical experts into a set of categories belonging to a taxonomy of legal concepts with around 6000 terms.

A preliminary evaluation showed that, from all potential categories only about 3000 terms were used and from all 8151 documents, only 6388 contained at least one word on all experiments. For these documents, we found 77723 distinct words, and averages of 1592 words and 362 distinct words per document.

Table 1 presents the top ten categories (the most used ones) and the number of documents that belongs to each one.

category	# docs
pensão por serviços excepcionais	906
deficiente das forças armadas	678
prisioneiro de guerra	401
estado da Índia	395
militar	388
louvor	366
funcionário público	365
aposentação	342
competência	336
exemplar conduta moral e cívica	289

Table 1. The top ten categories for the PAGOD dataset.

The Portuguese language is a rich morphological one: while nouns and adjectives have 4 forms (two *genres* – male and female and two *numbers* –

singular and plural), a regular verb has 66 different forms (two *numbers*, three *persons* – first, second and third and five *modes* – indicative, conjunctive, conditional, imperative and infinitive, each with different number of *tenses* ranging from one to five).

4 Experimental setup

This section presents the choices made in our study: document's representation, the process for obtaining the POS tags, the kind of procedure used to reduce/construct features and the measures used for evaluating learner's performance.

The linear SVM was run using the WEKA [16] software package from the University of Waikato, with default parameters. We performed a 10-fold cross-validation procedure.

Representing documents. To represent each document we chose the bag-of-words approach, a *vector space model* (VSM) representation. Each document is represented by the words it contains, with their order and punctuation being ignored. From the bag-of-words we removed all words that contained digits.

Obtaining the POS tags. To obtain each word's POS tag we used a parser for the Portuguese language – PALAVRAS [1]. It was developed in the context of the VISL[1] (Visual Interactive Syntax Learning) project in the Institute of Language and Communication of the University of Southern Denmark.

The POS tagger incorporated in the parser is reported to have more than 95% accuracy for texts written in Portuguese. Possible tags are:

- noun (*nn*),
- proper noun (*prop*),
- adjective (*adj*),
- verb (*vrb*),
- article (*det*),
- pronoun (*pron*),
- adverb (*adv*),
- numeral (*num*),
- preposition (*prp*),
- interjection (*in*) and
- conjunction (*conj*)

From all possible tags, we just considered *nn*, *prop*, *adj* and *vrb*.

Parser's output is the syntactic analysis of each phrase and the POS tag associated with each word.

[1] http://www.visl.sdu.dk/

For example, the morphological tagging of the phrase "O Manuel ofereceu um livro ao pai." is:

```
o [o] <artd> <dem> DET M S
Manuel [Manuel] PROP M S
ofereceu [oferecer] VRB PS 3S IND VFIN
um [um] <quant> <arti> DET M S
livro [livro] NN M S
o [o] <artd> <dem> DET M S
pai [pai] NN M S
```

Reducing/constructing features. On trying to reduce/construct features we used linguistic information: we applied a Portuguese stop-list (set of non-relevant words such as articles, pronouns, adverbs and prepositions) and POLARIS, a lexical database, to generate the lemma for each Portuguese word.

Measuring performance. To measure learner's performance we analysed precision, recall and the F_1 measures [9] (prediction *vs.* manual classification from the contingency table of the classification). For each performance measure we calculated the micro- and macro-averaging of the top ten categories.

Precision is the number of correctly classified documents divided by the number of documents classified into the class. *Recall* is given by the number of correctly classified documents divided by the number of documents belonging to the class. F_1 belongs to a class of functions used in information retrieval, the *F_β-measure*; it is the weighted harmonic mean of precision and recall.

Macro-averaging corresponds to the standard way of computing an average: the performance is computed separately for each category and the average is the arithmetic mean over the ten categories. *Micro-averaging* averages the contingency tables of the various categories. For each cell of the table, the arithmetic mean is computed and the performance is computed from this averaged contingency table.

All significance tests were done regarding a 95% confidence level.

5 Previous experiments

In previous work we compared SVM with other machine learning algorithms [5], namely Naïve Bayes and C4.5. While SVM performance was equivalent to C4.5, the learner building time was much shorter (from hours to minutes).

In [4] we made a set of preprocessing experiments on the PAGOD dataset: feature reduction/construction, feature subset selection and term weighting experiments. Next subsection describes them.

5.1 Description

Reduce/construct features. On trying to reduce/construct features we made three sets of experiments:

- all words;
- remove from the first experiment a *stop-list* of words and,
- from that set of words, transform each onto its *lemma*.

Feature subset selection. For the feature subset selection we used a filtering approach, keeping the features that received higher scores according to different functions:

- *term frequency*: the score is the number of times the feature appears in the dataset;
- *mutual information*: it evaluates the worth of an attribute by measuring the mutual information with respect to the class. Mutual Information, $I(C;A)$, is an Information Theory measure [3] that ranks the information received to decrease the uncertainty. The uncertainty is quantified through the Entropy, $H(X)$.
- *gain ratio*: the worth is the gain ratio with respect to the class. Mutual Information is biased through attributes with many possible values. Gain ratio tries to oppose this fact by normalising mutual information by the feature's entropy.

For each filtering function, we tried different threshold values. This threshold is the number of times the feature appears in all documents. We performed experiences for thr_1, thr_{50}, thr_{100}, thr_{200}, thr_{400}, thr_{800}, thr_{1200} and thr_{1600}, where thr_n means that all words appearing less than n are eliminated.

Term weighting. Finally, for the term weighting experiments, we made four different experiments:

- *binary representation*: each word occurring in the document has weight 1, all others have weight 0 and the resulting vector is normalised to unit length;
- *raw term frequencies*, $TF(w_i, d_j)$, with no collection component nor normalisation, being $TF(w_i, d_j)$ the number of times the word w_i occurs in document w_j;
- TF normalised to unit length;
- *TFIDF representation*, where TF is multiplied by $log(N/DF(w_i))$[2] and normalised to unit length.

[2] N is the total number of documents and $DF(w_i)$ is the number of documents in which w_i occurs.

5.2 Results

We made a total of 288 different experiments. As already mentioned, we measured precision, recall and the F_1 measures and calculated the macro- and micro-averaging for the top ten categories.

The best values were obtained for the following experiments:

- lemmatisation with the term frequency scoring function and term frequencies normalised to unit length;
- stop-list removal with the mutual information scoring function and the binary representation experiment.

The threshold value thr_{400} was chosen since it presented a good trade-off between performance and time consumed (to generate the learner) and it was the best or second best result obtained in all experiments.

6 Part-of-speech tag experiments

In order to assess the impact of the POS tags on the SVM performance, we made several trials, retaining only the words belonging to some specific tag(s). We considered the following experiments:

- nouns (nn)
- verbs (vrb)
- nouns and verbs ($nn + vrb$)
- nouns and proper nouns ($nn + prop$)
- nouns and adjectives ($nn + adj$)
- nouns, adjectives and proper nouns ($nn + adj + prop$)
- nouns, verbs and adjectives ($nn + vrb + adj$)
- nouns, verbs and proper nouns ($nn + vrb + prop$)

For each of these experiments and, on the basis of previous results (see Section 5.2), we used the term frequency for scoring words and the term frequency normalised to unit length for weighting them.

Using this setup, we examined the generated models using threshold values of thr_1 and thr_{400} and the original words and their lemmas.

To have a base value of comparison we also present the values for the best setting of the previous experiments, *base* (lemmatisation, the term frequency scoring function with a 400 threshold value and term weighting by term frequencies normalised to unit length).

Table 2 shows the averages per document (of all and distinct features) and the number of features (for each threshold value) obtained for each POS-tag experiment with original words (*word*) and their lemmas (*lemma*).

	averages per document				total features			
	all		distinct		thr_1		thr_{400}	
	word	lemma	word	lemma	word	lemma	word	lemma
nn	437	424	126	110	24597	20388	1168	1026
vrb	212	184	120	76	27689	8899	601	542
$nn + vrb$	638	598	237	179	49838	27031	1752	1533
$nn + adj$	559	540	175	148	33431	25720	1535	1349
$nn + prop$	547	514	149	130	35273	30123	1329	1165
$nn + adj + prop$	668	630	196	166	43229	34877	1679	1473
$nn + vrb + adj$	759	714	285	216	58052	31981	2122	1855
$nn + vrb + prop$	747	688	260	198	59742	36287	1917	1669
$base$	1592	912	362	255	77723	42421	2753	2114

Table 2. Averages per document and total of features for each experiment.

	micro-averaging				macro-averaging			
	word		lemma		word		lemma	
	thr_1	thr_{400}	thr_1	thr_{400}	thr_1	thr_{400}	thr_1	thr_{400}
nn	0.807	0.887	0.802	0.888	0.739	0.795	0.733	0.791
vrb	0.825	0.926	0.827	0.924	0.699	0.701	0.664	0.703
$nn + vrb$	0.831	0.850	0.801	0.858	0.760	0.775	0.742	0.786
$nn + adj$	0.809	0.871	0.807	0.880	0.745	0.793	0.744	0.796
$nn + prop$	0.812	0.879	0.807	0.879	0.746	0.801	0.744	0.795
$nn + adj + prop$	0.820	0.862	0.811	0.866	0.757	0.788	0.751	0.795
$nn + vrb + adj$	0.837	0.842	0.815	0.856	0.766	0.776	0.755	0.790
$nn + vrb + prop$	0.837	0.845	0.815	0.854	0.768	0.776	0.756	0.786
$base$	–	–	0.819	0.836	–	–	0.757	0.775

Table 3. Precision for each POS tag experiment.

6.1 Results

For each experiment, we, once again, analysed *precision*, *recall* and F_1 measures and calculated the micro- and macro-averaging of the top ten categories. Tables 3, 4 and 5 show, respectively, precision, recall and F_1 values.

Considering macro-averaging F_1 values, the worst significant experiments were vrb (with words or lemmas, for both threshold values) and nn (lemmas with thr_1). The micro-averaging F_1 worst significant values were obtained for the same experiments and also for $nn + vrb$ and $nn + adj$ (lemmas with thr_1) and nn (words with thr_1).

7 Conclusions and Future Work

From the previous section is possible to conclude that considering just the most frequent words (thr_{400}) learner's performance is not decreased, but we

	micro-averaging				macro-averaging			
	word		lemma		word		lemma	
	thr_1	thr_{400}	thr_1	thr_{400}	thr_1	thr_{400}	thr_1	thr_{400}
nn	0.767	0.753	0.760	0.748	0.717	0.694	0.707	0.690
vrb	0.722	0.683	0.707	0.680	0.662	0.611	0.645	0.610
$nn + vrb$	0.770	0.772	0.774	0.772	0.719	0.719	0.723	0.719
$nn + adj$	0.777	0.771	0.774	0.762	0.727	0.716	0.724	0.705
$nn + prop$	0.776	0.770	0.776	0.764	0.725	0.715	0.725	0.708
$nn + adj + prop$	0.783	0.777	0.784	0.783	0.732	0.724	0.734	0.731
$nn + vrb + adj$	0.768	0.778	0.781	0.783	0.717	0.726	0.732	0.731
$nn + vrb + prop$	0.771	0.776	0.782	0.784	0.719	0.723	0.733	0.733
$base$	–	–	0.782	0.787	–	–	0.731	0.737

Table 4. Recall for each POS tag experiment.

	micro-averaging				macro-averaging			
	original		w/ lemmas		original		w/ lemmas	
	thr_1	thr_{400}	thr_1	thr_{400}	thr_1	thr_{400}	thr_1	thr_{400}
nn	0.787	0.814	0.781	0.812	0.727	0.728	0.719	0.721
vrb	0.770	0.786	0.762	0.783	0.649	0.642	0.639	0.645
$nn + vrb$	0.799	0.809	0.787	0.813	0.737	0.742	0.732	0.747
$nn + adj$	0.793	0.818	0.790	0.817	0.735	0.745	0.733	0.739
$nn + prop$	0.794	0.821	0.791	0.817	0.735	0.746	0.734	0.738
$nn + adj + prop$	0.801	0.817	0.797	0.822	0.743	0.749	0.742	0.754
$nn + vrb + adj$	0.801	0.809	0.797	0.818	0.738	0.747	0.743	0.756
$nn + vrb + prop$	0.803	0.809	0.798	0.818	0.740	0.745	0.743	0.754
$base$	–	–	0.800	0.811	–	–	0.743	0.753

Table 5. F_1 values for each POS tag experiment.

have a decreased number of features with more than one order of magnitude (see Table 2), which implies faster model generation.

We can also conclude that verbs alone don't have enough information to produce good learners. This was expected since, as already mentioned, a Portuguese regular verb has 66 different modes. Nonetheless, this conclusion is true even when using words' lemmas.

On the other hand, using just the words of specific POS tags we obtain learners with equivalent performance of the ones with all words, with a reduced number of features. From all the best experiments, the one with less features is nn with lemma and thr_{400}, with 1026 features, and the one with more is $base$ with lemma and thr_1 with 42421 features.

As future work, and in order confirm these results, we to intend make these same experiments with other datasets (newspaper news datasets: *Público* written in European Portuguese and *Folha de S. Paulo*, written in Brazilian Portuguese).

On the other hand, and aiming to develop better classifiers, we intend to address the document representation problem by trying more powerful representations than the bag-of-words that allow us to use word order and syntactical and/or semantical information in the representation of documents. To achieve this goal we plan to use other kind of kernel such as the string kernel (see, for example, [11]).

References

1. E. Bick. *The Parsing System "Palavras". Automatic Grammatical Analysis of Portuguese in a Constraint Grammar Framework*. Aarhus University Press, 2000.
2. Cortes and Vapnik. Support-vector networks. *Machine Learning*, 20(3), 1995.
3. Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley Series in Telecomunication. John Wiley and Sons, Inc, New York, 1991.
4. T. Gonçalves and P. Quaresma. Evaluating preprocessing techniques in a text classification problem. (submitted to an international conference).
5. T. Gonçalves and P. Quaresma. A preliminary approach to the multilabel classification problem of Portuguese juridical documents. In F. Moura-Pires and S. Abreu, editors, *11th Portuguese Conference on Artificial Intelligence, EPIA 2003*, LNAI 2902, pages 435–444, Évora, Portugal, December 2003. Springer-Verlag.
6. T. Joachims. *Learning to Classify Text Using Support Vector Machines*. Kluwer academic Publishers, 2002.
7. D. Mladenić and M. Grobelnik. Feature selection for unbalanced class distribution and naïve bayes. In *Proceedings of ICML-99, 16th International Conference on Machine Learning*, pages 258–267, 1999.
8. P. Quaresma and I. Rodrigues. PGR: Portuguese Attorney General's Office decisions on the web. In Bartenstein, Geske, Hannebauer, and Yoshie, editors, *Web-Knowledge Management and Decision Support*, LNCS/LNAI 2543, pages 51–61. Springer-Verlag, 2003.
9. G. Salton and M. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, 1983.
10. H. Schütze, D. Hull, and J. Pedersen. A comparison of classifiers and document representations for the routing problem. In *Proceedings of SIGIR-95, 18th International Conference on Research and Developement in Information Retrieval*, pages 229–237, Seattle, WA, 1995.
11. J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.
12. C.F. Silva, R. Vieira, F.S. Osorio, and P. Quaresma. Mining linguistically interpreted texts. In *5th International Workshop on Linguistically Interpreted Corpora*, Geneva, Switzerland, August 2004.
13. V.M. Tikhonov and V.Y. Arsenin. *Solution of Ill-Posed Problems*. Winston, Washington DC, 1977.
14. R. Tong and L.A. Appelbaum. Machine learning for knowledge-based document routing. In Harman, editor, *Proceedings of 2nd Text Retrieval Conference*, 1994.
15. V. Vapnik. *Statistical learning theory*. Wiley, NY, 1998.
16. I. Witten and E. Frank. *Data Mining: Practical machine learning tools with Java implementations*. Morgan Kaufmann, 1999.

A Neural Network Based Morphological Analyser of the Natural Language

Piotr Jędrzejowicz[1] and Jakub Strychowski[2]

[1] Chair of Information Systems, Gdynia Maritime University, Gdynia, Poland
[2] Rodan Systems S.A., Sopot, Poland

Abstract. The paper proposes a morphological analyser supported by a neural network to inflect words written in Polish. The approach can be also applied to other languages. The main task of the analyser is to create base forms from the analysed words' forms. Other objective is to provide grammatical information for the analysed form. Computational experiment results confirm that both objectives are fulfilled by the proposed neural network based morphological analyser. The common words are inflected with a very high quality of nearly 99.9%. Other words like geographical names and people's names, thanks to the incorporation of neural network, inflect with a quality reaching 93.3%.

1 Introduction

A natural language processing consists of succeeding steps of a speech and text analysis. Every step of this process (i.e. speech recognition, tokenization, morphological analysis, parsing, semantic knowledge mining) causes some errors. A quality of a final product depends on cumulative influence of all analysis elements, therefore it is very important to make every step of the NLP as reliable and effective as possible.

A morphological analysis, one of the NLP elements, means processing word forms without considering context and has fundamentally two objectives:

- to obtain the grammatical information from a single word form,
- to assign the word form to its base form.

The quality of morphology analysis depends on the processed language [4]. For example, the morphology analysis of words in English is not very difficult, because English words contain little grammatical information. Unfortunately, morphology analysis must be expanded to the part-of-speech tagging process for this language. POS (Part Of Speech) tagger determines a grammatical function of the word through analysing the context of the word in the sentence [1]. Other languages may have rich inflection, and a morphological analyser development could be very difficult in such cases. Reward for this is the grammatical information obtained during morphology analysis process. For example in the Polish language, by examining only a single form of the word, one can determine tense, part of speech, case, number etc.

Such information could be very helpful in the successive steps of the natural language processing task [4].

Morphological analysers are usually constructed using one of the following approaches [12]:

- A dictionary approach, where the main part of the analyser is a dictionary containing all words' forms in a given language. Analysis of a given word is reduced to the searching for the matching form in the dictionary, and to reading grammatical information from this dictionary [13] [2] [8].
- An algorithmic approach, where analyser contains a set of the inflection rules. Appropriate rules are being applied to a given word to obtain its base form. Grammatical information depends on the chosen rules [11] [3] [7].

The first approach ensures high quality of the analysis, but only for the forms which exists in the dictionary. Its main weaknesses include a substantial memory requirements and labour-consuming need for the dictionary development. The most important advantage of the second approach is its ability to analyse forms which do not occur in the dictionary. This is an advantage for the Polish language which could have geographical names, and peoples' names written in many inflected forms. Weakness of this approach are lower quality and a problem with developing a good set of rules [5].

The paper proposes a new solution integrating both of the above approaches. A full dictionary of the Polish language is used as a training set. During the training process applicable inflection patterns (similar to the inflection rules) are developed. A decision tree helps to assign appropriate inflection pattern to the given word's form. The tree is developed through the affixes analysis during the training. The focus of the research was to show that an artificial neural network can increase analyser quality through reinforcing abilities of the decision tree.

The following section describes an architecture, training process, and working rules of the morphological analyser. In the next section a neural network approach, improving analyser abilities, is suggested. To verify the approach a computational experiment was carried. Its results are, subsequently, presented and discussed. The final section contains conclusions and ideas for future research.

2 The Morphological Analyser

One of the possible methods of carrying word morphology analysis is an assignment of a given word's form to a valid inflection pattern. An inflection pattern consists of the set of affixes describing how to create specified form from a word's root. An affix is a chunk of a word which can occur before (prefixes), after (suffixes) and even inside (infixes) word's root. Each word can be constructed from a single root and from many affixes. To make things

simplified, a group of affixes constructing specified form is called a *supplement* in this article. So, the inflection pattern consists of the supplements.

For example Polish word *dziadek* (*grandfather* in English) can be written in the following forms:

dziad**ek**, dziad**ka**, dziad**kowi**, dziad**kiem**, dziad**ku**, dziad**ki**, dziad**ków**, dziad**kom**, dziad**kami**, dziad**kach**

As we can see this word contains root *dziad-*, and can have following suffixes:

-ek, -ka, -kowi, -kiem, -ku, -ków, -kom, -kami, -kach.

Such a set of suffixes can be treated as an inflection pattern. An another Polish word — *szybko* (adverb *quick* in English) — can be written in the following forms:

szyb**ko**, szyb**ciej**, **naj**szyb**ciej**

The forms of this word have root: *-szyb-*; suffixes: *-ko*, *-ciej*; and prefix *naj-*. An inflection pattern valid for this word can be written as the following set of supplements:

-ko, -ciej, naj-ciej

Each supplement in the inflection pattern is described by the grammatical information. One of these supplements is marked as a base form. So, the full information about inflection pattern for the Polish word *szybko* can be written in the following form:

-ko [baseform + adverb], **-ciej** [adverb + comparative], **naj-ciej** [adverb + superlative]

All possible inflection patterns are stored in the inflection patterns base.

A simple way to make morphological analysis is to find a valid inflection pattern for the analysed form. An assignment of a given word's form to a valid supplement stored in the correct inflection pattern allows to describe word's form by the grammatical information, and allows to create any word's form (specially the base form).

The above method seems to be quite simple to apply, but unfortunately, it hides many difficulties. One of the problems is a creation of the inflection patterns. There are inflection patterns developed for many languages by the linguists, but these patterns are too general and cannot be used in the computational linguistics in many cases. Another problem is the required performance of an analyser. A search for the matching pattern could be very time-consuming especially when a number of patterns is high (i.e. greater than 1000). Nevertheless, the main problem for the analyser is to remove ambiguity. It turns out that supplement analysis can assign a single word's

form to many inflection patterns. For example the supplement **a* can be assigned to most of the inflection patterns in the Polish language (wildcard *** means any sequence of characters here). So, an analyser needs to assign the word's form to a valid inflection pattern (in some situations an assignment to many patterns is also correct).

The first problem — creation of the inflection patterns — is solved using a full dictionary of a given language. Such dictionary contains all possible forms of the frequently used words. It is possible to determine root and supplements of a given word having all its forms. Obtained supplements, sorted in the order proper for the part-of-speech, create an inflection pattern. Words inflected in the same way create a common inflection pattern. A set of inflection patterns consist of all different patterns. The grammatical information for all patterns can be easily added by a linguist.

A total number of inflection patterns depends on the language and a quality of the dictionary used as a training set. A set of inflection patterns can contain hundreds or even thousands of elements. The morphology analysis could be very time-consuming if it is applied to a huge collection of documents. Hence, it is important to optimize searching in the inflection patterns' set. This can be done by storing inflection patterns as a decision tree.

Successive characters of the supplements (starting from the supplement's end) are stored as the nodes of the tree. The nodes attached to the tree's root are related to the last characters in the supplements. The nodes from the next levels are related to the succeeding characters. Wildcart character (***) is also stored as a node. The leafs of the tree relate to the inflection patterns.

A decision tree created from the inflection patterns related to the Polish words *arktyczny* (*arctic* in English), *bakteria* (*bacteria* in English) and *szybko* is shown in the figure 1.

Analysis of the word's form is based on traversing a tree starting from its root and reaching its leafs through all possible pathes. On each node, a decision about traversing to the valid sub-nodes is made. Using a terminology from the machine learning domain the edges between nodes represent decisions, and the leafs represent all possible hypotheses. Finding inflection patterns is a simple classification task. An important feature of the "tree solution" is its high speed. Assignment of a single inflection pattern to the analysed form takes time which can be compared to the time used during iteration through the characters from the form's end to the form's begin.

The decision tree allows to find inflection patterns having supplements which match the analysed form. Unfortunately, it is quite often that some patterns towards which the decision tree is pointing should not be assigned to a particular form. The problem is to choose a valid pattern from all possible candidates returned by the decision tree.

For example, the decision tree shown in the figure 1 assigns inflection patterns *1* and *2* to the forms *arktyczne* and *bakterie*. For the first form only inflection pattern *1* is correct, and for the second form only inflection

form to many inflection patterns. For example the supplement **a* can be assigned to most of the inflection patterns in the Polish language (wildcard *** means any sequence of characters here). So, an analyser needs to assign the word's form to a valid inflection pattern (in some situations an assignment to many patterns is also correct).

The first problem — creation of the inflection patterns — is solved using a full dictionary of a given language. Such dictionary contains all possible forms of the frequently used words. It is possible to determine root and supplements of a given word having all its forms. Obtained supplements, sorted in the order proper for the part-of-speech, create an inflection pattern. Words inflected in the same way create a common inflection pattern. A set of inflection patterns consist of all different patterns. The grammatical information for all patterns can be easily added by a linguist.

A total number of inflection patterns depends on the language and a quality of the dictionary used as a training set. A set of inflection patterns can contain hundreds or even thousands of elements. The morphology analysis could be very time-consuming if it is applied to a huge collection of documents. Hence, it is important to optimize searching in the inflection patterns' set. This can be done by storing inflection patterns as a decision tree.

Successive characters of the supplements (starting from the supplement's end) are stored as the nodes of the tree. The nodes attached to the tree's root are related to the last characters in the supplements. The nodes from the next levels are related to the succeeding characters. Wildcart character (***) is also stored as a node. The leafs of the tree relate to the inflection patterns.

A decision tree created from the inflection patterns related to the Polish words *arktyczny* (*arctic* in English), *bakteria* (*bacteria* in English) and *szybko* is shown in the figure 1.

Analysis of the word's form is based on traversing a tree starting from its root and reaching its leafs through all possible pathes. On each node, a decision about traversing to the valid sub-nodes is made. Using a terminology from the machine learning domain the edges between nodes represent decisions, and the leafs represent all possible hypotheses. Finding inflection patterns is a simple classification task. An important feature of the "tree solution" is its high speed. Assignment of a single inflection pattern to the analysed form takes time which can be compared to the time used during iteration through the characters from the form's end to the form's begin.

The decision tree allows to find inflection patterns having supplements which match the analysed form. Unfortunately, it is quite often that some patterns towards which the decision tree is pointing should not be assigned to a particular form. The problem is to choose a valid pattern from all possible candidates returned by the decision tree.

For example, the decision tree shown in the figure 1 assigns inflection patterns *1* and *2* to the forms *arktyczne* and *bakterie*. For the first form only inflection pattern *1* is correct, and for the second form only inflection

tiplication is fed into the body of the neuron. The neuron adds up all the products. The weighted sum of the products is usually denoted as net in the neural network literature [6]. Finally, the neuron computes its output as a certain function of net. This function is called the activation or transfer function. The same output value produced by the activation function can be sent out through multiple edges emerging from the neuron. Some neurons are also connected to the input of the neural network. Similarly, the outputs of some neurons represent the output of the network. In most cases layered architecture of the neural network is used. Each layer consist of separate neurons. Each input of the layer is connected to all neurons in the layer. Neurons' outputs stand for layer output. Succeeding layers are connected in the serialized form.

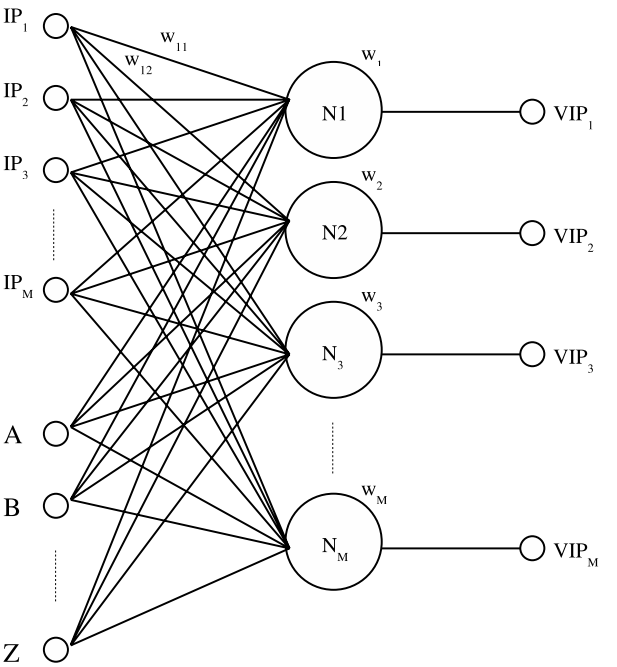


Fig. 2. Architecture of the proposed neural network

Having set up the architecture, the neural network can be trained. The learning task of the neural network is to adjust the weights so that it can output the target signal for each input signal. The neural network is trained by a learning algorithm, which modifies weights of the network during presentation of example pairs of input and target output signals. Deeper and more advanced analysis of the neural networks can be found, for example, in [6], [9].

In the morphology analysis, neural network selects the valid inflection pattern from all the candidates returned by a decision tree. The selection can be performed by a neural network constructed from a single layer of neurons. The inputs of this layer points to the inflection patterns, which are stimulated by the decision tree. Each output of the layer points to the target inflection pattern. Decision tree generates a list of candidates and stimulates the neural network and the network produces the output signals. Signals for all inputs pointing to the patterns selected by the decision tree are set to *1*. Remaining input signals are set to *0*. A neuron with the highest output value designates valid inflection pattern for the analysed form.

In some situations, it would be impossible to determine a valid inflection pattern having only list of candidates as an input, particularly if a different choice should be made for the same lists of candidates. The problem is resolved by adding inputs in form of the alphabetic letters. If a letter occurs in the analysed form then corresponding input is activated (set to *1*). This provides enough information to make an appropriate choice.

Architecture of the described neural network is presented in the figure 2. The input signals related to the inflection patterns are denoted as $IP_1 \ldots IP_M$ ($IP-$ inflection pattern, $M-$ number of all patterns). Symbols from "A" to "Z" stand for input signals related to the letters which can occur in any word in a given language. Neurons are denoted as $N_1 \ldots N_M$, and $w_1 \ldots w_M$ indicate weights of neurons. Symbol w_ij indicates weight of the connection between an input i, and a neuron j. An output signal related to the inflection pattern VIP_x (Valued Inflection Pattern number x) is calculated using a following activation function:

$$VIP_x = w_x \cdot ((\sum_{i=1}^{M} IP_i \cdot w_{ix})) + (\sum_{l="A"}^{"Z"} l \cdot w_{lx}))) \quad (1)$$

A neural network training was based on applying a back-propagation algorithm [9] to a training set. In the discussed case the training set includes all succeeding words' forms from the full dictionary.

The figure 3 shows an example neural network stimulated during an analysis of the Polish word *bakterie*. In this case the decision tree returns 2 candidates: inflection patterns *1* and *2*. Values of the related neural network's inputs are set to *1*. Also inputs related to the letters which occurs in the word *bakterie* are stimulated. Each of the neurons N_1, N_2, N_3 produces output signal. Output denoted as VIP_2 has the highest value so the second inflection pattern stands for the output of the analysis.

4 Computational experiment results

The quality of the proposed morphological analyser was evaluated through analysis of all word's forms available in the full dictionary of the Polish language. The analysis of any word is correct if this word is converted to the

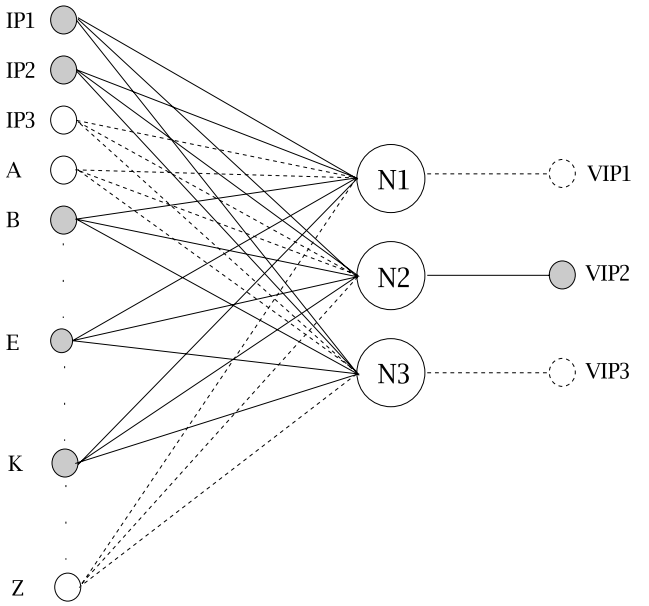


Fig. 3. Neural network analysing the Polish word "bakterie"

valid base form. This condition is fulfilled if a valid inflection pattern is selected. The quality measure is a ratio between a number of correctly inflected words and a number of all analysed forms. The quality calculated in such way is weighted down by the error rate of the dictionary.

For the Polish language only 58,3% of all available forms (about 2 500 000) are inflected correctly if only simple decision tree is used. Usage of the extended tree increases the quality measure to 77%. Usage of a dictionary of the base forms (which contains about 100 000 words) allow to inflect with quality near 99.9%. Such result can be achieved only for the common Polish words. The words which base forms aren't presented in the dictionary are inflected with a quality near 93.3%, which is achieved by a neural network based analyser.

Replacing a back-propagation algorithm as a training tool by a genetic algorithm [14] allows to obtain a quality of the analysis reaching 78%. Thus, the back propagation algorithms seems to perform better. Nevertheless it is worthwhile to possibly test more advanced training algorithms with a view to further improving a quality of the morphological analyser.

The described morphological analyser was implemented in a Java language. The tests have been carried on a PC computer with AMD Athlon XP 1900+ CPU, and 512MB RAM on the board. Analyser has been trained and tested using Sun JVM 1.4.2 for Linux. Table 1 shows experiment results.

Table 1. Summary of the experiment results

Property	Value
Number of the training words	126263
Number of the all words' forms	**2450612**
Number of the supplements	13967
Number of the inflection patterns	1589
Number of frequently used inflection patterns	**73**
Inflection patterns determination time	2h 02m 51s
Extended decision tree construction time (extended tree)	36s
ANN training time (back-propagation, single iteration)	1h 12m 33s
Total construction time	3h 17m
Memory allocated by the inflection patterns	445 kB
Memory allocated by the extended decision tree	1245 kB
Memory allocated by the base forms dictionary	950 kB
Memory allocated by the exceptions dictionary	707 kB
Memory allocated by the ANN	468 kB
Total memory allocated by the morphological analyser	**4261** kB
Analyser initialization time	**345** ms
Analyser speed (only decision tree)	7391 words/s
Analyser speed (decision tree + ANN)	814 words/s
Analyser speed (decision tree + base forms dictionary + ANN)	**4244** words/s
Analyser quality (only decision tree)	77.41 %
Analyser quality (decision tree + ANN)	**93.30** %
Analyser quality (decision tree + base forms dictionary + ANN)	**99.95** %

5 Conclusions

The presented morphological analyser was developed as a part of the ICONS project [10]. Its version is used in a full-text categorisation tool as a stemmer. The analyser is also integrated with the Lucene, an open source full-text search engine, and allows to search words written in the Polish language. The proposed solution provides a very high quality of the inflection analysis for the common words. Using a neural network based analyser with the extended decision tree increases quality of analysis from 58.3% to 93.3% for words not available in the dictionary.

The proposed morphological analyser is fully functional but it seems still possible to increase quality of the analysis. One of the possible approaches is using a better training dictionary containing less errors and more training examples. Some errors can be removed by the inflection patterns analysis. For example, many of the inflection patterns related to less then 10 words were created because of the errors in the dictionary. It is possible to attain a good inflection analysis for the words related with such inflection patterns, by improving a dictionary and reducing a number of inflection patterns.

Another possibility for improving the analyser is upgrading inflection patterns. It is possible to create a diminutive or augmentative forms adding spe-

cial affixes to the Polish nouns. The number of well inflected forms can be doubled by adding to the inflection patterns the supplements adequate to these forms. Also supplements for the derivatives (i.e. noun created from the verb) can be added.

The morphological analyser should work better for the frequently used words in the real applications. A good approach is to include a distribution of the words' occurrence in the training process. This could increase the quality of the analysis of the real documents.

Back-propagation algorithm trains neural network with good results. Nevertheless, other learning algorithms or network's architectures can be tried to reach higher quality.

References

1. Allen J. (1995) Natural Language Understanding. — Redwood City. CA: Benjamin Cummings
2. Dubisz S. ed. (2003) Uniwersalny słownik języka polskiego. — Wydawnictwo Naukowe PWN
3. Frakes W.B., Baeza-Yates, R. (1992) Stemming Algorithms. Information Retrieval — Data Structures and Algorithms. 131-160 . Prentice Hall, London
4. Hajic J. (2000) Morphological Tagging: Data vs. Dictionaries. — In Proceedings of ANLP-NAACL Conference, 94-101. Seattle, Washington, USA
5. Karttunen L., Beesley K.R. (2001) A Short History of Two-Level Morphology. — Presented at the ESSLLI-2001 Special Event titled "Twenty Years of Finite-State Morphology.". Helsinki, Finland
6. Korbicz J., Obuchowicz A., Uciński D. (1994) Sztuczne Sieci Neuronowe. Podstawy i Zastosowania. — Akademicka Oficyna Wydawnicza PLJ
7. Porter M. (1980) An algorithm, for suffix stripping. — Program.
8. Prywata M., Gackiewicz P., Macewicz W. (2004) Polish dictionary for aspell,ispell,myspell. — http://www.kurnik.pl/slownik http://ispell-pl.sourceforge.net/
9. Rutkowska D., Piliński M., Rutkowski L. (1999) Sieci neuronowe, algorytmy genetyczne i systemy rozmyte. — Wydawnictwo Naukowe PWN.
10. Staniszkis E., Nowicki B. (2004) ICONS based Knowledge Management in the Process of Structural Funds Projects Preparation. — Accepted for the e-Challenges, e-2004 Conference. Vienna, Austria
11. Szafran K. (1993) Automatyczna analiza fleksyjna tekstu polskiego. — Rozprawa doktorska, Wydział Polonistyki UW. Warszawa
12. Viks (1994) A morphological analyzer for the Estonian language: the possibilities and impossibilities of automatic analysis. — Automatic Morphology of Estonian 1: 7-28
13. Weiss D. (2002) Polski lematyzator. — http://www.cs.put.poznan.pl/dweiss/
14. Xin Yao (1999) Evolving Artificial Neural Networks. — Proceedings of the IEEE, vol 87, No 9

An Oracle Grammar-Rule Learning Based on Syntactic Knowledge

Minghu Jiang, Huiying Cai, Dafan Liu, and Junzhen Wang

Lab of Computational Linguistics, Dept. of Chinese Language, Tsinghua University, Beijing, 100084, China
jiang.mh@tsinghua.edu.cn

Abstract. In this paper, we put forward two algorithms for Chinese oracle-bone grammar rules learning: automatically learning rule sets and the error-emended learning from the corpus. Through analysing the syntactic knowledge of oracle-bone inscription, an error-emended learning method is used to construct rule-base of oracle-bone phrase structure, which combines linguist's introspection and summarize with corpora to capture rules and describe them with formalized symbols. By using part-of-speech and contextual information together, our experimental results show the learning and expression are effective for oracle-bone grammar rule-base.

1 Introduction

Oracle-bone inscription is the ancient Chinese character in dynasty SHANG at 1400 B.C. It was called Oracle-bone inscription because these characters were engraved on animal bones or tortoise shells.

Since the oracle-bone inscription was founded in 1899, more than 100000 bones and shells have been excavated (see Fig.1), including about 4500 different kinds of characters. After many scholars' research, about 2000 characters have been identified and understood, most of which are proper nouns (personal name and placename) [1].

With the fast development of computer sciences the information of oracle-bone inscription can be used and exploited sufficiently, scholars need to put forward a whole new method to improve the processing of oracle-bone inscription. At the present time many research works about oracle-bone inscription has been done. Especially, the grammar research of oracle-bone inscription also has got some academic achievements [2]. By exploring more than 100 years, people have created a integrated oracle-bone inscription corpus [3], and achieved the word segmentation and part-of-speech (POS) syntax analysis [4], and made the syntax analysis of oracle-bone inscription possible. However, until today there are no any reports about oracle-bone inscription' automatic recognition and explanation systemically.

2 Syntactic Knowledge of Oracle-bone Inscription

2.1 The construction of dictionary

First, a segmentation dictionary was constructed. We've set up a dictionary of oracle-bone inscription including 2373 words. The dictionary is built into open form, as shown in Fig.2. Users can manage it and add new words when necessary. On the other hand, we developed a multiple-information electronic dictionary which make use of four-dimension description approach to discover and mine lexical, syntax, semantic and pragmatic words and phrases of oracle-bone inscription. The basis of lexicon attribute is based on glossary knowledge combining syntax, semantic and pragmatic description, including the following four parts:

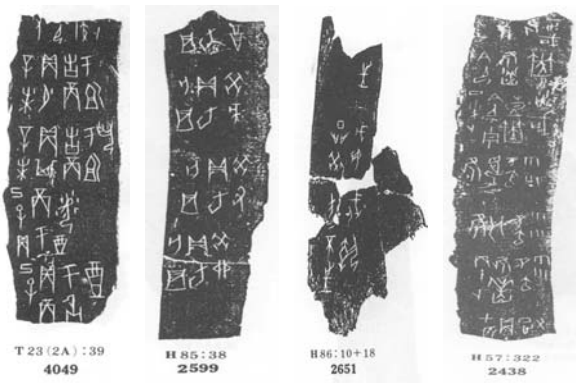


Fig. 1. Oracle-bone inscriptions carved on animal bones or turtleback at 1400 B.C

1. Basis information: including oracle-bone inscription, modern Chinese characters, Chinese phoneticize spell (pinyin), usual fake characters, variant characters, word-formation manner, font style explanation etc.
2. Syntax attribute: including part of speech, in common use arrange in pairs or groups, example paraphrase etc.
3. Semantic attribute: including semantic sort, basic acceptation, explicative acceptation and semantic arrange in pairs or groups etc.
4. Pragmatic attribute: including lexical appearance era, in common use style, serial number of lexical appearance and evolvement process (for example, vehicle's evolving process is etc.

2.2 Syntactic Knowledge

Oracle-bone inscription has a history more than 3000 years. Although never previous reference instances can be used for analysis of oracle-bone inscrip-

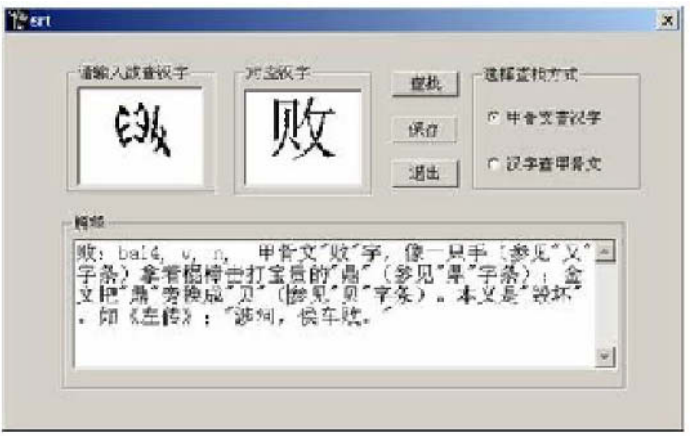


Fig. 2. The dictionary of oracle-bone inscription

tion, at that time the language system of oracle-bone inscription was unabridged and showed a strong resemblance to the modern Chinese system. Therefore, when the oracle-bone inscription is processed with computer, we can consult the methods of modern natural language processing. Oracle-bone inscription with computer processing is divided into three modules, i.e., pre-processing module, syntax analysis module, interpreting and explaining module, as shown in Fig. 3. In the paper, we concerned the second module of intelligent processing for oracle-bone inscription, mainly providing oracle grammar rule learning based on syntactic knowledge.

The phrase of oracle-bone inscription is a basic structure of a sentence. By surveying the evolving process from oracle-bone inscription to modern Chinese language, although the development of oracle-bone inscription enable morphology, structure and semantic glossary to change greatly over 3000 years, they are belong to the same systematic language, the relation of modern Chinese and oracle-bone inscription comes down in one continuous line which is exhibited as follows:

1. The sentence structure and phrase structure are basically identical in oracle-bone inscription, which is "realization relation" between phrase and sentence;
2. There is no simple relationship in one corresponding to one between POS and syntax components;
3. Syntax components are layer upon layer;
4. Words and phrases' changes in sequence have agility and levity.

The sentence of oracle-bone inscription, like that of modern Chinese, is composed of eight parts: subject, predicate, object, attribute, adverbial modifier, complement and governor. The structure of the oracle-bone inscription, like that of modern Chinese too, is organized on the basis of SVO pattern

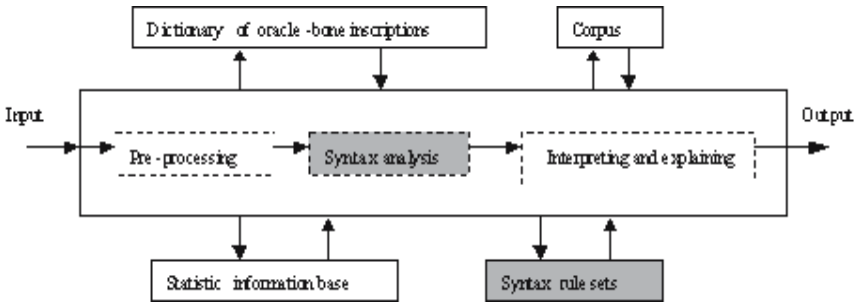


Fig. 3. Oracle-bone inscription with computer processing

(Subject + Verb + Object). Moreover, it also shares some special sentence structures with ancient Chinese, such as subject-verb reversal, prepositive object, prepositive syllepsis, postpositive time-noun, prepositive prepositional phrase and postpositive attribute. We selected and classified 1000 random sentences from corpus of oracle-bone inscription [1], which result as shown in the Table 1.

Table 1. The Structure and Sentence Genres of the Oracle-bone Inscription

Structure of sentence	Numbers	Sentence Genres	Number
Base structure	763	Interrogative sentence	883
Subject-verb reversal	24	Narrative sentence	102
Prepositive object	83	Imperatival sentence	7
Prepositive syllepsis	37	Excalmatory sentence	8
Postpositive time-noun	28	Other sentence	2
Prepositive prepositional phrase	43		
Postpositive attribute	22		

Table 2. The Function Description of Words and Phrases for Oracle-bone Inscription

Tag	Function class	Function															
		A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
Dj	Single sentence	+	+									+			+			
Np	Noun	+					+					+		+		+		+
Vp	Verb		+	+						+			+		+		+	
Ap	Adj.		+		+				+	+	+	+						
Avp	Adverb					+		+										
Mp	Quantifier	+							+	+								

A: subject, B: predication, C: adnex1, D: adnex2, E: adverbial modifier 1, F: object, G: complement, H: attribute, G: adverbial modifier 2, H: center phraseology 1, I: center phraseology 2, J: front-back term of join predications, K: front-back term of unite structure, L: front term of pluralism words and phrases, M: middle term of pluralism words and phrases, N: back term of pluralism words and phrases, O: front-back term of parity structure, P: front term of prepositional object, Q: back term of prepositional object.

From the perspective of sentence genres, the oracle-bone inscription is divided into four kinds of classes, which are respectively narrative sentence, interrogative sentence, imperatival sentence and excalmatory sentence. However, due to the divination features of the oracle-bone inscription, there are many problems in the oracle-bone inscription. Interrogative sentences take up a majority of the oracle-bone inscription expressions. Comparatively speaking, imperatival and excalmatory sentences are seldom adopted by oracle-bone inscription, the distribution of sentence genres as shown in the Table 1, and the function description of words and phrases for oracle-bone inscription as shown in Table 2.

3 Method of Research

By using syntax rules and other related knowledge, syntax analysis transforms the linear order words of a sentence into a structure of grammar tree, which has two main functions:

1. To confirm whether a sentence conforms with grammar;
2. To regularize the syntactic structure.

Because syntax analysis is one of the important and difficult problems in natural language process (NLP), scholars put forward many valid methods in this area, which can be classified as follows [4]:

1. Based on rules: Phrase structure grammar by Chomsky is the representative of the method which sets up a series of syntax rules by unrelated grammar with context, and then analyzes sentences in virtue of such rules. The main representations of its analysis methods are top-down algorithm, Tomita algorithm and chart algorithm, etc.
2. Based on corpus: With the fast development of corpus linguistics, the statistical methods come to be a hot issue. There are several statistical models usually used in such methodology: PCFG model, history-based model, cascading parsing model and head-driven parsing model.
3. Combining rules and corpus: For oracle-bone inscription language, combining rules and statistical methods have many benefits and advantage. Statistics is a better way to collect rules, which could be used to analyze syntax. In this way, comparatively better results could be attained in the closing test for oracle-bone inscription corpus. Furthermore, the

grammar and syntax phenomena in the oracle-bone inscription language are quite focused. Therefore, it is easy and accurate to obtain rules for syntax analysis.

3.1 Rule classification and formalization

In our research, the rules of dependency grammar were divided into context free rules and context sensitive forbidden rules. A Context Free Grammar (CFG) rule can be formalized as:

$$R_b =< A, B, FLAG > \tag{1}$$

Where A and B are POS tags of two neighboring words. $FLAG \in \{BACK, FRONT\}$ expresses the dependence direction. Its value is either backward (i.e., B is the governor) or forward (i.e., A is the governor).

But context-free POS rules alone are rather restrictive for parsing. Many disagreements can be found in a large-scale corpus. One disagreement is the dependence direction of A and B in the actual text is opposite to the direction specified in a POS rule. e.g., the POS rule $< vg, ng, FRONT >$ is learned from the phrase  (to look at river). The rule indicates that  (river) is the object of (look at). Afterwards, if we apply the rule to parse the phrase  (timely rain), an error would appear,  (timely) is usually used as a verb, but here (timely) modifies
 (rain), thus (rain) is the governor instead. This type of ambiguity is referred to as primary ambiguity.

Another disagreement is that two independent tags, A and B in a sentence, are mistakenly identified by a related context free rule. For example, the rule $< a, ng, BACK >$ is valid in general causes, but in the phrase  , the word does not modify , it relates to the preceded verb by serving as an adverb. That is to say, the rule $< a, ng, BACK >$ does not apply to the context that "there is a verb before the adjective." This kind of ambiguity is referred as second-class ambiguity. The rules of agreement are non-ambiguous rules.

Ambiguous rules arose due to inadequate contextual knowledge or semantic information. For oracle-bone inscription, we can use the contextual knowledge to generate context sensitive rules which are referred as forbidden rules and formalized as follows:

$$R_f =< A, B, FLAG, E >, \quad E = LOC_1 \quad C \tag{2}$$

Where R_f expresses in the context environment E, the rule $< A, B, FLAG >$ must be forbidden.

Assume that two words corresponding to A and B are w_1 and w_2. The sequence of the sentence where they appeared to be $\cdots w_{l_2} w_{l_1} w_1 w_2 w_{r_1} w_{r_2} \cdots$, and POS tag of w_i is $c_i \quad (i \cdots, l_2, l_1, 1, 2, r_1, r_2, \cdots)$. In an observed example,

if there is an element $< c_1, c_2, FRONT >$ in R_b, the governor of w_2 is not w1 but w_n, then it can be defined as follows:

1. $n \geq r_1$: the value of E is "$+C$", it expresses the actual governor of w_2 is behind w_2, C is the POS tag of w_n.
2. $n \leq l_1$: the value of E is "$-C$", it expresses the actual governor of w_2 is ahead of w_1, C is the POS tag of w_n.

If w_n is a function word, then the information of governor of w_n (w_m) needed to be added. At this moment, we added another expression of E:

$$E = LOC_1 \quad C \quad LOC_2 \quad D \tag{3}$$

Where LOC_1, $LOC_2 \in \{+, -\}$. LOC_1 and C are defined in Eq.(2), D is the POS tag of w_m. When $m > n$, then $LOC_2 =$ "$+$", else $LOC_2 =$ "$-$". When $FLAG = BACK$, we can define the rules like upwards too.

3.2 Algorithm 1: Automatic learning rule sets

The symbol explanation: assumed that the context-free rule set B, the non-ambiguous POS rule set B_c, the ambiguous rule set B_a, F_c as avoided ambiguousness and F_u as ambiguousness in the forbidden rule set.

In the learning process, we also used two other supporting rule sets S and P, which are the forbidden rule set at the POS level and the ambiguous rule set, respectively. The learning algorithm is shown as follows:

1. Let $B = B_c = B_a = F_c = F_u = S = NULL$.
2. By considering the statistics of all edges in a dependency tree, we acquired the CFG rule set B at the POS layer.
3. $B_c = B$.
4. For each element of the set B, i.e., $b_i < c_1, c_2, FLAG >$ (c_1, c_2 express the POS tags and $FLAG$ takes the value of either $FRONT$ or $BACK$), scan all the dependency trees to examine all the neighboring word-pairs (w_1, w_2). If it satisfies the form of b_i, but does not satisfy the dependency relationship, get rid of it from B_c and add it to B_a; at the same time add a rule to S , formalize $< c_1, c_2, FLAG, e_i >$.
5. For each element of the set S, i.e., $s_i < c_1, c_2, FLAG, e_i >$, scan all the dependency trees to examine all the neighboring word-pairs (w_1, w_2) and its context environment. If they satisfy the form of s_i, then this word-pair and the context environment were recorded by the rule, dependency relationship in the set A_i were corrected. Apparently, A_i is the set composed of all isomorphic structures of forbidden rule s_i in POS layer.
6. For each element of the set S, i.e., s_i, examine its corresponding element A_i, if the dependence directions don't satisfy the $FLAG$ for all in A_i, it means the context e_i in s_i is non-ambiguous in this case. Record the rule s_i in the forbidden rule set F_c. On the other hand, if the direction is the same as $FLAG$, record it in the set F_u, go to artificial operation or use other method to avoid ambiguousness.
7. Repeat procedures (2)-(6) until sets F_c, F_u, B_c and B_a do not increase.

3.3 Algorithm 2: The error-emended learning

The error-emended learning method is used to capture and optimize of oracle-bone rule-base which characteristic as follows:

First, intuitive oracle-bone inscription rule sets (initial rules) are defined in advance, the rule sets had relatively complete and scientific rules, which had already their formalization description.

Second, the error-emended learning method is different from transformation-based error-driven learning algorithm. The former has not transformation process which artificial proofread occupy the important position, the comparison process seeks only error.

Third, these rules were described by the complex feature sets, the task after proofread is to revise and add to the attribute and feature.

The concrete algorithm is shown as follows:

1. Input text of oracle-bone inscription, word segmentation and POS tag are carried out, then artificial proofread;
2. After word segmentation and POS tag, combined part semantic information with the indicated words and expressions, a sentence of oracle-bone inscription is divided into several linguistic sections, i.e., the picked-up phrases, then artificial proofread linguistic sections.
3. The syntax and semantic analysis are processed for the linguistic sections. For the new rules, POS attribute is added to the analysis results, and the feedback information is given out.
4. Compared the analysed results and the correct results of syntax and semantic structure, these rules are learnt which are preconcerted and closer to their really results. If there are new rules or mistake rules are found, then add and revise these rules, and save the revised results to rule sets. During the learn process of each iteration, those rules of the highest score are learnt and added to the order rule list, the tagged text is updated. Return initial state, input new text and repeat above process.
5. These selected rules to re-tag the text, iterative learning is carried out, the learning algorithm is stopped when new and correct rules can't be found to improve tagged text.

4 Experimental Results of Learning

Experimental data is 367 sentences which were picked out from oracle-bone inscription corpus, word segmentation and POS tag are carried out by the preprocessing program, for example:

1. Word segmentation: 
   POS tag: n/ v/ v/ v/ n / v/ v/ v
2. Word segmentation: 
   POS tag: n / v/ n / n/ d/ v/ p / n/ d/ v
3. Word segmentation: 
   POS tag: v/ p/ n / v/ m/ n

4.1 Experiment for algorithm 1

Then an algorithm of oracle grammar rule learning which mention above is used. The experimental results of syntax learning rules are: 145 bigram POS rules, 87 ambiguous rules and 23 forbidden rules. Word segmentation and POS tagged texts are analysed, summed up and processed statistically. According to multiple-information dictionary and our algorithm, we analyzed the syntax relation layer of oracle-bone inscription, checked up and verified the corpus tagging process. Our experimental data included 367 sentences selected from our corpus-tagging program, of which 341 were analyzed correctly, 26 sentences were analyzed mistakenly, demonstrating the correct rate 92.9%. Because the mistakes of lexical and syntax analysis take up 50% of all kinds of mistakes, the more semantic structure analysis can improve the system performance.

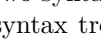
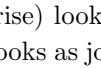
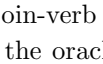
4.2 Experiment for algorithm 2

Combining the result of the error-emended learning and multiple-information dictionary of oracle-bone inscription, we can obtain the results of syntax analyses and parse the oracle-bone inscription sentence. We transform the linearity and ordinal structure of words and phrases into a syntax tree structure.

For example: 

POS: n/ adv/ v / n / v

According to context syntax rules, the sentence can bring several syntax trees, for example, two syntax trees, as shown in Fig.4.

For the former syntax tree,  (watch sunrise) looks as verb+object structure,  (sunrise) looks as predication structure as object. The latter syntax tree,  looks as join-verb structure,  (watch sun), then  (come out), by searching the oracle-bone inscription dictionary, we knew predicate object attribute of verb  is be, it shows that verb (watch) can have a predication structure as object. Therefore, the former syntax tree structure is right,  (that) can't modify join-verb structure, the latter syntax tree structure is a mistake.

Ambiguity is very serious for rules expressed in POS forms. Therefore, it is very important to solve the ambiguity problem of grammar rules. Our experimental results shows that our word segmentation and POS tag process are satisfactory. From an application point of view, our algorithm plays an important role in the relation structure analysis of oracle-bone inscription. With the development of research forming a larger integrated semantic corpus, we can solve the problem better and better.

5 Conclusion

The most hard part of syntax analyses is how to create suitable grammar rules. To solve the problem using the learning rules based on corpus is a

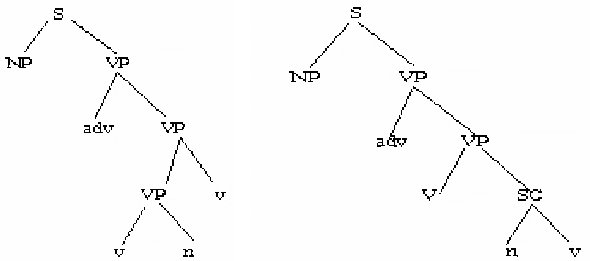


Fig. 4. Two syntax trees for the analysis of

good way, and also, the context environment and the sematic information can solve the ambiguousness of rules.

We proposed two algorithms: automatically learning rule sets and the error-emended learning from the corpus. We combine rules and multiple-information dictionary to parse the oracle-bone inscription sentence which is based on corpus to generate rules, and used the result of learning to parse the oracle-bone inscription sentence. Combining part-of-speech and contextual information together in the learning and expression grammar rules, we obtained an effective algorithm and better syntax analysis results. On the way of the computer information processing of oracle-bone inscription, there are many problems need to be solved.

References

1. Yao, X., Xiao, D.: Collective of Yinxu Oracle-bone inscription. Zhonghua Press, Beijing, 1988
2. Zhang, Y.: Oracle-bone Linguistics in 20 Century. Xuelin Press, Shanghai, 2003
3. Jiang, M., Liao, P., Zhang, B., et al.: Construction on Word-base of Oracle-Bone Inscription and Its Intelligent Repository. In: Proceedings of The 20th International Conference on Computer Processing of Oriental Languages. (2003)272-278
4. Jiang, M., Cai, H., An Application of Natural Language Processing: Oracle-bone Inscription Information Processing. In: Issues for Chinese Information Processing. Scientific Press, Beijing, (2003)415-422
5. Zhang, Y.: Oracle-bone Inscription Linguistics. Xuelin Press, Shanghai, 2001
6. Yuan, C., Chen, G., Wong, K., et al.: An Effective Method for Chinese Grammar Rule Learning. International Journal of Computer Processing of Oriental Languages, 14(2001)361-374
7. Jiang. M., Zhu. X., Gielen. G., et al.: Braille to Print Translations for Chinese. Information and Software Technology. 44(2002)91-100
8. Brill, E.: Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part of Speech Tagging. (1995) http://citeseer.nj.nec.com/brill95 transformationbased.html

Speech Interface for Internet Service "Yellow Pages"

Alexey Karpov and Andrey Ronzhin

Saint-Petersburg Institute for Informatics and Automation of the Russian Academy of Sciences, 14-th line, 39, Saint-Petersburg, 199178, Russia

Abstract. The paper describes new automatic service intended for using telephone directory inquiries without human-operator. We propose to use the automatic speech recognition system for answering the queries of the users. The information about firms and organizations can be taken from the electronic catalogue "Yellow Pages of Saint-Petersburg" located in the Internet and regularly updated with new information. During development such system for Russian language there are sufficient problems, connected with complex structure of word-formation of Russian language. The developed speech recognition system SIRIUS has one additional level of Russian language representation - morphemic level. As a result the size of vocabulary and time for speech processing are significantly decreased. Also the dialogue model for voice control of electronic catalogue "Yellow Pages" is presented in the paper. The first experimental results allow to say about good perspectives of development of telephone speech recognizer for large vocabulary for Russian.

1 Introduction

Infotelecommunication is a new perspective direction in modern science and technique, which presents a symbiosis of information technologies and phone industry (telecommunications) [1]. At present the infotelecommunication market in the world is developing very quickly. One of the most perspective directions in this market is the elaboration of new services and systems, which could maximally use various communication abilities of a human and, first of all, the natural speech. Some examples of perspective applications are [2]:

- automation of operator services
- automation of directory assistance
- voice dialing
- reserve directory assistance
- voice messaging
- voice response systems
- extended banking services, etc

The main difficulty of development of Russian speech recognizer is complexity of Russian language. In contrast to English the Russian language has much more variety on word-form level and so the size of recognized vocabulary sharply increases as well as quality and speed of the processing decreases.

To solve this problem we propose to use original approach based on morphemic analysis of speech and language. This approach uses several new methods and allows to increase the speed of speech processing for Russian without decreasing of speech recognition accuracy.

At present we are realizing the scientific-technical project funded by Saint-Petersburg Government and devoted to introducing the speaker independent system for Russian speech recognition into the electronic catalogue "Yellow pages of Saint-Petersburg". As a result the united automatic information service, which will help to find address and telephone of necessary organization by usual telephone line or mobile phone, will be developed. In this system the speech recognition and speech generation modules will be used for processing the user calls.

2 Specifics of the subject domain "Yellow Pages"

The electronic catalogue "Yellow pages of Saint-Petersburg" is located on web site http://www.yell.ru (see Fig. 1). It is full telephone directory of Saint-Petersburg.

Fig. 1. The electronic catalogue "Yellow pages of Saint-Petersburg"

The catalogue contains the list of all the organizations and companies with reference on address, phones and types of activity. The total number of topics is 1567, the total number of companies is over 20000.

We propose to use the automatic system based on speech recognition instead of human-operator to answer the queries by telephone concerning seeking the information in electronic catalogue Yellow Pages. For work of such

automatic system the vocabulary size for speech recognition has to contain more than 45000 words, including 5000 names of geographic objects of Saint-Petersburg (regions, streets, etc.).

A user can realize two types of requests to the system:

1. The search of an organization by rubricator of activity kinds. In this mode a user should say a name of interesting topic (as example, "Car schools") and the system generates and sounds the list of organizations corresponding to recognized rubric title.
2. The search of an organization by its attributive information. It is title of company, address or phone. In this mode a user should choose the type of search ("title","address" or "telephone number")firstly and then pronounce the interesting text.

The base structure of man-machine dialogue is presented in Fig. 2. A user can enter the known information and the list of organizations satisfying the query are dictated to a user. At first the system informs about the number of organizations in the result set after that a user can specify the query to make more precise search. Also a user can select to dictate whole the list of the organization corresponding to the query ("Dictate all") or the part of this list ("Dictate first N"). In our research work for dictation of the answers SAPI Text-To-Speech Engine DIGALO for Russian is used.

The developed system works on system-centered dialogue i.e. the automatic system begins the dialogue and gives the hints to the user on all the stages of dialogue. For the answering to a user the system can use speech synthesis or generation of speech answers from pre-recorded fragments. In the next section the main modules of speech recognition system as well as the process of database preparation will be considered.

3 Structure of SIRIUS system

The main aim of the research is the development of computer model of voice interface, which provides speaker independent input of Russian speech. The developed software complex and databases will provide the base for future development of the model of recognition of Russian speech with large vocabulary.

The architecture of automatic system SIRIUS (**S**PIIRAS **I**nterface for **R**ecognition and **I**ntegral **U**nderstanding of **S**peech), developed in Speech Informatics Group of SPIIRAS, is presented in Fig. 3.

In contrast to English the Russian language has much more variety on word-form level and so the size of recognized vocabulary sharply increases as well as quality and speed of the processing decrease. Moreover usage of syntactic constraints leads to that the errors of declensional endings cause the recognition error of the whole pronounced phrase.

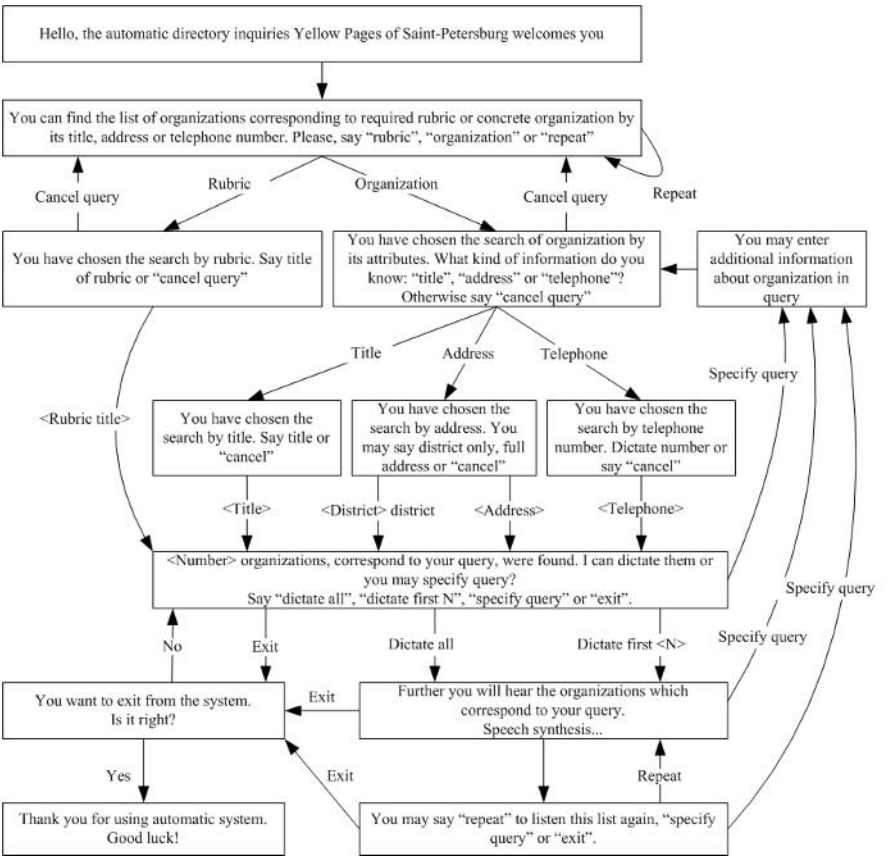


Fig. 2. The base structure of dialogue

To avoid these problems the additional level of speech representation (morphemic level) was inserted [3]. Owing to division of word-form into morphemes the vocabulary size of recognized lexical units is significantly decreased, since during the process of word formation the same morphemes are often used. For creation of creation of morphemes database we used the basic vocabulary of Russian language with the size over 160000 words [4]. The databases of various types of morphemes (prefix, root, interfix, suffix, ending) and automatic methods of text processing were developed on the base of the rules of Russian word-formation [5]. It has allowed to reduce the size of vocabulary of base lexical units in several orders. The co-ordination of morphemes is calculated using bigram statistics based in text corpuses from applied area. As a result of such processing the speed of recognition and robustness to syntactical deviations in the pronounced phrase were significantly improved.

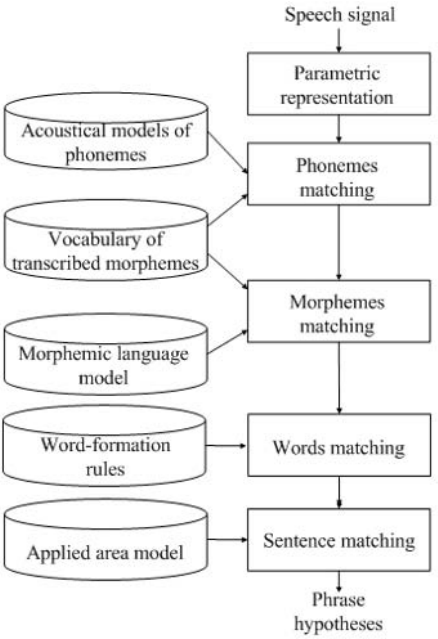


Fig. 3. The base structure of dialogue

The speech signal captured by microphone firstly passes the stage of parametric representation, where starting and ending pauses in the signal are eliminated but the residual part is encoded into the sequence of feature vectors. The parameterized speech signal follows to the module of phoneme recognition. The recognition of phonemes (triphones are used in last version) and formation of morphemes are based on methods of Hidden Markov Models (HMM). As acoustical models we used HMM with mixture Gaussian probability density functions [6]. For feature extraction we used the mel-cepstral coefficients with first and second derivatives. The phonemes are used as triphones (the phoneme in some phonetic context). HMM of triphones have 3 meaningful states (and 2 "hollow" states intended for concatenation of triphones in the models of morphemes).

The process of HMM training is performed by means of consecutive fulfillment of the following procedures:

- Initialization of HMM of phonemes
- Training HMM of phonemes based on training speech databases (Viterbi algorithm)
- Transition from the models of phonemes to the models of triphones and their initialization
- Training HMM of triphones based on training speech databases (Viterbi algorithm)

- Tying the similar HMM of triphones which have little training data
- Consecutive increasing the number of Gaussians in models with simultaneous training based on speech databases.

As phonetic alphabet we use the modified International phonetic alphabet SAMPA. In our variant there are 48 phonemes: 12 for vowels (taking into account stressed vowels) and 36 for consonants (taking into account hard consonants and soft consonants).

In contrast to existent analogues our model uses the morphemes instead of words. Due to this change the recognition of lexical units was significantly accelerated. At that in comparison with whole word recognition the accuracy of morpheme recognition was slightly degreased but due to the following levels of processing the accuracy of phrase recognition was not changed practically.

After phoneme recognition and matching most probable sequences the obtained set of hypotheses is used for synthesis of word sequences. The process of word synthesis from different types of morphemes is realized by the scheme presented in Fig. 4.

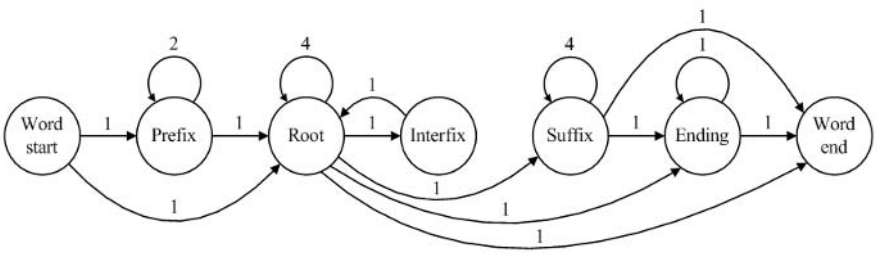


Fig. 4. Synthesis of word from different types of morphemes

The starting and ending nodes are given in this model. Other nodes present different types of morphemes. The arcs denote possible transitions. In future this model would be stochastic but now the maximal number of transition from node to node is given inflexibly. Every phrase hypothesis presented by morpheme sequence and followed into this level processing forms another hypotheses presented by sequence of words.

The last processing level is the sentence matching. The most errors of word recognition in previous levels are connected with the errors in recognition of ending morphemes and prefixes. Thus, for instance, the recognized word-form and said word-form can belong to the same word, but have diverse endings. To solve this problem we use the approach based on dynamic warping of sentences. The objective of the approach is the search of the optimal matching for two sentences A and B (recognized phrase and reference phrase) which are presented as sequences of characters. The measure D, which allows to define difference degree between characters a and b, is logical comparison of characters (0 - if two compared characters are equal, and 1 - otherwise).

Then the optimal path is searched. The usual recurrent equations are used here [7]. The comparison of recognized phrase and all reference phrases (for instance, rubric titles) gives the optimal reference phrase which correspond to the recognized phrase according to the criteria of minimum of deviation. This phrase is the result of speech recognition.

Additionally to the SIRIUS engine required software and databases were developed: (1) databases of different types of morphemes; (2) software modules for automatic word formation; (3) module for automatic text transcribing; (4) module for morphological parsing and accumulating the statistics of existent pairs of morphemes by text.

The results of experiments with usage of developed voice interface for "Yellow Pages of Saint-Petersburg" are presented in the next section. The presented results describe the experiments done in office environments using headset microphone according to recognition of Russian speech with medium vocabulary and first preliminary results according to telephone speech recognition with small vocabulary.

4 Experimental results

At the preparation of experiments the following materials were used. The size of word vocabulary was 1850 words from the rubricator of the electronic catalogue "Yellow Pages of Saint-Petersburg" and after dividing to morphemes the size of morphemic vocabulary was reduced to 1360. Such insignificant reduction depends on specific of the task. At first such parts of speech as verb, participle, pronoun, which have especially complex structure of word formation, are used seldom. Second, practically all nouns and adjectives are used in nominative only. At the same time, for instance, the application of morphemic analysis to processing the book of M.A. Bulgakov "Master and Margarita" has given the reduction of vocabulary about 8 times (22984 words via 2920 morphemes). Therefore the introduction of morphemic level will play important role at the development of the systems with medium or large vocabulary.

This time only one part of the system was developed. A user can search the list of organizations corresponding to thematic rubric, said by a user. In the system there are 1567 thematic rubrics, which contain 1850 diverse words. To test the developed model the 635 phrases were recorded by 5 male speakers in office environment. The recorded files were used for words-based recognition as well as morphemes-based recognition with following words and phrase synthesis. The experimental results are presented in Table 1.

It can be seen from the Table 1 that the accuracy of word recognition during morphemes-based recognition is slightly decreased in comparison with words-based recognition but using the following levels of processing the accuracy of phrase recognition is not changed almost.

Moreover, the test was performed intended for comparison of performance of morphemes-based recognizer with words-based recognizer. The results are presented in Table 2. Total amount of test phrases in test speech database is 635 phrases (which contain 2574 words).

Table 1. Comparison of recognition accuracy

Speaker	Words-based		Morphemes-based	
	Word rec. accuracy	Phrase rec. accuracy	Word rec. accuracy	Phrase rec. accuracy
1	94	95	82	92
2	90	92	78	90
3	93	94	81	92
4	91	91	79	91
5	93	94	82	91
Aver.	92	93	80	91

It can be seen from the Table 2 that the developed system works in 1.7 times faster than the words-based recognizer that taking into account minimal decrease of accuracy allows speaking about creation of perspective speech recognition system for large vocabulary in the next step of investigation.

Table 2. Comparison of recognition speed

	Words-based recognition	Morphemes-based recognition
Time spent for test speech set	2993 sec.	1740 sec.
Average time for one phrase	4,71 sec.	2,74 sec.
Average time for one word	1,16 sec.	0,67 sec.
Average time for one morpheme	-	0,47 sec.

All these tests were fulfilled using headset microphone in office conditions. The further step will consist in gathering the speech database with real telephone speech and training and testing the developed system using this database. To provide the speaker-independency the voices of several tens of speakers are required to collect. At present we have first results according to

recognition of telephone speech for Russian. As hardware for connection PC with telephone line (Public Switched Telephone Network) we used the internal voice fax-modem 3COM US Robotics and the Telephone API as software for work with telephone device. In future we plan to use for PC-Telephone connection the specialized multi-channel plats Intel Dialogic for analog telephone lines. It was defined experimentally that voice modem, which has special set of voice commands, provides the quality of speech recording comparable with Intel Dialogic devices, however one essential disadvantage of voice modems that they are half-duplex devices in contrast to specialized telecommunication devices like as Dialogic, which are full-duplex.

Also during development of telephone speech recognizer there is essential problem connected with diverse characteristics of used telephone lines and telephone sets. At first the spectral band of telephone line is litimed to 4000 Hz. During experiments the best results were obtained using frequency filter 210-4000 Hz. Also during the telephone calls diverse types and levels of noises can be observed and it is required to filter them to increase the quality of speech recognition. To reduce the influence of static noises, with spectrums, which do not change essentially during time (for instance, white noise or low-frequency noise), the Cepstral Mean Subtraction method was successfully applied. Further also RASTA filter will be used for decreasing the influence of telephone noises.

For testing the quality of work in telephone conditions the dialogue diagram was realized for telephone dialogue without recognition of concrete rubrics or organizations (only control commands, like "specify query", "exit", etc.). In the system there are 20 control commands and the accuracy of recognition of this set is about 97 percents using standard analog telephone line, diverse speakers and diverse telephone sets. This quality is enough to continue this research and realize the telephone speech recognizer for large vocabulary for Russian with further creation of internet service with speech interface.

5 Conclusion

The conducted research is directed to investigation of peculiarities of Russian speech and creation of intellectual services for information retrieval. Now we are developing and testing the voice input system for Yellow pages. In this task the size of vocabulary was significantly decreased due to introduction of the morphemes level of speech representation. The databases of various types of morphemes were elaborated and the statistics of morphemes coordination was obtained. As a result of such processing during automatic speech recognition the invariance to grammatical deviations as well as the speed of recognition of Russian speech and other languages with complex mechanism of word formation are provided.

The future work in this research will consist in gathering the required databases with Russian telephone speech for training the speech recognizer

taking into account diverse speaker voices, telephone sets and quality of analog telephone line; finding the effective methods for decreasing the noises in telephone channel; testing the system with experienced users as well as attraction of unexperienced users for testing the developed speech interface for internet service "Yellow Pages of Saint-Petersburg".

6 Acknowledgements

This work is supported by the Saint-Petersburg Government and Saint-Petersburg Scientific Center, Project PD04-3.17-39 "Russian Voice interface" and Project 04-2.1-50 "Complex for training the acoustical models for speaker-independent Russian speech recognition system", as well financed by "Russian Science Support Foundation".

References

1. Ivanova T.I. (2002). Computer technologies in telephony. Moscow: Eco-Trendz
2. Cox R.V., Kamm C.A., Rabiner L.R., Schroeter J. and Wilpon J.G. (2000). Speech and Language Processing for Next-Millennium Communications Services, In Proceedings of the IEEE, Vol. 88, No. 8, 1314-1337
3. Ronzhin A.L., Karpov A.A. (2004). Implementation of morphemic analysis for Russian speech recognition, In Proc. of 9-th International Conference SPECOM'2004, St. Petersburg: "Anatoliya", 291-296
4. Zaliznjak A.A. (1977). Grammatical dictionary for Russian language, Moscow: Rus. jaz.
5. Russian Grammar (1980). 2 vols, Moscow: Nauka
6. Young S., et al (2000). The HTK Book (v3.0), Cambridge University Engineering Department
7. Kosarev Yu.A., Lee I.V., Ronzhin A.L., Skidanov E.A., Savage J. (2002). Survey of the approaches to speech and text understanding. SPIIRAS Proceedings, Issue 1, Vol. 2. - St. Petersburg, SPIIRAS, 2002, 157-195

Automatic Information Classifier Using Rhetorical Structure Theory

Hassan Mathkour, Ameur Touir, and Waleed Al-Sanie

Department of Computer Science, King Saud University,
Kingdom of Saudi Arabia

Abstract. Information classification is aimed to secure the documents from being disclosed. The information is classified according to their critical semantic. The decision of classifying a portion of the document as a 'secret' depends on the effect of its disclose in the organization the document written for. However, understanding the semantic of the document is not an easy task. The rhetorical structure theory (RST) is one of the leading theories aimed for this reason. In this paper, we will explain a technique to classify the information using RST.

1 Introduction

1.1 Information Classification

The information could be classified according to their importance [8], or their potential effect on a specific organization. The classification divides the information in a certain document(s) to one of the following classes:

- *Top secret* – The compromise of this information or material would be likely to cause a tremendouse effect on the organization.
- *Secret* – The compromise of this information or material would be likely to cause a big effect (but less than the above class) on the organization.
- *Confidential* – A private information which should not be disclosed by a person other than the intended one.

The term "unclassified" was not formally a classification but could be used to indicate positively that information or material did not carry a classification.

Companies have documents that carry private information. They need to label them with one of the lables descibed above to tell the reciepiant about their importance. Classifying the information on a certain document depends mainly on what each classified portion is talking about. If a certain part of the document is talking about something that is considered to be a companys privacy, then this part could be classified as secret. For example, part of the document talks about the salaries of the executive board members, in some cases this information is secret and should not be disclosed by anyone. Well, the classification takes into acount the main idea behind the document, and the message behind each part of it. In the same organization, certain

information could be considered as a top secret in a document talking about a certain subject, while it may be considered as unclassified in another document talking about another subject. If the salaries of the executive board members in a company, for example, are mentioned in a document talking about the richest people in the city, it might be considered as unclassified information. Whereas, the same information might be classified as a secret in a document of the financial analysis of the comopany. So it is a matter of what information appeared in what document. The best calssification is done on the semantic of the information not on the word appeared in it; certain word or sentence may have different meaning in different contexts. Because of this, the classification should work on the semantic of the information. Most likely, the classification is done manually be spme person who follow a certain guide lines. This process may take long time when the document is huge handereds of pages. However, trying to do the classification automatically would be a nice idea to make the process faster and easier. The idea of automating the process has the challenge of understanding the semantic of the information. We will give a technique which could be used to solve the proble.

1.2 Rhetorical Structure Theory

RST was originally developed as part of studies of computer-based text generation [1, 7]. A team at Information Science Institute (Bill Mann, Sandy Thompson and Christian Matthiessen) developed this theory to serve as a discourse structure in the computational linguistic field.

RST offers an explanation of the coherence of the text [3]. It is intended to describe texts, rather than the process of creating or reading and understanding them. The rhetorical relations can be described functionally in term of the writer purposes and the writer assumptions about the reader. These rhetorical relations hold between two adjacent spans of texts (there are some exceptions). The output of applying the rhetorical structure theory to a text is a tree structure that organizes the text based on the rhetorical relations [3]. This structure is called the rhetorical schema. Each relation connecting two spans of a text might be one of two cases: one of the two spans is more important to the reader than the other and it represents the semantic of the two spans, the other case is that the two spans have the same importance to the reader. In the first case, the important span is called the nucleus and the other span is called satellite. The other case is called multinuclear relation where both spans are considered nucleus. The process of parsing the text and building the rhetorical structure is called the rhetorical analysis. During the process of the rhetorical analysis the elementary units that participate in building the rhetorical schema are determined, and then the rhetorical relations that hold among these units are determined to connect each two spans. Determining the potential relations that connects each two spans could be done using several techniques; one of which is determining the rhetorical relations through the cue phrases [5, 6]. In [4] Marcu has given many cue phrases

that can be used in the English language processing. The process of building the rhetorical structure may lead to more than one structure which is a consequence of the nature of the natural language text, that more than one relation could be assumed to connect two spans. At the end, the emergent structures are most likely closed, but in some cases, they may lead to ambiguity.

2 Automatic Classifier Concept

The rhetorical structure that is built from the rhetorical analysis process is mainly represented as a binary tree that connects between two spans of a text [3]. Each node of the tree has a status which has the value of either nucleus or satellite, a promotion which represents the most important text unit in this sub tree, and the type which represents the relation the connects the two spans [3]. The rhetorical structure tree could be built using the algorithm proposed in [5, 6]. Marcu stated that the promotion of the root of the whole tree gives the reader the most important unit(s) in the text [2, 4]. However, the classifier can use this fact to determine if a certain portion of the document is a secret or not. Looking at the promotion, the classifier may look if it talks about something that belongs to the first class, this information could be easily stated as a secret, and so on. Notice that, the promotion could be only one unit; therefore, the classifier may go levels down if it wants to cover more units. It depends on the importance of the information to determine to which level the classifier should go for.

We propose a technique that could be used to classify the different parts of a certain document. The technique parses each paragraph in the document and builds the rhetorical tree that represents its structure. Then it looks what each paragraph talks about through looking to the promotion of the root of the tree. It uses the promotion to determine if this paragraph is mainly talks about the subject that the user instructs the classifier to label it under a certain class. Well, the technique could be adopted to work on pages or sections rather than paragraphs. It is a matter of study to determine which text unit the technique could be applied on. The technique also could be adopted to go deeper to lower levels in the rhetorical tree to determine to which class this part of the text belongs to. However, the main idea is to extract the promotion of the text unit you want the classifier to work on and look what it mainly talks about. We have done a simple experiment to test this technique. The following section explains this experiment in detail.

3 Experiment

We have done a small experiment on the Arabic text taking into account that the classifier works on paragraphs, that it looks what each paragraph talks

about and then determine which class it belongs to. The cue phrases and the relations they determine were settled by (Table1).

Table 1. Cue Phrases used in the experiment.

Cue Phrase	Relation
و	Joint
حيث	Elaboration
لذا	Evidence
بالرغم	Concession
لكن	Contrast
على أن	Condition
بما أن	Background

Some cue phrases links tow spans without caring of the position these cue phrases appear in. Cue phrases 1, and 5 trigger a multinuclear relation in which both participating spans are nucleus; these two cue phrases always join the span that comes before them and the span comes after them. Meanwhile, the other cue phrases have two cases. They may join the span comes before them and the span comes after them, or the may join the span comes after them and the span comes after this span the two next adjacent span; however, this depends on the token comes before the cue phrase, if the token is dot (”.”), new line (”\n”), or nothing (first sentence in the document) the second case is applied, first case is applied otherwise. The following example explains the two cases with the cue phrase (بما أن):

[بما أن الشركة وقعت العقد،][1] [فإن العمل في المشروع سيبدأ بعد شهر من الآن.][2]

In the above example, the second case of linking the two spans is applied, in which the two consecutive spans that come after the cue phrase are linked. Let’s now see an example of the first case in which the cue phrase comes between the two spans it links:

[ستتم دراسة جميع المشاريع المطروحة][1] [بما أن العمل في المشروع السابق قد أوشك على الانتهاء.][2]

In the two examples above, the nucleus and satellite positions differ. In the first example, span (1) is the satellite and span (2) in the nucleus. In the second example, the nucleus and satellite exchange the places –span (1) is the nucleus and span (2) is the satellite. Therefore, the relations that represent the two examples are:

rhet_rel (Background, 1, 2)
rhet_rel (Background, 2, 1)

The following two tables (Table 2 and 3) show the cue phrases in the two cases, and the satellite and nucleus positions in each case.

Table 2. Cue phrases that can come in the middle of a paragraph.

Cue Phrase	First Span	Second Span
و	Nucleus	Nucleus
حيث	Nucleus	Satellite
لذا	Satellite	Nucleus
بالرغم	Nucleus	Satellite
لكن	Nucleus	Nucleus
على أن	Nucleus	Satellite
بما أن	Nucleus	Satellite

Table 3. Cue phrases that can come at the beginning of the paragraph.

Cue Phrase	First Span	Second Span
حيث	Satellite	Nucleus
بالرغم	Satellite	Nucleus
على أن	Satellite	Nucleus
بما أن	Satellite	Nucleus

In case that a sentence is not joined to any other sentence via a cue phrase this sentence is joined with the one before it through the relation:
rhet_rel (Joint, Second_sentence, First_sentence)

For example, in the following text, the parser will consider the second sentence as another fact which the author joins to the first one:

[التقارير المالية الصادرة من الإدارة المالية تفيد بزيادة المبيعات في هذه السنة.][1]
[هذه الزيادة ناتجة عن الانتعاش الاقتصادي في السوق المحلية][2]

Let's now take an example to see how the classifier works. As we mentioned in the system description, the classifier works on each paragraph and

classify it. Now lets show the process on a certain paragraph. The following is the test text:

[وقعت الشركة في هذه السنة عدة عقود مع بعض شركات التقنية لتزويدها بالتجهيزات اللازمة.][1] [بالرغم من الجهود المبذولة من قبل إدارة التدريب،][2] [الا أن موظفي الشركة لم يصلوا بعد للمستوى المأمول للتعامل مع هذه التقنية،][3] [لكن الموظفين لديهم الدافع للتعلم.][4] [لذا ستقوم الشركة بالتعاقد مع خبير تقني ليستفيد منه الموظفون][5] [بما أن تكلفة التعاقد أقل من تكلفة الدورات التدريبية،][6] [على أن لا تزيد مدة التعاقد على خمس سنوات][7] [و تشمل إقامة بعض الندوات العلمية،][8] [حيث تثري هذه الندوات معلومات الموظفين.][9]

According to the description given about the cue phrases used, the following relations hold in the above paragraph:

rhet_rel (Joint, 2, 1)
rhet_rel (Concession, 2, 3)
rhet_rel (Contrast, 4, 3)
rhet_rel (Evidence, 4, 5)
rhet_rel (Background, 6, 5)
rhet_rel (Evidence, 7, 6)
rhet_rel (Joint, 8, 7)
rhet_rel (Elaboration, 9, 8)

Using the algorithm mentioned in [5, 6], the classifier will build the RST that represents this paragraph. Figure 1 shows a generated tree that the classifier will use to classify this paragraph. The classifier will look at the promotion of the root node and finds that sentence (5) is the most important unit in this paragraph and have the semantic of it. It looks what it is talking about to determine its class. For example, if the user wants to label any information regarding the company contracts with any part (another company, person etc) as secret, this paragraph will be classified as secret since the promotion is telling that the company is going to make a contract with an expert to train its employees which is:

لذا ستقوم الشركة بالتعاقد مع خبير تقني ليستفيد منه الموظفون

If user configures the classifier to classify this information under another class it will do so, otherwise it will make it unclassified.

4 Further Research

A further research could be done to determine the unit of the document (paragraph, page, section... etc) that could be used to build the rhetorical tree and then classify it. The depth level that the classifier goes for in looking for the promotions is a potential subject of future research. The research should state the language that it will work on, since the nature of one language may differ from another one. Techniques applied on certain language might not be applicable (or difficult to apply) in another one.

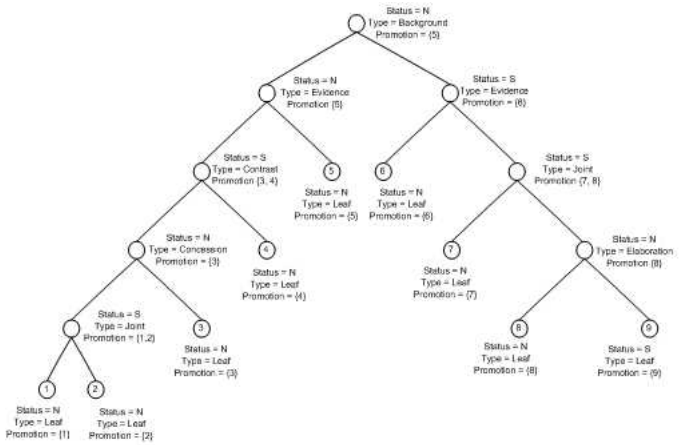


Fig. 1. RST representing the test text.

5 Conclusion

We have given an introduction to the information classification and its usage. We have shown its importance to save the privacy of the organizations. The process of classifying the information might be slow and takes too much effort when the document is huge. Finding a way to do it automatically is a good idea and lead to fast generation of the classified documents. The classification process works on the semantic of the text rather than the syntax; therefore, the technique that should be used to classify the information automatically should take care of the semantic. The rhetorical structure theory (RST) is one of the techniques that try to extract the semantic of the text. We have proposed a classification technique that uses this theory to extract the semantic of the text and then do the classification. We have done a simple experiment on the Arabic text which produced an optimistic result.

References

1. Bill Mann, "An introduction to rhetorical structure theory (RST)", http://www.sil.org/ mannb/rst/rintro99.htm, 1999.
2. Daniel Marcu, "Building Up Rhetorical Structure Trees", The Proceeding of the Flexible Hypertext Workshop of the Eighth ACM International Hypertext Conference, 1997.
3. Daniel Marcu, "Discourse trees are good indicator of importance in text", Advances in Automatic Text Summarization, pp 123-136, The MIT Press, 1999.
4. Daniel Marcu, "From discourse structure to text summaries", The Proceeding of the ACL'97/EACL'97 Workshop on Intelligence Scalable Text Summarization, pp 82-88, Madrid, Spain, 11 July 1997.

5. Daniel Marcu, "The Rhetorical Parsing Summarization, and Generation of Natural Language Texts", PhD Thesis, Department of Computer Science, University of Toronto, December 1997.
6. Daniel Marcu, "The theory and practice of discourse parsing and summarization", The MIT press, 2000.
7. William C. Mann, Sandra A.Thompson, "Rhetorical structure theory: Toward a functional theory of text organization", Text, Vol. 8, No.3, pp.243-281, 1988.
8. W. Nicolls, "Implementing Company Classification Policy with the S/MIME Security Label", The Internet Society, 2002.

Rule-Based Medical Content Extraction and Classification

Agnieszka Mykowiecka, Anna Kupść, Małgorzata Marciniak

Institute of Computer Science, Polish Academy of Sciences,
Ordona 21, 01-237 Warsaw, Poland

Abstract. We present the final version of the system for automatic content extraction from Polish medical data. The system combines general IE techniques with an external post-processing. The obtained data is normalized and linked to a simplified ontology. Then, it is automatically grouped to form more complex structures representing medical reports.

1 Introduction

The paper describes the final version of the system used for automatic content extraction from Polish mammogram reports. The system serves to extract and standardize mammographic data so that it can be stored in a uniform form in a database which will support physicians in decision making and diagnosing.

The extraction process is divided into template extraction using SProUT (a general-purpose IE system, adapted to Polish, see [7]) and external merging of the extracted data into more complex structures, according to our domain model presented in [5]. Three main types of blocks have been identified in the reports: findings, breasts' composition and the overall diagnosis. The merging procedure has been specialized for each part of the report, as different types of information require a separate treatment. The extracted data is normalized and linked to a simplified mammographic ontology. The ontology is based on the general domain knowledge and the analysis of medical reports by a human expert. Our main goal was to capture information contained in the reports rather than to reflect all complexities of the domain.

The organization of the paper is as follows. First, the general system architecture is presented, then, a partial description of adopted ontology is provided and a sample extraction rule is given. Next, merging procedures for specific blocks are outlined and their partial evaluation is presented. The final section contains conclusions.

2 System Architecture

As stated in [6], the system is a pipeline of simpler modules, dedicated to solving specialized tasks: pre-processing, getting partial information by SProUT IE rules, cleaning and merging the results, see Fig. 1. Currently, the merging

procedure has been split into two phases: the annotation of report's main blocks (breast's composition, findings, general diagnosis and recommendations) and then segmenting data in these blocks (e.g., type of dominant and remnant tissue, their localization or concentration).

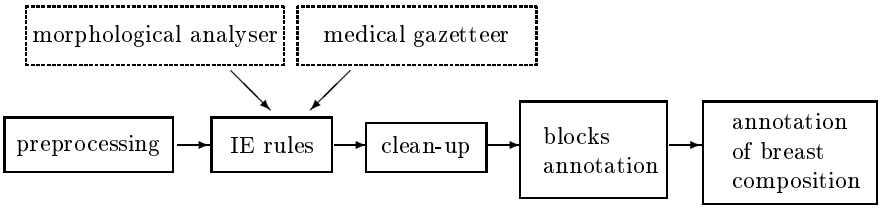


Fig. 1. System architecture

The first two stages (pre-processing and IE) remain essentially the same as reported before ([6], [5]). The clean-up module has been simplified as variable coreference is done automatically with the new version of SProUT. Thus, the module is responsible only for deleting morphological information from output structures, removing duplicate analyses and performing psudo-unification of localizations (see [6] for details). As mentioned above, merging has been currently separated into two phases: first, general blocks are recognized, and then data within these blocks are segmented so that they fully correspond to structures which will be stored in the final database. The latter phase was mostly motivated by data in the breast's composition block. Information about the type of dominant tissue is often interleaved with localization of the remnant tissue and a sequential annotation is impossible. Therefore, an additional procedure is introduced which linearizes and groups the results, see sec. 5.2.

3 Ontology

On the basis of the Polish mammographic ontology developed by T. Podsiadły-Marczykowska, we prepared a simplified model adjusted to our needs, and represented as attribute-value pairs (AVMs). The attributes representing general information about examination, findings and localization have been described in [6]. In Table 1, we present attributes used in the representation of the breast's composition.

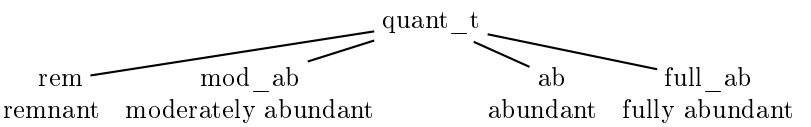
Values of most attributes are the direct subtypes of their most general type and form a shallow type hierarchy; e.g., the value of GLAND|QUANT is of the quant_t type, with subtypes given in (1). For some values, however, it is convenient to specify more complex relations in order to take advantage of strong typing and unification in the current version of SProUT (3.7.2). The hierarchy of tissue types uses this kind of complex constraints, see (2). Names

breasts' description	
BRSURG|SURGERY	undergone surgeries
BRSURG|REASON	the reason of the surgery
BTISSUE	main type of the breast tissue
FIBRE	fibrosis
CHAR	features of breast tissue, e.g., displastic or difficult for evaluation
description of glandular tissue	
GLAND|QUANT	quantity of glandular tissue
GLAND|REGULAR	regularity of glandular tissue
GLAND|DENSITY	density of glandular tissue
GLAND|MACULATION	maculation of glandular tissue
comparison of glandular tissue	
CMP_REG	comparison of regularity
CMP_DENSITY	comparison of density
CMP_QUANT	comparison of quantity
general attributes	
LOC	tissue localization
DIAGNOSIS_RTG	radiological diagnosis of breast's composition
RECOMMENDATION	recommendation

Table 1. Attributes for the representation of breast's composition and their meaning

of tissue types, with the exception of the glandular and fat tissue, consist of two elements: the dominant tissue type (the first element of the name) and the remnant tissue (the remaining part). For example, if the remnant tissue is glandular-fibrosis (gl_fib) and the dominant type is the fat tissue, the resulting tissue is fat_gl_fib.

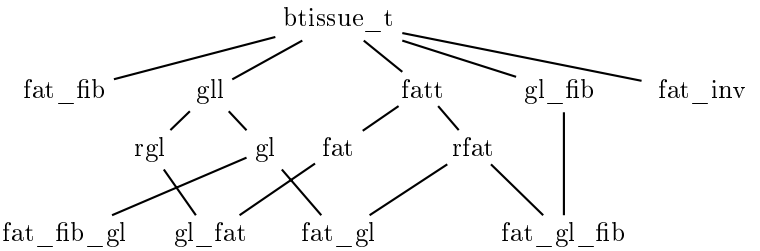
(1)



In (2), the hierarchy of tissue types is presented. The most general type is btissue_t. The only types which cannot be the most specific are the direct subtypes of the glandular (gll) and fat (fatt) tissue, i.e., rgl, gl, rfat, fat. The types rgl and rfat denote dominant tissue types. The other two types represent the remnant tissue. Tissue types can appear directly in the text, e.g., *o tkance tłuszczowo-gruczołowej* 'with fat-glandular tissue' or can be combined from two separate types, e.g., *resztkowa tkanka gruczołowa z przewagą tłuszczowej* 'remnant glandular tissue with dominating fat tissue'. In both cases, the resulting type is gl_fat. The hierarchy in (2) does not reflect all possible

combinations of tissue types but only those which follow from the domain specification.

(2)



4 Breast Tissue Extraction Rules

The rule in (3) recognizes different phrases describing the main tissue type if the dominant tissue type is stated in the text and there is no information about the remnant type. We decided that the default remnant tissue type is glandular (represented by gl). This decision follows from the general medical knowledge and the analysis of mammogram reports.

(3)

```
t_majority:>
   (@seek(loc) & [LOC #loc])?
   (morph & [POS prep, SURFACE 'o', INFL infl_prep &
                                          [CASE_PREP #c1]]
    @seek(tiss)& [C #c1])?
   (morph & [POS prep, SURFACE 'z', INFL infl_prep &
                                          [CASE_PREP ins]])?
   (morph & [STEM 'przewaga'] | morph & [STEM 'przeważać'])
   (@seek(tiss)& [C gen])?
   (gazetteer & [GTYPE gaz_med_utkanie, G_CONCEPT #typ])
-> btiss_str & [BTISSUE gl & #typ, LOC #loc].
```

The `t_majority` rule[1] captures, among others, the following phrases: *sutki o utkaniu z przewagą tłuszczowego* or *sutki z przewagą tkanki tłuszczowej*, both meaning 'breasts with the dominant fat tissue'. The type of the dominant tissue is recognized by a gazetteer entry (GTYPE *gaz_med_utkanie*), indicating a tissue type with a specific G_CONCEPT value. The words *tłuszczowego* or *tłuszczowej* 'fat' are represented in the gazetteer by the concept fatt (the G_CONCEPT value). The variable `#typ` transports information about the tissue type to the resulting structure, where it is unified with the *gl* type (the default value of the remnant tissue). As the result of unification of these two values, we obtain the *fat_gl* type (see (2)).

[1] Details of the SProUT rules' syntax can be found, e.g., in [7].

5 Automatic Annotation

The IE results are stored in a text file as a sequence of attributes and their values. In order to separate the main blocks (findings, breast's composition and diagnosis) and their components (e.g., if there are several findings, their scope has to be identified), we automatically insert tags indicating beginning/end of each element. The next three subsections present the relevant procedures.

5.1 Findings

In order to separate findings, we insert tags identifying the beginning (zp) and end (zk) of each finding. The algorithm presented in [6] remains unchanged (except for updating attributes' names) and is briefly outlined below.

According to their content, attributes are divided into 4 classes concerning: 1) mammographic findings, 2) breast's composition, 3) an overall report evaluation, and 4) attributes used in descriptions of both findings and breast's composition. The annotation procedure is build around one of the two attributes which unequivocally identify the finding: ANAT_CHANGE (anatomic change) or INTERPRETATION. Starting with the first such attribute, we are trying to cover the maximal part of the report until attributes from a different block appear or attributes unique for the finding block are repeated. The final zp and zk boundaries can be corrected if any localization attributes remain unattached to any other block. Sample annotation results are given in (4).

(4)
```
bp
   EXAM_ID:32742||PATIENT_ID:41590
up
   LOC|BODY_PART:breast||LOC|L_R:left-right
   QUANT:rem
   BTISSUE:gll
   LOC|LOC_CONV:uoq
   BTISSUE:fat_gl
uk
zp
   LOC|BODY_PART:breast||LOC|L_R:left-right
   GRAM_MULT:number||MULT:single
   ANAT_CHANGE:macro||GRAM_MULT:plural
   DIAGNOSIS_RTG:benign
zk
   DIAGNOSIS_RTG:no_susp
           ||LOC_D|BODY_PART:armpit||LOC_D|L_R:left-right
rp
   PREV_EXAM|MONTH:1_||SATURATION_CHANGE:none||SIZE_CHANGE:none
bk ============================================
```

5.2 Breast's Composition

The next annotation step is to divide the breast's composition block (delimited by `up`/`uk` tags) into logical subblocks referring to the AVM attributes presented in (5).

(5)
$$\begin{bmatrix}
\textsc{brsurg}\ \text{brsurg_avm} \\
\ldots \\
\textsc{recommendation}\ \text{recommendation_t} \\
\textsc{main_tissue}\ \begin{bmatrix}\textsc{btissue}\ \text{btissue_t} \\ \textsc{loc}\ \text{loc}\end{bmatrix} \\
\textsc{gland_tissue}\ \langle\ \begin{bmatrix}\textsc{btissue}\ \text{btissue_t} \\ \textsc{loc}\ \text{loc} \\ \textsc{gland_attr}\ \text{gl_attr_avm}\end{bmatrix}, \ldots \rangle \\
\textsc{comparison}\ \langle\ \begin{bmatrix}\textsc{cmp_attr}\ \text{cmp_attr_avm} \\ \textsc{loc}\ \text{loc}\end{bmatrix}, \ldots \rangle
\end{bmatrix}$$

The breast's composition block is further annotated as follows. Lines containing general attributes: SURGERY, REASON, CHAR, DIAGNOSIS_RTG, RECOMMENDATION or EXAM_ID are marked as `info`. Lines containing the LOC attributes are marked as: 1) `log` if there is only general localization information, e.g., breast and lateralization; 2) `lsz` if there are only attributes denoting an anatomic part or a conventional localization, 3) `loc` if both `loc` and `lsz` attributes are present.

A line containing the attribute BTISSUE of type gll or gl_fib is marked `tsz`; for other types the marker is `tog`. If a line contains a GLAND attribute, we mark it as `gland` and CMP_* attributes are tagged with `cmp`.

The breast's composition block is processed from the end, lines marked as `info` are left unchanged and the subblock's closing tag, `utk`, is inserted. Then, we process the block according to one of the following rules: a) lines marked with `log` or `loc` remain unchanged. When it is `lsz`, we look upward for the line marked `log` and copy it after `lsz` (a kind of localization unification). Then we recognize a block of consistent information; for simplicity, we assume that it corresponds to lines marked `tog`, or `tsz`, or `gland` with `tsz`, or `cmp` with `tsz`; b) a line marked `tog` is left unchanged; then we look upward for the nearest `log` and add this localization to the `tog` tissue (unification); c) a block of consistent information is recognized and then the localization with unification are added to it.

Next the subblock's opening tag (`utp`) is inserted and we repeat the operation of determining a subblock.

The breast's composition block in (4) refers to the text: *Sutki o resztkowym utkaniu gruczołowym w kwadrantach górno-zewnętrznych. Przewaga tkanki tłuszczowej.* 'Breasts with the remnant glandular tissue in upper-outer quadrants. The dominant fat tissue.' The results of the algorithm used for the compositon block showed in (4) are presented below:

(6) `LOC|BODY_PART:breast||LOC|L_R:left-right`

```
utp
   GLAND|QUANT:rem
   BTISSUE:gll
   LOC|BODY_PART:breast||LOC|L_R:left-right
   LOC|LOC_CONV:uoq
utk
utp
   LOC|BODY_PART:breast||LOC|L_R:left-right
   BTISSUE:fat_gl
utk
```

5.3 Overall Diagnosis and Reliability

We have used three attributes to provide a compact summary of the report: REPORT_CLASS (for the overall diagnosis), MMG_REL (to indicate how reliable the image is) and REPORT_WITH_FINDINGS (a binary distinction specifying if any findings have been detected). The value of REPORT_CLASS is inferred from component diagnoses (if any) and recommended examinations: the most severe diagnosis is taken for the overall diagnosis unless an oncological consultation is required, which yields the *diag_mal* (malicious) value, or a biopsy is recommended, which results in the *diag_susp* (suspicious) REPORT_CLASS value.

The reliability of the diagnosis depends on the type of the breast's composition. The values of the MMG_REL attribute are assigned as follows: if the breast tissue is very dense or dysplastic, or it is explicitly stated that the image is difficult for evaluation, MMG_REL is *unreliable*. If the fat tissue is dominant, the report is *reliable*; in all other cases, MMG_REL takes the *avg_reliable* value.

6 Evaluation

The evaluation was done on 705 new mammogram reports. The results of the automatic annotation were checked manually. We tagged all places where any feature or block marking was inserted incorrectly, was not inserted or was inserted in a wrong place. Afterwards, we counted all correct, misplaced and incorrect occurrences of all attributes. A selected part of the results is presented in Fig. 2. The main problem observed was the unproper recognition of the end of the composition block. Several times the last localisation attribute was incorrectly attached to a composition block instead of a finding block following it. This was the primary reason for necessity of manual corrections of the blocks boundaries.

Another important issue observed is a crucial role of a clear distinction between two main concepts specifying finding blocks: an anatomical change, which can occur at most once in a finding block, and interpretation, which can appear several times in the block.

Some errors were also caused by incomplete grammar coverage for negation and comparison phenomena. For example, a phrase *a density seen in the previous examination now is not visible* was incorrectly recognized as identifying a finding description. On the other hand, a finding described by the phrase *the biggest of them (was located in...)* was not recognized.

	nb	%
patient records	705	
FINDINGS	338	100
unrecognized findings	17	5.0
incorrectly recognized findings	34	10.1
correctly recognized positions of block beginnings	275	81.4
incorrectly recognized positions of block beginnings	46	13.6
incorrectly recognized positions of block endings	27	8.0
BREAST COMPOSITION SUBBLOCK	968	100.0
incorrectly recognized subblocks	9	1.0
unrecognized subblocks	3	0.3
incorrectly recognized positions of subblock endings	24	2.5
ANAT_CHANGE	276	100.0
correctly recognized	269	97.5
incorrectly recognized	22	0.8
INTERPRETATION	174	100.0
correctly recognized	163	93.7
incorrectly recognized	3	1.7
WITH_CALCIF	28	100.0
correctly recognized	26	92.9
incorrectly recognized	0	0.0
RECOMMENDATION	793	100.0
correctly recognized	789	99.5
incorrectly recognized	11	1.4

Fig. 2. Evaluation of automatically identifying blocks' boundaries/attributes

7 Conclusions

We have presented a system for automatic processing of Polish mammogram reports. In particular, we focused on obtaining information about radiological findings, breast's composition, an overall diagnosis and its reliability. Since the extracted data is normalized, it allows for an easy access to similar cases and can support physicians in diagnosing or comparing the data. The evaluation results are encouraging and indicate that the presented method is quite successful: for the most complex task, i.e., grouping several attributes into blocks, we obtained about 82% accuracy, and even 10% higher for single attributes. We believe that the main problems which caused the accuracy decrease can be relatively easy to eliminate in case of a practical application of the system.

References

1. Busemann S. and Krieger H.-U. Resources and Techniques for Multilingual Information Extraction. In: *Proceedings of LREC 2004, Lisbon, Portugal*, 2004, pp. 1923–1926.
2. Drożdżyński W., Krieger H.-U., Piskorski J., Schäfer U., and Xu F. Shallow Processing with Unification and Typed Feature Structures — Foundations and Applications. In: *German AI Journal KI-Zeitschrift*, 01/04. Gesellschaft für Informatik e.V, 2004.
3. Jain N. L. and Friedman C. Identification of Findings Suspicious for Breast Cancer Based on Natural Language Processing of Mammogram Reports. In: *Proceedings of the American Medical Informatics Association Annual Fall Symposium*, 1997, pp. 829-833.
4. Hahn U., Romacker M., and Schultz S. MEDSYNDIKATE — a natural language system for the extraction of medical information from findings reports. In: *International Journal of Medical Informatics*, 2002, pp. 63-74.
5. Kupść A., Marciniak M., Mykowiecka A., Piskorski J., and Podsiadły-Marczykowska T. Information Extraction from Mammogram Reports. In: *KONVENS 2004*, Vienna, Austria, 2004, pp. 113–116.
6. Marciniak M., Mykowiecka A., Kupść A., and Piskorski J. Intelligent Content Extraction from Polish Medical Reports. In: *Proceedings of ICMIT 2004*, Warsaw, Poland, 2004.
7. Piskorski J., Homola P., Marciniak M., Mykowiecka A., Przepiórkowski A., and Woliński M. Information Extraction for Polish using the SProUT Platform. In: *Proceedings of ISMIS 2004, Zakopane*, 2004, pp. 225–236.
8. Ruch P., Baud R., and Geissbruhler A. Evaluating and reducing the effect of data corruption when applying bag of words approaches to medical records. In: *International Journal of Medical Informatics*, 2002.

A Rule-Based Tagger for Polish Based on Genetic Algorithm

Maciej Piasecki and Bartłomiej Gaweł

Wrocław University of Technology, Computer Science Department,
Wybrzeże Wyspiańskiego 27, Wrocław, Poland

Abstract In the paper an approach to the construction of rule-based morpho-syntactic tagger for Polish is proposed. The core of the tagger are modules of rules (classification systems), acquired from the IPI PAN corpus by application of Genetic Algorithms. Each module is specialised in making decisions concerning different parts of a tag (a structure of attributes). The acquired rules are combined with linguistic rules made by hand and memory-based rules acquired also from the corpus. The construction of the tagger and experiments concerning its properties are also presented in the paper.

1 Introduction

The process of *Part of Speech Tagging*, whose task is to identify a contextually proper morpho-syntactic description for each ambiguous word in a text, is one of the basic steps in processing of the natural language. PoS taggers for English are well developed and give practical results with accuracy of tagging close to 99%[1]. A lot of interesting work have been done in case of many other languages, but only a few for Polish. Still accuracy of Polish taggers, e.g. 90.4% in [4], are much worse than English ones and worse than Czech one [9] i.e. 95.16%. The main cause was the lack of large corpus of Polish, and this has been changed since the appearance of the *IPI PAN Corpus* [14].

Most English taggers are based on stochastic methods, e.g. Hidden Markov Models or maximum entropy modelling, see [3]. 'Stochastic' taggers make decisions depending on probability of appearance of the given tag in the sequence of n-surrounding (or only preceding) tags. The number of English tags (100–200) is relatively small in comparison to Polish ones, e.g. more than 1000 in IPI PAN Corpus. This is caused by the rich morphology of Polish, and the necessity of encoding the values of morphosyntactic attributes in tags. The increase causes an even more dramatic increase in the number of possible n-sequences, and makes stochastic modelling demanding much more training data. Moreover, the English syntax presents a lot of 'rigidity' according to the linear order of constituents what results in often fixed order of tags. On the contrary, Polish behaves mostly as a free word order language and

[1] If it is not stated overtly, the accuracy will be given in percentage of words properly described in comparison to all words in a text.

Polish syntactic constructions are often built by far 'ranging' links encoded by morphological features, e.g. [2].

Being familiar with existing stochastic approaches to tagging of Polish [4] and Czech [8], we formulated the goal of our work as an experiment in construction of a tagger for Polish based on a different rule-based approach. The motivations for our work come from: a possibility of encoding long distance dependencies in decision rules, applications of rule-based tagging for Czech, e.g. [12,13], an improvement introduced by the combined approach in [9] and an the work on construction of tagging rules for Polish [15]. Realising the difficulty of discovering tagging rules by hand, we wanted to develop a process of automatic acquisition of rules from the *IPI PAN Corpus*. However, the overall construction of a tagger should allow an easy integration of rules acquired automatically with rules coming from other sources, including rules constructed by hand, like the rules in [15].

2 Rules Discovery

In the classical Brill's approach [1] to rules learning, only a limited context of the given tag is tested in a rule. However, if we want to acquire rules describing long distance dependencies, we need to make the context more flexible, in special cases limited only by the boundaries of a sentence or an utterance. Unfortunately, a flexible context complicates learning process. That is why, we have chosen the general paradigm of *Genetic Algorithms* (henceforth GA) as a tool for rules acquisition.

The acquisition of rules have been based on a subset (up to 582179 words, see the sec. 4) of the annotated part of *IPI PAN Corpus* (shorten to IPIC) and the tagset proposed there. In IPIC, 12 *grammatical categories* have been introduced, e.g. number, case, gender, person, degree, aspect. Each category has been associated with a set of possible values, e.g. accentability with 2 values: *akc* (accented) and *nakc* (non-accented); or case with 7 values: *nom*, *gen*, *dat*, ... A notion of *grammatical class* has been introduced in IPIC, instead of traditional PoS. Simplifying, grammatical classes are defined on the base of a shared set of grammatical categories and a similar syntactic distribution of the members of the given class. The IPIC tag is a list of elements given in an order fixed for the given class. The subsequent positions encode: the class (first) and values of the appropriate categories, e.g. the word *przyczyną* receives the tag[2]: `subst:sg:inst:f`, which means: class *noun* (`subst`), number *singular* (`sg`), case *instrumental* (`inst`) and gender *feminine* (`f`). Because the total number of tag-sequences is huge [3], and the accuracy of a tagger is influenced by the number of tags, we decided to treat the IPIC tag as a *structure of attributes* and to divide the overall problem of choosing the

[2] The original XML encoding of the corpus has been omitted in this paper

[3] There were 1642 different tags, i.e. different sequences, (Przepiórkowski, p.c., 2005) in the part of IPIC used in the experiments presented here, see the sec. 4

right tag into subproblems of choosing values for its attributes (i.e. elements of a tag).

There are three types of rules in our tagger:

1. acquired directly from training examples — *memory-based learning*,
2. discovered by GA,
3. and linguistic — constructed by hand.

The task of hand-made rules is to directly express fixed knowledge delivered by an expert, and to avoid its rediscovery by different methods of rules acquisition. The hand-made rules used in our tagger are a subset of the rules proposed in [15]. Unfortunately, because of their imprecise description in [15], we were able to implement only a part of them. The rules are encoded in a special formal language, and are kept in a separate module. The hand-made rule can make positive decisions — choosing a tag, as well, as negative decisions — eliminating some classes/values of categories from a tag.

Because of problems with achieving the complete coverage by GA rules (discussed in the sec. 4), we introduced the memory-based (MB) rules (the first type) as a supplementary source of knowledge. The MB rules were extracted as sequences of tags of a fixed length (see the sec. 4). The MB rules supplement the two other kinds by 'closing the gaps' in the coverage during tagging, see the sec. 3. The accuracy of each MB rule is assessed during acquisition and stored with it. The accuracy F is calculated as following:

$$F = \frac{|P\&C| - penalty}{|P|} \tag{1}$$

where $|P\&C|$ is the number of learning cases matching the premise and the conclusion of a rule and $|P|$ is the number of cases matching only the conclusion. Here 'matching' means that the given rule is in the *distance* 0 from the case — the computation of distance is discussed below. The *penalty* (here 0.5) in (1) decreases accuracy of too specific rules matching only one case.

The acquired rules are separated into subsets according to the grammatical classes of their conclusions (GA rules also according to the grammatical categories). The separation is a consequence of the assumed division of the tagger into 'subtaggers' specialised in one type of decision.

Rules computed by GA are the core part of the tagger. The GA rules form a kind of a *classification system*, have a general shape: `IF ...THEN ...`, and are learned according to the *Michigan paradigm* [11], i.e. the *population* represents a set of rules, where each *individual* corresponds to a rule and the *adaptation function*, as well, as *genetic operators* are applied to rules (=individuals) and their encodings (=a genotype). There are two types of GA rules differing in actions defined by their conclusions: *positive* and *negative*. Positive rules *choose one* tag from the set of possible ones. Negative rules eliminate one or more of the possible tags (depending on a rule and a context). The negative rules have been introduced to encode (and to discover) a kind

of linguistically necessary constraints that must be preserved by sequences of tags. Moreover, the problem of elimination of a fraction of possibilities seems to be less complex than one-step choice of the right tag. During tagging the negative rules should be applied before the positive rules (regardless of the other types). We hope that in that way the decision problem would be relaxed before the final choice is done by the positive rules.

Because, in our approach, a tag is a structure of attributes, the rules can operate on its parts. This happens on both sides of a rule: the conclusion can affect any part of the resulting tag and the premises can test any part of the tags in context. There are three types of conclusions:

1. a grammatical class, e.g. `IF ...THEN subst`,
2. a grammatical class together with values for selected categories, e.g. `IF ...THEN subst:sg:_:m3`,
3. and only a value of grammatical category, e.g. `IF ...THEN _:sg`

In the examples above, the *empty symbol* '_' can be used instead of any part of a tag, as well, as instead of the whole tag. It makes the substituted attributes/tag meaningless for the given rule.

The premises of the rules are divided into two layers: a layer of *tests*, and a layer of *operators*. The number of elements in both layers is equal to the length of context. The tests check presence of a certain class and/or certain values of selected categories, e.g. (a first layer of a rule):

(R1) `IF BEFORE: _ AND _:sg:n:_ AND qub`
`AFTER: subst:_:gen:_ AND _:gen THEN adj:_:gen:_:_`

The operators of the second layer implement linguistic constraints, that test (generalised) agreement of the given tag with tags located inside, and, (what is even more important) also outside the context. The operators originate from corpus analysis and some of them are just relaxed versions of the rules of Rudolf [15]. The operator definition includes: a *trigger* (set of preconditions) limiting its applicability[4] and a test returning a boolean value. The operator is applied only if its *preconditions* are fulfilled, i.e. it can return three values: true/false and 'not applied' (an 'empty value') e.g. *the operator of case agreement of a sequence after a preposition* (a simplified description):

- *preconditions*: there is a preposition on the left of the given tag and between only adjectival tags and simple adverbs,
- *result*: returns `true` iff there is agreement on *case*, *number* and *geneder* across the whole sequence[5].

The range of an operator is limited to the boundaries of an *utterance* — a sentential segment of the corpus. Each utterance is processed separately.

[4] The basic part of the set is the class of the tag to which it is being applied.
[5] It is one of many operators, that is why is so restrictive, but very informative.

The genotype (encoding a rule) comprises two chromosomes: the first one describing 'standard' premises, i.e. patterns of tags, e.g. see **R1**, and the second one encoding operators attached to subsequent positions in the context, i.e. there can be one operator for each tag in the context, which tests agreement of the tag in a broader context.

The whole set of rules is divided into subsets of similar rules. There are several subsets of rules having a particular grammatical class as conclusion, e.g. the subset of `adjective`-rules. The other subsets concern decision on a value for a particular category, e.g. the *case*-rules. However, the class rules can also choose values for categories (whole tags as conclusions). The division was intended to simplify interpretation of *GA operators* (limiting their work to a smaller subset of rules — individuals) and relax the complexity of learning, and to mimic the work of human annotators (steps in decision). We were aware of relative difficulty in different types of tagging decisions, too [8,4].

In order to fill initial populations (sets of rules) with some rules with non-zero GA *adaptation*, the populations are not set totally randomly and some rules are acquired directly from the training data. Tags corresponding to conclusions are processed according to the type of the given subset, e.g. only class is left non-empty. Operators are randomly selected and attached under the condition they are triggered and return `true`.

We used several GA operators in learning, namely: *selection*, *crossing*, *exchange*, *permutation*, *mutation* and *preselection of de Jong*. Not all of them appeared to be 'helpful'. The usefulness of *crossing* has been questionable from the very beginning. Finally, we limited crossing to the exchange of subsequences of the whole tags between two rules from the same subset. Exchange and permutation are a kind of mutation: the first one exchanges positions of two genes from the premise, the second one permutes positions of all genes. Both operations affect both chromosomes. Mutation works in many ways, but always with respect to the meaning of the tag structure, i.e. it is not a simple change of some bits. Firstly, it can generalise a tag (its premise and/or conclusion) by changing any part of it to the empty value. Secondly, it can modify a tag introducing some value in place of another. Thirdly, there is a special mutation for chromosome of operators. It randomly chooses a new operator depending on the given tag and the grammatical class requested by the operator. An operator can also be changed to the empty value.

The proper performing of selection was a hard problem. The selection chooses individuals (rules), but for us the most important is the work of the whole population, i.e. the tagger. Assuming a very simple method of valuation: a percentage of good decisions in the case of matching premises, the best rules are rules tightly matching particular cases and firing once. As the result, a small set of rules can dominate the whole population. Trying to increase diversity of populations, we have used a technique of *niches*[6]. As the base we assumed the *preselection of de Jong* in the version presented in

[6] But, to be honest, still not achieving the complete coverage of training examples.

[7]. We have tested experimentally several versions of scaling of adaptation of individuals and finally decided to use a technique called *participation in shared examples* inspired by [16]:

$$s(r_m) = \frac{\sum_{i=1}^{k} \frac{1}{\sum_{j=1}^{n} cov(e_i, r_j)}}{k}, \text{ where } cov(e,r) = \begin{cases} 1 \text{ when } d(e,r) = 0 \\ 0 \text{ in other cases} \end{cases} \quad (2)$$

$$f(r_i) = f(r_i) * s(r_i) \quad (3)$$

In (2), r_i means individual (the i-th rule), e_j is the j-th training example, *cov* means *covering*, s is *uniqueness* of the given individual, k is the number of examples matched by r_m, f is the adaptation, and d is the *distance* between a conclusion and a tag in the centre of an example, calculated in learning and tagging in the following way (the simplified description):

- the empty symbol is equal to anything on the given position,
- a difference in classes gives 10 'units', a difference in categories — 1 unit,
- in the case of a different number of elements of two tags, the distance is equal to the smallest distance between the shorter tag, and any subsequence of the longer one.

Several versions of adaptation function were tested experimentally, all based on solutions proposed for GA applications in data mining [16,5]. Finally, the best accuracy have been achieved with the simple function:

$$f_{ad}(r) = f_1(r) * Simp(r), \text{ where } Simp(r) = \begin{cases} penalty \text{ when } cntGen(r) > \eta \\ 1 \qquad\quad \text{in other cases} \end{cases} \quad (4)$$

In (4), f_1 is the function (1), $cntGen(r)$ returns the number of empty symbols used in r, and η is a threshold.

3 Tagging Algorithm

Being concentrated on the process of rules acquisition, we assumed a simple (probably too simple) algorithm of tagging. The tagger applies successively four kinds of rules: linguistic, memory-based, GA negative, and GA positive. In preceding GA rules by MB rules we wanted to use 'remembered cases' before 'discovered knowledge', but when we had changed the distance threshold for MB rules to non-zero one, this order became not so clear. Anyway, we kept it in experiments. The modules of GA rules are applied in a fixed order. The modules of grammatical classes are used first. Next, the modules of categories are applied in the following order: case, number, gender, person, i.e. from more difficult to less frequent.

In tagging, ambiguous words are first marked with the empty symbol, and next according to the possible values of categories and possible classes the appropriate modules are being run in parallel. When more than one class

module is presenting an answer, the adaptation of the rules decides which one to use. The subsequent words are decided in the most simple sequential way i.e. the tagger starts with the first ambiguous word (in the linear order of a corpus utterance) and next proceeds to the subsequent ambiguous words. Decisions for the following words will depend on the preceding ones if they enter the context window of the rules being applied.

4 Experiments and Results

There were two phases of experiments. The first phase was devoted to setting up the parameters realised on a small subset of IPIC including: 59616 words for training and 2923 for testing, where 1617 words were ambiguous. In the second phase, full scale tests were realised using the best values for parameters chosen in the first one. During the full scale, the tests were performed on a larger subset of 578279 words for training and 3900 for testing including 1589 ambiguous words. Here we present only the most important data, the full report is given in [6].

In the first phase, we checked the accuracy of the linguistic rules, and assigned priorities to them. Next, we tested the length of context getting the best results for: 1 + 1 for memory-based rules and 2+1 for GA. The particular technique of niche, and the version of adaptation function have been also chosen experimentally. The best results were achieved with changing GA parameters during learning, i.e. on the start of learning we had: 100% of exchange of population, 0 probability of crossing, and 0.5 of mutation, and, after 6th iteration, we changed them to: 50% of exchange, 0.1 probability of crossing, and 0.3 of mutation. At the beginning of learning we wanted to achieve more generalised rules introducing later 'fine tuning' of them to particular groups of examples. In the first experiments, we proved also that the use of negative rules before positive ones increases accuracy of tagging, and that a class with values for categories is better conclusion than a class used alone. In all experimentes the populations consisted of 300 rules and there were 60 iterations. We tested also the influence of the agreement operators on accuracy of GA rules, but the results were not obvious. One could observe a significant improvement in the best accuracy from 72.2% (positive rules without the operators) to 74.0% (positive rules + operators). But, one could also observe cycles of improvement and rapid descrease in accuracy without any visible tendency of overall improvement in the last 30 iterations. Probably, the cycles are caused by mutation, and the number of iterations is too small for much incresed complexity of learning with operators. Due to long time of learning, the operators have been abandon in the full scale experiments.

In the full scale experiments, we tested separately accuracy of the three kinds of acquired rules on larger corpus, getting: 89.2% for MB rules (73.7% for ambiguous words), 83.4% for positive GA rules (59.5%), and 78.7% for

negative GA rules (48.0%). Next, the three different ways of combinations of different kinds of rules have been tested, too:

- negative + positive — 84.2% (61.5%),
- memory-based + negative + positive — 89.8% (75.1%),
- linguistic + memory-based + negative + positive — **90.0%** (**75.6%**).

5 Conclusions, Problems, and Further Development

The best result of 90.0% is not fully satisfying in comparison to 90.4% in [4], or to 95.16% in [9]. However, if one considers this tagger as the preliminary experimental system built in a half a year, the result will get more value.

The experiment showed that a rule-based tagger for Polish can achieve accuracy comparable with the stochastic approach. It was also visible, that combining several kinds of rules one could get an increased accuracy, higher than any of the sets of rules alone! The positive side of the proposed approach is, that it makes integration of rules made by hand very easy. They work according to the same paradigm like the other two types of rules, thus their influence is predictable. They can make positive, as well, as negative decisions (decreasing complexity of the problem). Moreover, also the GA rules are readable for a human reader and can even be manually corrected.

It is possible, that the final result is decreased in large extent by the simple sequential algorithm of the tagger. It makes each decision only once in the first go. The wrong decision made for a preceding word can mislead the tagger in decisions made for the following words. The overall accuracy of the tagger could be improved by the introduction of some more sophisticated algorithm, e.g. based on a general pattern of Viterbi algorithm used in stochastic taggers or based on the second application of GA. This algorithm should compute the optimal sequence of tagging rules according to their adaptations. The application of rules with higher adaptation (as assessed in learning) should improve significantly the final accuracy (even using the same set of rules). The initial experiment with application of the agreement operators has showed, that they can improve the accuracy. However, their use in GA based learning is problematic. Finally, the better use of MB rules should also be developed.

We used GA as a flexible learning paradigm in face of complexity of the problem, but the results are debatable. It was practically impossible to enforce GA to acquire rules covering the whole range of examples, at least in the Michigan approach to classification system. 300 rules for a module is to little to get a good coverage, but processing more rules would be very costly in time. The best would be to make a whole set of rules being a GA in training directly subtaggers (for classes/categories) not particular rules. However in such a ideal approach, the time of learning would be enormous, i.e. one population for one class/category would be consisted of about 100 sets of thousands of rules each. Encoding of such an individual would request an extremely large chromosome, slowing down the speed of learning very much.

Keeping the overall scheme of rule based approach to tagging of Polish, one need to look for a more efficient algorithm of automatic acquisition of rules.

References

1. Eric Brill. Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study of Part-of-Speech Tagging. *Computational Linguistics*, 21:543, 1995.
2. Magdalena Derwojedowa. *Porządek linearny składników zdania elementarnego w języku polskim*. Dom Wydawniczy Elipsa, Warszawa, 2000.
3. Łukasz Dębowski. Tagowanie i dezambiguacja morfosyntaktyczna. Przegląd metod i oprogramowania. Reports of IPI PAN, Nr 934., 2001.
4. Łukasz Dębowski. Trigram Morphosyntactic Tagger for Polish. In Mieczysław A. Kłopotek, Sławomir T. Wierzchoń, and Krzysztof Trojanowski, editors, *Intelligent Information Processing and Web Mining. Proceedings of IIS:IIPWM'04*, pages 409–413. Springer Verlag, 2004.
5. A. A. Freitas. A Survey of Evolutionary Algorithms for Data Mining and Knowledge Discovery. In A. Ghosh and S. Tsutsui, editors, *Advances in Evolutionary Computation*, pages 819–845. Springer-Verlag, 2002.
6. Bartłomiej Gaweł. Zastosowanie metod programowania genetycznego do oznaczania wyrazów w polskim tekście. Master's thesis, Wydział Informatyki i Zarządzania, Politechnika Wrocławska, Wrocław, 2004. `www.ci.pwr.wroc.pl/~piasecki/pd/gawel\_pd.pdf`.
7. D. E. Goldberg. *Algorytmy genetyczne i ich zastosowania*. WNT, Warszawa, 1998.
8. J. Hajič and B. Hladká. Tagging Inflective Languages: Prediction of Morphological Categories for A Rich, Structured Tagset. In *Proceedings of COLING-ACL Conference*, Montreal, 1998.
9. J. Hajič, P. Krbec, P. Květoň, Karel Oliva, and Vladimir Petkevič. Serial Combination Rules and Statistics: A case study in czech tagging. In *Proceedings of The 39th Annual Meeting of ACL*, 2001.
10. Peter Kosta, Joanna Błaszczak, Jens Frasek, Ljudmila Geist, and Marzena Żygis, editors. *Investigations into Formal Slavic Linguistics: Contributions of FDSL'01*, volume 1. Peter Lang, Berlin, 2003.
11. Zbigniew Michalewicz. *Algorytmy genetyczne + struktury danych = programy ewolucyjne*. Wydawnictwa Naukowo-Techniczne, Warszawa, 1999.
12. Karel Oliva. Linguistics-based POS-Tagging of Czech Disambiguation of *se* as A Test Case. In Kosta et al. [10].
13. Vladimir Petkevič and Karel Oliva. Subject-predicate Agreement and Automatic Morphological Disambiguation of The Czech National Corpus. In Kosta et al. [10].
14. Adam Przepiórkowski. *The IPI PAN Corpus Preliminary Version*. Institute of Computer Science PAS, 2004.
15. Michał Rudolf. *Metody automatycznej analizy korpusu tekstów polskich: pozyskiwanie, wzbogacanie i przetwarzanie informacji lingwistycznych*. PhD thesis, Uniwersytet Warszawski, 2003.
16. Y. Yuan and H. Zhuang. A Genetic Algorithm for Generating Fuzzy Classification Rules. *Fuzzy Sets and Systems*, 84:1–19, 1996.

Part III

Regular Sessions: Search Engines

On Some Clustering Algorithms for Document Maps Creation*

Krzysztof Ciesielski, Michał Dramiński, Mieczysław A. Kłopotek, Mariusz Kujawiak, and Sławomir T. Wierzchoń

Institute of Computer Science, Polish Academy of Sciences,
ul. Ordona 21, 01-237 Warszawa, Poland

Abstract. In this research paper we pinpoint at the need of redesigning of the WebSOM document map creation algorithm. We insist that the SOM clustering should be preceded by identifying major topics of the document collection. Furthermore, the SOM clustering should be preceded by a pre-clustering process resulting in creation of groups of documents with stronger relationships; the groups, not the documents, should be subject of SOM clustering. We propose appropriate algorithms and report on achieved improvements.

1 Introduction

In our research on application of Bayesian networks and artificial immune systems we want to deal with challenging issues of presentation of document collections. Similarly to WebSOM, the goal of the project is to create personal tools for exploration of free text documents by creating a navigational 2-dimensional document map in which geometrical vicinity would reflect conceptual closeness of the documents. We extend, however, WebSOM's goals by a multilingual approach, new forms of geometrical representation and we consider also various modifications to the clustering process itself.

For purposes of soft classification of documents and creation of concept closeness graph being a basis for unsharp similarity measures of documents Bayesian networks will be used [13]. For optimization of parameters of map representation immuno-genetic algorithms are envisaged [15].

Our research targets at creation of a full-fledged search engine (with working name BEATCA) for small collections of documents (up to several millions) capable of representing on-line replies to queries in graphical form on a document map. We follow the general architecture for search engines, where the preparation of documents for retrieval is done by an indexer, which turns the HTML etc. representation of a document into a vector-space model representation, then the map creator is applied, turning the vector-space representation into a form appropriate for on-the-fly map generation, which is then used by the query processor responding to user's queries.

* Research partialy supported under KBN research grant 4 T11C 026 25 "Maps and intelligent navigation in WWW using Bayesian networks and artificial immune systems"

At the heart of the overall process is the issue of clustering documents. Clustering and content labeling is the crucial issue for understanding the two-dimensional map by the user It has been recognized long time ago, that clustering techniques are vital in information retrieval on the Web [12] We started our research with the WebSOM approach which, however, was a bit unsatisfactory for us. Both speed and clustering stability were not very encouraging.

We guess that the basic problem with WebSOM lies in the initialization process of so-called reference vectors, being the centroids of the clusters to grow. In the original WebSOM they are initialized randomly, to be corrected later on in the clustering process. We may be unlucky with the initialization in such a way, because the learning process of WebSOM possesses a "learning speed" parameter α, which may turn out to be too low to ensure convergence for a particular initialization. Another problem lies in the general concept of clustering. In WebSOM, it is tightly coupled with a (non-linear) projection from a multidimensional to a two-dimensional space. Now, there may be infinitely many such projections with equal rights. So one needs really a sense of goal for selecting the appropriate one.

In some other papers we described our dictionary optimization strategies and their speed-up effects to tackle the complexity issue [6]. Dictionary reduction means speed on the one hand, and reduction of arbitrariness of projection on the other. Another research direction was to obtain better clustering via fuzzy-set approach and immune-system-like clustering, described in [7]. So in all, our appproach to document clustering is a multi-stage one:

- clustering for identification of major topics (see [9])
- initial document grouping (see [8])
- WebSOM-like clustering on document groups (see [6])
- fuzzy cell clusters extraction and labelling (see [7])

In an earlier paper [9] we described an approach to major topic identification based on LSI. In the current paper, in section 3 we suggest a naive bayesian network approach, which was a result of our investigation of the behavior of the PLSA algorithm [11].

Earlier, we have either assumed that each document is a document group itself, or we applied simple heuristics (like cosine function) to identify similar documents and to put them into a same small group. In this paper we report on alternative approaches targeting at better clustering stability and quality. Section 4 describes a neural gas approach.

Before we go into details, we outline the basic system architecture for better understanding of the proposed algorithms.

2 System Architecture

The architecture of our system has been designed to allow for experimental analysis of various approaches to document map creation. Software consists

of essentially five types of modules, cooperating via common data structures. The types of modules are: (1) the robot, collecting documents for further processing, (2) indexer, transforming documents into vector space representation, (3) optimizer, transforming the document space dictionary into more concise form, (4) mapper, transforming the vector space representation into a map form (5) search engine, responding to user queries, displaying the document maps in response to user queries. Beside main modules, there are two auxiliary modules: documents clustering module (executed in a preprocessing phase) and cells clustering module (executed right after map creation).

The main data structures, interfacing the modules, are of type: (1) HT Base (Hypertext base), (2) Vector base, (3) map, (4) Base Registry.

A HT Base is the result of a robot activity. We have currently two types of robots, one collecting documents from the local disk space, and another for the Web. A robot collects the hypertext files walking through links connecting them, stores them in a local directory and registers them in an SQL (currently MySQL) database. Standard information like download (update) date and time, original URL, summary (if extractable), document language and the list of links (together with information if already visited) is maintained by the robot.

A HT base can be processed subsequently by an indexer and possibly an optimizer to form a Vector base for the document collection. A Vector Base is a representation of a document space — the space spanned by the words (terms) from the dictionary where the points in space represent documents.

A Vector base is then transformed to a document map by a mapper process. A map is essentially a two-level clustering of documents: there are clusters of documents and clusters of document clusters. Document clusters are assigned a graphical representation of elementary "pixels" in a visual representation, whereas clusters of document clusters are assigned "areas" consisting of "pixels" and are labeled by appropriate phrases.

We use several versions of visualization of document maps. The user can choose to see the data in a square or hexagonal grid. He may also select between a planar representation, or 3D representation on a cylinder, torus or sphere.

Note that the same HT base may be processed by various indexers and optimizers so that out of a single HT Base many Vector bases may arise. Similarly one single Vector base may be processed by diverse mappers to form distinct maps. To keep track of the various descendants of the same HT Base, the Base Registry has been designed. The search engine makes use of all the maps representing the same HT Base choosing the one most appropriate for a given user query.

3 Initial grouping of documents

Initialization of SOM requires several term sets that describe natural groups in our collection of documents. These sets should represent separate topics and contain disjoint terms.

For initial grouping of documents we combined Naive Bayes classifier and EM algorithms. After selection of natural clusters in our documents collection, algorithm selects description of each cluster. To describe clusters it uses terms that come from search engine's dictionary.

Naive Bayes classifier assumes that events are described by set of attributes $a_1, a_2, ..., a_n$ and these attributes have nominal values and are independent from each other. Independency of attributes in our case means that document contains attributes(terms) without any order and presence or absence of each attribute depends on cluster only. These assumptions simplify grouping of huge collection of documents. We need to remember that dictionary which contains 100 000 terms is nothing unusual and each term is related to one attribute.

Given a test set T, we estimate probabilities of particular values of all attributes for each decision concept c (a cluster, in our case). We need to estimate probability of each concept also. Let us assume that there is an event $x' \in X$, Ω denotes some probability distribution and g is a concept identifier. Hypothesis represented by probability distributions evaluated by Naive Bayes classifier is following:

$$h(x') = argmax_g \, P_{x\in\Omega}(c(x) = g) \prod_{i=1}^{n} P_{x\in\Omega}(a_i(x) = a_i(x')|c(x) = g) \quad (1)$$

To group a collection of documents we used EM approach. First of all, we randomly choose initial cluster for each document. Number of clusters K is a parameter that can be set on any positive number. However, we decided on this stage to group documents into 4 clusters. This number of clusters results in a natural map initialization (see also [8]). After random initialization of clusters there is time for estimation of probabilities:

$$\forall g \; P_{x\in\Omega}(c(x) = g) \quad (2)$$

$$\forall g \; P_{x\in\Omega}(a_i(x) = v|c(x) = g) \quad (3)$$

for $i = 1, 2, ..., n$.

Next we evaluate new cluster for each document using equation (1). This process is iterated, after each evaluation of a new cluster for all documents we calculate (2) and (3) for whole document collection and so on. Number of iteration is a parameter but for initial experiments was set to 10.

For clustering phase each document is represented by a vector. This vector has length equal to the size of the dictionary. Each value in this vector

corresponds to the particular term in BEATCA dictionary. If a given term is present in the document then term's value is set on 1, if it does not — value is 0. For the initial clustering of documents we do not take into consideration frequency of the terms. It allows to calculate only the following probabilities for each term $P_{x\in\Omega}(a_i(x) = 1|c(x) = d)$ and $P_{x\in\Omega}(a_i(x) = 0|c(x) = d)$.

Previously [7], we implemented alternative approach to SOM initialization (PLSA [11]), but we didn't obtain good results using it. PLSA approach has some advantages over Naive Bayes, since it allows to have fuzzy membership in each cluster (sum of membership levels need to be 1) and works on a frequency of terms in documents. However, there is a problem with algorithm convergence. We presume that the direct reason is too high number of degrees of freedom and existence of more than one maximum for the likelihood function. Finally, we decided to use Naive Bayes and crisp (not fuzzy) cluster membership. This approach gives us sharp clusters that are easily processed and stable.

After creation of cluster we need to find terms that describe created groups. To achieve this goal we implemented the following algorithm. For each group we select set of terms that maximizes τ within the group.

$$\tau_{i,g} = \frac{\log(\sum N_{ij,g} * avg(N_{ij,g})}{\sqrt{stddev(N_{ij,g})}} * \frac{|D_{t_i,g}|}{|D_g|} \tag{4}$$

Where $N_{ij} = Freq(t_i, d_j)$ is the number of occurrences of term t_i in document d_j. Index g denotes terms that occur in considered group. Abbreviation avg — means arithmetic average, $stddev$ — means standard deviation, $|D_{t_i,g}|$ — means the number of documents that contain term t_i, and $|D_g|$ — means the number of document in considered group.

After calculation of (4) for each group, we start to add terms to description sets. Some terms can have high $\tau_{i,g}$ for several groups, therefore such term will be chosen for a cluster that maximizes $\tau_{i,g}$. If term was selected as a description for one group, it cannot be selected later for any other group.

4 Growing Neural Gas for Document Clustering

In our previous papers [6–9] we described WebSOM-like approach to document clustering. Another approach to this problem offer the Growing Neural Gas (GNG) networks, proposed in [1].

4.1 Basic GNG approach

Like Kohonen (SOM) networks, GNG can be viewed as topology learning algorithm, i.e. its aim can be summarized as follows: given some collection of high-dimensional data, find a topological structure which closely reflects the topology of the collection. In typical SOM the number of units and topology of the map is predefined. As observed in [1], the choice of SOM structure

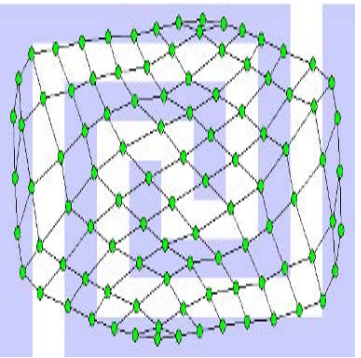


Fig. 1. SOM map for the data distribution which is uniform in the shaded area (output of the DemoGNG applet [4])

is difficult, and the need to define a decay schedule for various features is problematic. A typical SOM produced by the Kohonen algorithm is shown on Figure 1. On the contrary, GNG network does not require specification the size of the network, and the resulting network adapts very well to a given data topology — see Figure 2.

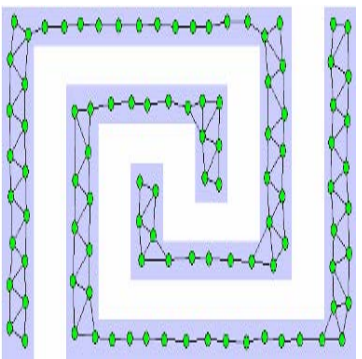


Fig. 2. GNG network for the data distribution which is uniform in the shaded area (output of the DemoGNG applet [4])

GNG starts with very few units[1] and new units are inserted successively every each few iterations. To determine where to insert new units, local error measures are gathered during the adaptation process; new unit is inserted near the unit, which has accumulated maximal error. Interestingly, GNG cells of the GNG network are joined automatically by links, hence as a result a possibly disconnected graph is obtained, and its connected components can

[1] It should be noted here that the initial nodes reference vectors are initialized with our broad topic initialization method.

be treated as different data clusters. Figures 3 and 4 compare SOM and GNG networks for well-known Wisconsin Breast Cancer set consisting of 683 nine-dimensional data points representing two different cancer classes: malignant, and benign.

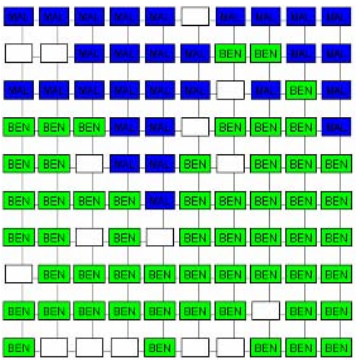


Fig. 3. SOM map for Wisconsin Breast Cancer data set

Both the algorithms discover regions typical for each diagnostic class and they present regions where both classes overlap. The complete GNG algorithm details and its comparison to numerous other soft competitive methods can be found in [2].

4.2 Utility factor

Typical problem in web mining applications is that processed data is constantly changing — some documents disappear or become obsolete, while other enter analysis. All this requires models which are able to adapt its structure quickly in response to non-stationary distribution changes. Thus,

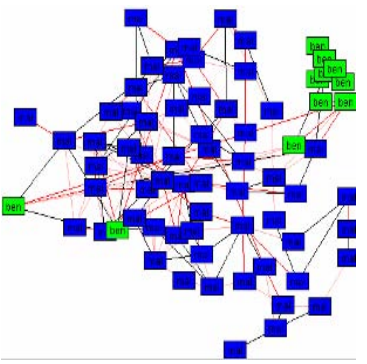


Fig. 4. GNG network for Wisconsin Breast Cancer data set

we decided to implement and use GNG with utility factor model, presented by Fritzke in [3].

A crucial concept here is to identify the least useful nodes and remove them from GNG network, enabling further node insertions in regions where they would be more necessary. The utility factor of each node reflects its contribution to the total classification error reduction. In other words, node utility is proportional to expected error growth if the particular node would have been removed. There are many possible choices for the utility factor. In our implementation, utility update rule of a winning node has been simply defined as

$$U_s = U_s + error_t - error_s$$

where s is the index of the winning node, and t is the index of the second-best node (the one which would become the winner if the actual winning node would be non-existent). Newly inserted node utility is arbitrarily initialized to the mean of two nodes which have accumulated most of the error:

$$U_r = \frac{U_u + U_v}{2}$$

After utility update phase, a node k with the smallest utility is removed if the fraction

$$\frac{error_j}{U_k}$$

is greater then some predefined threshold; where j is the node with the greatest accumulated error. Detailed description of the GNG-U algorithm can be found in [3].

4.3 Robust winner search in GNG network

Similarly to Kohonen algorithm, most computationally demanding part of GNG algorithm is the winner search phase. Unfortunately, neither local-winner search method nor joint-winner search method (which we have developed for SOM learning) are directly applicable to GNG networks. The main reason for this is that GNG nodes graph can be not connected. Thus, basic local-winner search approach would prevent document from shifting between separated components.

A simple modification consist in remembering winning node for more than one connected component of the GNG graph[2] and to conduct in parallel a single local-winner search thread for each component. Obviously, it requires periodical (i.e. once for an iteration) re-computation of connected components, but this is not very expensive[3].

[2] Two winners are just sufficient to overcome the problem of components separation.

[3] In order of $O(V + E)$, where V is the number of nodes and E is the number of connections (graph edges).

Fig. 5. GNG visualization via SOM mapping for Syskill&Webert data set

Another modification is related to the possibility of a node removal. When the previous iteration winning node for a particular document does not exist anymore, we activate global winner search.

4.4 GNG network visualization

Despite many advantages over SOM approach, GNG has one serious drawback: high-dimensional networks cannot be easily visualized. Nevertheless, instead of single documents, we can build Kohonen map on GNG nodes reference vectors, treating each vector as a centroid representing a cluster of documents. Such map is initialized in the same way as underlying GNG network (i.e. with the same broad topics) and next is learned in the usual manner. The resulting map is a visualization of GNG network with the detail level depending on the SOM size (since a single SOM cell can gather more than one GNG node). User can access document content via corresponding GNG node, which in turn can be accessed via SOM node — interface here is similar to the hierarchical SOM map case.

Exemplary map can be seen on Figure 5.

5 Concluding Remarks

In this paper, we have presented new approaches to two stages of our multistage clustering process of document map creation. We are still running experiments confirming improvements both in the map quality and investigating possibilities of the processing speed.

In the near future we would like to replace the naive Bayesian network approach with one based on quick trees, and extend the document group clustering mechanisms to an incremental process based on artificial immune systems.

References

1. B.Fritzke, A growing neural gas network learns topologies, in: G. Tesauro, D.S. Touretzky, and T.K. Leen (Eds.) Advances in Neural Information Processing Systems 7, MIT Press Cambridge, MA, 1995, pp. 625-632.
2. B.Fritzke, Some competitive learning methods, draft available from http://www.neuroinformatik.ruhr-uni-bochum.de/ini/VDM/research/gsn/JavaPaper
3. B.Fritzke, A self-organizing network that can follow non-stationary distributions, in: Proceeding of the International Conference on Artificial Neural Networks '97, Springer, 1997, pp.613-618
4. H.S.Loos, B.Frizke, DemoGNG v.1.5, 1998
5. J.C.Bezdek, S.K. Pal, *Fuzzy Models for Pattern Recognition: Methods that Search for Structures in Data,* IEEE, New York, 1992
6. K. Ciesielski, M. Dramiński, M. Kłopotek, M. Kujawiak, S. Wierzchoń: Architecture for graphical maps of Web contents. Proc. WISIS'2004, Warsaw
7. M. Kłopotek, M. Draminski, K. Ciesielski, M. Kujawiak, S.T. Wierzchon: Mining Document Maps. Proc. W1 — Statistical Approaches to Web Mining (SAWM) of PKDD'04, M. Gori, M. Celi, M. Nanni eds., Pisa, Italy, September 20-24, pp.87-98
8. K. Ciesielski, M. Dramiński, M. Kłopotek, M. Kujawiak, S. Wierzchoń: Mapping document collections in non-standard geometries. B.De Beats, R. De Caluwe, G. de Tre, J. Fodor, J. Kacprzyk, S. Zadrony (eds): Current Issues in Data and Knowledge Engineering Akademicka Oficyna Wydawnicza EXIT Warszawa 2004.. pp.122-132.
9. K. Ciesielski, M. Dramiński, M. Kłopotek, M. Kujawiak, S. Wierzchoń: Clustering medical and biomedical texts — document map based approach. Proc. Sztuczna Inteligencja w Inynierii Biomedycznej SIIB'04, 19.10.2004, Krakw. ISBN-83-919051-5-2
10. D.Dubois, H.Prade, *Fuzzy Sets and Systems. Theory and Applications*, Academic Press, 1980
11. T.Hoffmann, *Probabilistic Latent Semantic Analysis*, in: Proceedings of the 15th Conference on Uncertainty in AI, 1999, pages 289-296
12. M.A.Kłopotek: Intelligent information retrieval on the Web. in: Szczepaniak, Piotr S.; Segovia, Javier; Kacprzyk, Janusz; Zadeh, Lotfi A. (Eds.): (2003) Intelligent Exploration of the Web Springer-Verlag ISBN 3-7908-1529-2, pp. 57-73
13. M.A.Kłopotek: A New Bayesian Tree Learning Method with Reduced Time and Space Complexity. Fundamenta Informaticae, 49(no 4)2002, IOS Press, pp. 349-367
14. K.Lagus, *Text Mining with WebSOM*, PhD Thesis, Helsinki University of Technology, 2000
15. S.T.Wierzchoń: Artificial immune systems. Theory and applications (in Polish). EXIT Academic Publishing House, Warsaw. 2001.
16. R.R.Yager, D.P.Filev, *Approximate clustering via mountain method*, IEEE Trans. on Systems, Man and Cybernetics, 24:1279-1284, 1994
17. C.T.Zahn, *Graph-Theoretical Methods for Detecting and Describing Gestalt Clusters*, IEEE Transactions on Computers, vol. C-20, no.1, 1971

Filtering Web Documents for a Thematic Warehouse Case Study: eDot a Food Risk Data Warehouse (extended)

Amar-Djalil Mezaour

Laboratoire de Recherche en Informatique **(LRI)**, Université Paris Sud
91405 Orsay cedex, France. **email : mezaour@lri.fr**

Abstract. Ordinary sources, like databases and general-pupose document collections, seems to be insufficient and inadequate to scale the needs and the requirements of the new generation of warehouses: thematic data warehouses. Knowing that more and more online thematic data is available, the web can be considered as a useful data source for populating thematic data warehouses. To do so, the warehouse data supplier must be able to filter the heterogeneous web content to keep only the documents corresponding to the warehouse topic. Therefore, building efficient automatic tools to characterize web documents dealing with a given thematic is essential to challenge the warehouse data acquisition issue. In this paper, we present our filtering approach implemented in an automatic tool called *"eDot-Filter"*. This tool is used to filter crawled documents to keep only the documents dealing with food risk. These documents are then stored in a thematic warehouse called ***"eDot"***. Our filtering approach is based on *"WeQueL"*, a declarative web query langage that improves the expressive power of keyword-based queries.

1 Introduction

A thematic warehouse can be a good solution for collecting, organising and storing large amounts of information related to a given thematic. This information can be provided by different and heterogeneous sources like : databases, databanks, document collections etc. Nowadays, The web is the most used source for supplying warehouses with data from web documents. The supplying process consists of a crawler that fetches the web to gather a lot of relevant documents to the warehouse. Due to the huge amount of available information in the web and its poor organisation, a crawler gathers also a lot of irrelevant and useless documents for the warehouse. Finding relevant material from the gathered documents can be a very hard task. To avoid the storage of irrelevant data in the warehouse, efficient and precise filtering tools must be used downstream of the data acquisition process. Two different filtering approaches can be implemented to filter the warehouse candidates documents (i.e., the gathered documents for a warehouse).

The simplest one consists in evaluating thematic keyword-based queries over search engines repositories to obtain a list of urls. These urls correspond to the documents that are more likely to be related to the warehouse topic (according to the search engine query evalutor). We call these documents the relevant documents. Note that, with this filtering approach, all the indexed documents (the relevant documents and the non relevant ones) in the chosen repositories constitute the candidates of the thematic warehouse to implement. These candidates are already known and crawled by the search engine crawler. So, intensive web crawling can be avoided since these documents (the relevant ones) are known and indexed by the existing search engines. The only difficulty consists on expressing the most precise keyword query that best describes the targeted thematic. This filtering approach is used in the Thesus project [1] to populate a thematic data warehouse. The thematic of Thesus deals with technology documents. The authors have manually constructed a set of keyword queries to describe the technology topic. This set of queries has been submitted to Google [2] in order to collect the urls of the documents dealing with technology (relevant documents). This approach is very dependent on the precision of the used search engines and of the quality of the submitted keyword queries. In addition to this, keyword-based queries are not enough expressive to express complex requirements or to query documents with poor textual content. For example, it is not possible to formulate a keyword query that can filter web documents according to their mime type, the fact that they contain tables having a given property...

The second filtering approach consists of achieving the filtering process during the web crawling. To do so, one has to build a specific web crawler suited to the thematic warehouse requirements. Such a crawler traverses the web in a restrictive mode to find the most possible relevant documents to the warehouse without exploring the entire web. This technique is known as Focused crawling [3–5] and is used to populate different topic-specific warehouses such as CORA [6] and CiteSeer [7], on-line scientific papers warehouse. Focused crawling is built upon an intensive learning process that requires an important training set as input. In fact, a classifier is trained to learn a filtering function from a preliminary set of web documents labelled as "relevant" and "irrelevant" documents. This preliminary set is called the training set. To produce a precise and efficient filtering function, the training set must contain enough relevant and representative documents. Constructing such a set leads to the same issue as populating the warehouse itself : how to locate representative documents in the web starting from nothing ! In addition to this, the labelling is often done manually which can be tedious for an important training set.

To summarize, two different approaches can be used to implement a filtering tool for a thematic warehouse: using a list of thematic keywords as

a filtering query over a collection of documents or using machine learning techniques, during or after the web crawl, to recognize the desired relevant documents. Each approach can be viewed as a process that describes and characterizes the warehouse thematic. We strongly believe that an elaborate description and characterisation of the warehouse thematic can be very useful in the filtering process. This is what motivates the use of ***WeQueL*** [8,9] in the implementation of *"eDotFilter"*, the filtering tool of the ***"eDot"*** warehouse. WeQueL is a declarative and multi-criteria web language that improves the expressive power of keywords queries. ***"eDot"*** is a project aiming at building automatic and semi-automatic tools for the implementation of thematic warehouses. The chosen thematic is food risk. The targeted web documents (relevant documents) are defined as documents containing microbiological data for food risk prevention and assessment.

We present in this paper an extension of the preliminary work on filtering web documents presented in [10].

We organised our article into 5 sections. In section 2, we present ***"WeQueL"*** the web query langage that is in use in our filtering process. We introduce in section 3 our filtering approach. The experiments to validate our approach are presented in section 4. Finally, we conclude with the improvements and the on-going work.

2 WeQueL : *Web Query Language*

WeQueL [8,9] is an attribute-value language that allows a user to define its needs by combining several and different keyword-based criteria in one query. Each combined criterion targets its keywords to a specific part of a web document or describes a particular property of a document. For example, the title criterion allows the user to specify a list of keywords to search them in the title of the documents to evaluate whereas the mime criterion restricts the evaluation to the documents having the specified mime type. WeQueL proposes 9 different criteria that are the basic building blocks for complex queries. We call those criteria *"atomic queries"*. An atomic query is an attribute-value like query having the following form:

$$q_a \ : \ \texttt{attribute} \quad \overset{\text{contains}}{\longrightarrow} \quad [\texttt{value}_0, \ldots, \texttt{value}_N] \quad \overset{\text{evaluated by}}{\longrightarrow} \quad \texttt{semantic\_constructor}$$

The attribute of a WeQueL atomic query specifies the part of the document or one document property to target. As mentioned above, WeQueL offers 9 different attributes to focus the search on 9 predefined parts (sections) or properties of a web document:

1. `page_title`: When used in an atomic query, this section targets the search of `value` strings in the title of a document. For an HTML document, we defined the title of a document as the set of words contained between the tags `<title>` and `</title>` or between `<h1>` and `</h1>` or in the value of the attribute `content` of the `<meta name="title">` tag.
2. `mime`: This section restricts the evaluation of the documents to those having their mime type mentioned in the values part of q_a. The values of an atomic query using `mime` section must be valid mime types (like `text/html`, `image/jpeg`, ...)..
3. `title_incoming_page`: This section targets the search of the specified values of q_a in each title of the documents that point to the evaluated document. Suppose that q_a is evaluated over a document d. Suppose that d' is a document pointing to d. Then, the evaluation of q_a over d is equivalent to evaluate q'_a over d' such as q'_a is the atomic query that uses the same values and the same semantic constructor as q_a but targets the title instead of the incoming title.
4. `url`: This section targets the search of `value` strings in the url of each evaluated document.
5. `outgoing_links`: The outgoing links section allows the user to focus the search of the values strings in the words appearing in the neighborhood of the outgoing links of a document. We defined the neighborhood of a link as the words appearing in : its anchor, the tokens of the urls that this link refers to and the 10 words before and after the link (before the `<a href>` tag and after the closing tag `</a>`).
6. `incoming_links`: An atomic query using an incoming links section is evaluated in the same manner as an atomic query using the previous section (outgoing links). The only difference lies in the fact that this section (incomong links) targets the neighborhood of the links pointing to the document to evalute.
7. `keywords`: This section is specific to HTML documents. It is used to target the content of the `content` attribute in the `<meta name = "keywords">` tag (when this last one exists in the document to evaluate).
8. `object`$_{[type='T']}$: This section allows the user to focus the evaluation on the words appearing an HTML object. An HTML object can be a table, a graphic, an applet... For the eDot needs, we handled only tables and lists (i.e., $T \in \{$`'table_or_list'`$\}$). We define a table in an html document as the words appearing between `<table>` and `</table>` tag, between `<dl>` and `</dl>` tag, between `<ul>` and `</ul>` and between `<ol>` and `</ol>`.

The values of a WeQueL atomic query are separated by commas. They are sequentially searched in the content of the targeted section of the document to evaluate. Each value represents a string that contains a single keyword or a list of keywords separated by white space characters. Consider "q_a" an atomic query that targets a given section "s" of web documents. Let's consider also `value="`w_1 w_2 ... w_n`"` one string value specified in q_a. This value

defines an occurrence order that we note: $\texttt{w}_1\texttt{<w}_2\texttt{<}\ldots\texttt{<w}_n$. The search of this string in the section s of a document consists in mapping each $\texttt{w}_i$ with a word $\texttt{w}'_i \in s$ such that :

1. $\texttt{w}_i$ is a substring of $\texttt{w}'_i \in s$. We refer to this inclusion by: $\texttt{w}_i \subseteq \texttt{w}'_i$;
2. the mapped words in s occurs in the same order as in `value` i.e., $\forall i \in \{1, \ldots, n\} : w'_i < w'_{i+1}$.

We note this mapping by μ.
Example: consider `value="food data"` to search in the title of 3 documents:

Document title	value-title mapping (μ)	matching success
`"food database"`	μ(`food`)=`food` $<$ μ(`data`)=`database`	yes
`"data food table"`	μ(`food`)=`food` $\not<$ μ(`data`)=`data`	no
`"on-line book store"`	μ(`food`)=μ(`data`)=∞	no

When all the words contained in a given value are mapped, this value is said *matched over the section "s"*. A word $\texttt{w}' \in s$ is said to be *relevant* when: $\exists$ `value` $\in q_a$, $\exists w \in$ `value`: (`value` is matched) $\wedge$ (μ(`w`) =`w`$'$).

Semantics of a WeQueL atomic query The WeQueL language offers the possibility to determine how to exploit the matched values of an atomic query over a document to estimate the relevance of this document. In fact, the third part of an atomic query is reserved for the specification of the calculation method of a score that represents the relevance of the evaluated document according to the evaluated atomic query. We call this calculation method a "semantic constructor". WeQueL offers two categories of semantic constructors: logical constructors and numerical constructors. A logical constructor is a calculation method that interprets the matched values in a propositional logic setting. The calculated score is then equals to 0 or 1. We implemented 5 logical constructors: `"disjunct"` which assigns 1 to the atomic query score when at least one value is matched, `"conjunct"` which assigns 1 when all the values of q_a are matched, `"atleast(k)"` which assigns 1 when `k` or more values are matched, `"atmost(k)"` and `"exactly(k)"`. Unlike a logical constructor, a numerical constructor assigns a score comprised between 0 and 1. We implemented three numerical constructors. `"precision"` calculates the percentage of the relevant keywords in the targeted section of a document. When this score is high, this means that the evaluated document section contains a lot of the keywords mentioned in the atomic query. `"recall"` calculates the percentage of the matched values of the atomic query. A high `"recall"` score means that the evaluated section of a document covers a lot of the values specified in the query. Finally, `"fscore"` is a combination of the previous scores: $\frac{2\times\texttt{precision}\times\texttt{recall}}{\texttt{precision}+\texttt{recall}}$.

Having the definition of the atomic queries, WeQueL provides a combination formalism that permits the simultaneous (i.e., in the same query)

targeting of different parts and properties of a document. We call this formalism a "WeQueL combination" and we note it $\mathcal{C}$.

$$\mathcal{C} : 1 \cdot \underbrace{q_{\texttt{mime}} \cdot q_a^1 \cdot q_a^2 \ldots q_a^{N-1} \cdot q_a^N}_{\text{1-weighted part}} \; \underbrace{\langle\, \alpha_1 \cdot \widehat{q_a^1} \;, \ldots\ldots, \alpha_M \cdot \widehat{q_a^M} \,\rangle}_{\alpha\text{-weighted part}}$$

A WeQueL combination is an enumeration of weighted atomic queries where: one and only one atomic query must refer to the mime type and there can not be atomic queries duplicates (i.e., a combination can't have two title atomic queries). According to the used semantic constructors and the assigned weights, a WeQueL combination contains two disconnected parts: the part of the 1-weigthed part and the part of the α-weighted part where $\alpha < 1$. The 1-weighted part contains the mime type atomic query and only logical queries. This part expresses the necessary conditions that a document must satisfy (i.e., evaluated to 1) to carry on the evaluation to the next part. The α-weighted part contains logical and/or numerical atomic queries that will determine the final score of each evaluated document. This final score is calculated over a document d by the following formula:

$$score(d, \mathcal{C}) \;=\; score(d, q_{\texttt{mime}}) \times \prod_{i=1}^{N} score(d, q_a^i) \sum_{j=1}^{M} \alpha_j \cdot score(d, \widehat{q_a^j})$$

To avoid the developement of a specific syntactic parser, we defined an equivalent syntax for WeQueL queries based on the XML language. Therefore, we defined an XML-schema that describes how to build a WeQueL query. This schema is available at: `http://www.lri.fr/~mezaour/WeQueLschema.xsd`. Thus, a WeQueL query is an XML document that must conform to this schema.

3 eDotFilter : *the eDot filtering module*

"eDot" is the acronym of *"**E**ntrepôt de **D**onnées **O**uverts sur la **T**oile"* which means *"data warehouses open to the web"*. Four partners are involved in this project : IASI-LRI team (`http://www.lri.fr/iasi/`), GEMO-INRIA project (`http://osage.inria.fr/verso/`), BIA-INAPG team (`http://www.inapg.inra.fr/recherche/`) and Xyleme start-up (`http://www.xyleme.com/`). For more details on the project see `http://www-rocq.inria.fr/~amann/edot/`.

"eDotFilter" module is the filtering tool that determines from a set of crawled candidates documents, the ones to include in the eDot warehouse. To do so, our module uses a WeQueL combination query to estimate the relevance of eDot candidates to the food risk thematic. We call this combination the "filtering query". The filtering query describes, in a declarative way, the content properties of the documents to include in the eDot warehouse (i.e., dealing with food risk). Microbiologist experts defined a food risk

document as an HTML document having atleast one HTML table that contains the most possible keywords from given food risk ontologies. In eDot, we use two food risk ontologies : sym'previus [11] and ComBase [12]. Therefore, we used in the filtering query an atomic query that targets the html tables with all the keywords of the eDot food risk ontology. We also used the set of few relevant document examples given by our experts to enhance the previous atomic query with other different atomic queries. We wrote a java program that parses the content of these relevant examples and suggests in its output the values (strings of two consecutive words) to put in the different atomic queries. For example, our program extracted from the title tag of one of our relevant examples the string: `"fda/cfsan - approximate ph of foods and food products"`. This string is cleaned from its common words (like articles: of, and...) and splitted into a list of 6 suggestions to put in a title atomic query: `fda cfsan, cfsan approximate, approximate ph, ph foods, foods food` and `food products`. We manually discard irrelevant suggestions (`fda cfsan, cfsan approximate, foods food`). The choice of the semantic constructors and the weights to use with each atomic query were determined after discussion with experts according to the eDot relevant document specifications. We finally obtained an advanced WeQueL combination [13] having 8 atomic queries. The second step consists of calculating the relevance score of each eDot candidate document. To do so, we developed in java the WeQueL query engine that parses the HTML content of each candidate and the content of the filtering query xml file to output the evaluation score. The final output is ordered from the highest candidate score to the lowest one. The filtering is then done by setting a threshold relevance and retaining only the candidates that have a score above this threshold. As it is hard to predict this threshold, it is experimentally set.

4 Experimental results

The crawling module of the eDot warehouse gathered from the web 2058 candidate documents as an input to "eDotFilter". The html content of these 2058 candidates were fetched, parsed and evaluated over the filtering query. As a result, we obtained a ranked list of pairs `(candidate_url,score)`. In this experiment we intented to show the robustness of the score ranking achieved by our filtering tool. In other words, are the food risk documents ranked first or not? We intended also to show that the numerical evaluation score of each candidate "d" over the filtering query reflects the effective relevance of "d" to the considered thematic. To do so, we used a `ROC` curve [14] and a `lift` curve to achieve our experimental goals.

A `ROC` curve shows how good and how earlier the relevant documents are covered across the established ranking. The relevant documents are represented in the Y-axis and the non relevant in the X-axis. A `ROC` curve starts

from the first position of the ranking and ends at the last one. At each position "i", the curve grows by one unit if the document ranked at position "i" is relevant. When this document is not relevant, the curve keeps its last value. A ROC curve represents a continuous and increasing function that is bounded by 1. The quality of the ranking system is then mesured by the AUC: Area Under Curve according to this classification table [15]:

ROC curve AUC	Quality
$0.9 < AUC \leq 1.0$	Excellent
$0.8 < AUC \leq 0.9$	Good
$0.7 < AUC \leq 0.8$	Fair
$0.6 < AUC \leq 0.7$	Poor
$0.0 \leq AUC \leq 0.6$	Fail

To construct our ROC and lift curve, we manually labeled the 2058 evaluated documents into two categories A and B: A for "food-risk documents" (relevant documents) and B for "non food-risk documents" (non relevant documents). We obtained: 12 documents for A and 2046 for B. The twelve relevant documents of category A were ranked by our filtering tool among the 40 first positions of the ranking. The 9 first positions of the filtering ranking corresponds to relevant documents (i.e., $\frac{9}{12} = 75\%$ of relevant documents were covered by the 9 first positions of the ranking). So we have two possibilities to choose the minimal threshold that will determine the documents to include in the eDot warehouse: the score of the document at position 9 (to be very precise) or the score of the document at position 40 (to cover all relevants). Knowing that all the relevant documents among the evaluated candidates were highly ranked, we built our validation approach using the 100 first positions of the ranking and we obtained the ROC and lift curves as shown in the diagrams 1 and 2 at page 277.

We can see from the increasing rate of the ROC curve of figure 1 that most of the relevant documents were covered in the first positions of the filtering ranking. The AUC corresponding to this curve is 0.9036 which shows the efficiency of our filtering approach according to the classification given above. After analyzing manually the content of a sample documents, our experts noticed that food risk documents obtained high scores whereas the non relevant ones obtains low scores. This is why the lift curve of figure 2 keeps the precision to 1 for the 9 first positions and then it decreases this precision to the final precision of the 100 sampled documents (i.e., $\frac{12}{100} = 12\%$). This shows experimentally that the numerical score of our filtering approach is a good estimator of the thematic-relevance of a document.

5 Conclusions

In this paper, we have presented "eDotFilter" a filtering tool that uses WeQueL, a multi-criteria web query language that is more expressive than simple

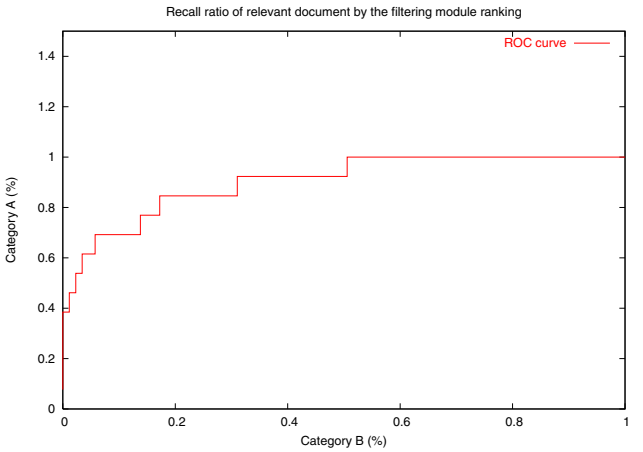


Fig. 1. ROC curve

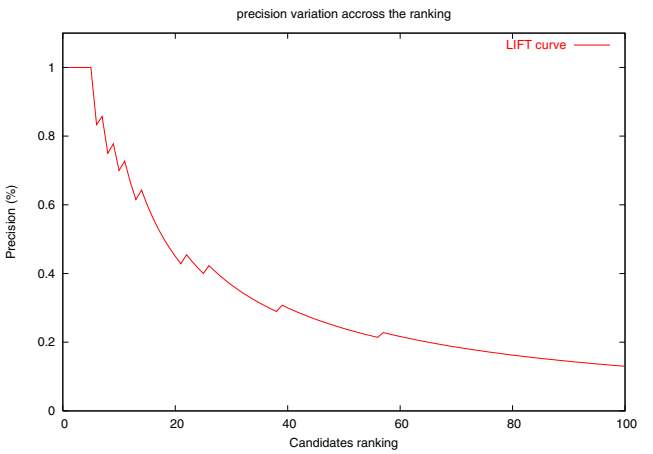


Fig. 2. Lift curve

keywords and less costly than learning the filtering function from relevant examples. This tool is used to filter the crawled candidates to include in the eDot food risk warehouse. We showed in our experiments the benefit of a wequel query having a numerical semantic in the estimation of the relevance of a document to the food risk thematic. 12 relevant documents among 2058 crawled candidates can seem very unsufficient to build a robust warehouse. This is due to the imprecision of the general-purpose crawler used to gather candidates for the eDot warehouse. Therefore, we are developping alternative crawling techniques that combines our filtering approach with a self-learning

technique similar as the one used in the intelligent crawling [16] to fetch the web in a more efficient manner. The intended goal is to learn the topology of the web graph near the thematic documents so that the crawler can focus in a more efficient way its crawling priorities.

References

1. Halkidi, M., Nguyen, B., Varlamis, I., Vazirgiannis, M.: (2002) Organising web documents into thematic subsets using an ontology (THESUS). In: Actes électroniques des Journees Web Semantique, Paris
2. Brin, S., Page, L.: (1998) The anatomy of a large-scale hypertextual Web search engine. Computer Networks and ISDN Systems **30**, 107–117
3. Rennie, J., McCallum, A.K.: (1999) Using reinforcement learning to spider the web efficiently. In: Proc. 16th International Conf. on Machine Learning, Morgan Kaufmann, San Francisco, CA 335–343
4. Diligenti, M., Coetzee, F., Lawrence, S., Giles, C.L., Gori, M.: (2000) Focused crawling using context graphs. In: 26th International Conference on Very Large Databases, VLDB 2000, Cairo, Egypt 527–534
5. Chakrabarti, S., van den Berg, M., Dom, B.: (1999) Focused crawling: a new approach to topic-specific Web resource discovery. Computer Networks (Amsterdam, Netherlands: 1999) **31**, 1623–1640
6. whizbang: Cora version 2.0 : Computer science research paper search engine (website) http://cora.whizbang.com.
7. CiteSeer: http://citeseer.nj.nec.com/cs (website)
8. Mezaour, A.D.: (2003) Focused Search on the Web using WeQueL. In: Proceedings of the 10th International Workshop on Knowledge Representation meets Databases (KRDB 2003), Hamburg, Germany. 63–74
9. Mezaour, A.D.: (2004) Recherche ciblée de documents sur le web. Revue RNTI-E2, numéro spécial EGC'2004 **2**, 491–502
10. Mezaour, A.D.: (2004) Filtering Web Documents for eDot, a food risk warehouse. In: Proceedings of the 2nd International Conference on Computational Intelligence (ICCI 2004), Istanbul, Turkey, Prof. Dr. Ali OKATAN 249–252.
11. Sym'Previus: Système de prévision du comportement des micro-organismes dans les aliments : http://www.symprevius.net/index.htm (website)
12. ComBase: Combined database for predictive microbiology: http://wyndmoor.arserrc.gov/combase/ (website)
13. Mezaour, A.D.: (2004) Numerical filtering query of edot : http://www.lri.fr/∼mezaour/wequel/edot/qnum.xml
14. Ferri, C., Flach, P., Hernandez-Orallo, J.: (2002) Learning decision trees using the area under the ROC curve. In: Proceedings of 9th International Conference on Machine Learning, ICML'02. 139–146
15. Nebraska university, the area under an ROC curve: http://gim.unmc.edu/dxtests/roc3.htm (website)
16. Aggarwal, C.C., Al-Garawi, F., Yu, P.S.: (2001) Intelligent crawling on the world wide web with arbitrary predicates. In: World Wide Web. 96–105

Data Merging in Life Science Data Integration Systems

Tadeusz Pankowski[1,2] and Ela Hunt[3]

[1] Institute of Control and Information Engineering,
Poznań University of Technology, Poland, tadeusz.pankowski@put.poznan.pl
[2] Faculty of Mathematics and Computer Science, Adam Mickiewicz University
[3] Department of Computing Science, University of Glasgow, UK,
ela@dcs.gla.ac.uk

Abstract. An index-driven integration system provides access to a multitude of data sources: it uses pre-compiled indexes covering content of these sources. Such a scenario is especially attractive in life science applications which integrate data from hundreds of very valuable carefully maintained databases. A key bottleneck in building such systems is data merging where partial answers obtained from different data sources are to be merged and the problem of overlapping data should be solved. In response to a query the *most informative* redundancy-free answer should be constructed. In the paper we propose a formal foundation for merging XML-like data and discuss indexing support for data merging.

1 Introduction

Life scientists use information from a variety of web databases to interpret the empirical data obtained in the laboratory. Such data interpretation is hard to achieve without the help of advanced database applications [11,12]. Most existing systems, however, do not really integrate data, but merely present lists of links which may be grouped according to the data source they came from. Such data integration is offered by the SRS system (http://srs.ebi.ac.uk). Deeper data integration involves delivering the data available via the links to the user, and better still, it should include data merging and removal of duplicates.

In this paper we propose a new methodology for data merging which takes advantage of mappings and classifications at the schema and at the instance level, as well as of key constraints discovered via data mining. We emphasise the role of indexes to support efficient execution of data merging operations on very large data sets [9] and to perform reconciliation of ontologies, data mining, and schema matching operations.

The rest of the paper is organised as follows. In Section 2 we discuss a system architecture for data integration. Key discovery for XML data is proposed in Section 3. A subsumption relation between data and some inference rules for subsumptions are defined and proven in Section 4. In Section 5 we describe and illustrate an application of our method to data merging. Section 6 concludes the paper.

2 System architecture for data integration

In Figure 1 we present an architecture of a data integration system. The design takes into account the following facts:

1. We witness an explosion in specialised life science databases. These databases are carefully maintained by specialists in particular domains, so many researchers regard them as extremely valuable [10]. Only in biology do we observe about five hundred databases of this type.

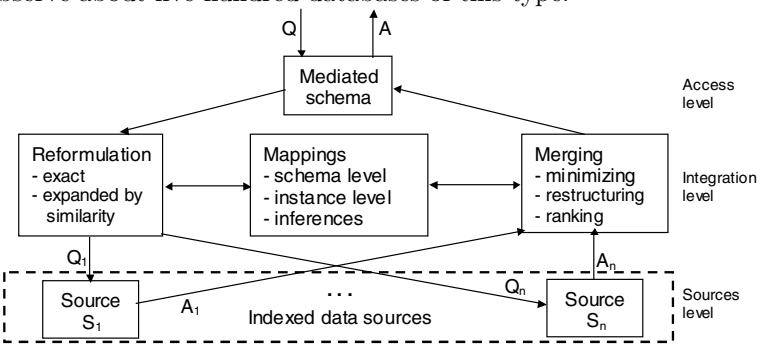


Fig. 1. Overall architecture of index-driven data integration

2. A user submits a query Q as a Boolean combination of keywords. The query must be understood by the system properly, and the system must decide how and where the query should be processed. For example:
   - Query "fge-om": a bioinformatician is looking for references to a functional genomics experiment object model, FGE-OM, i.e an object model for data coming from a new set of experiments carried out on proteins.
   - Query "phage display IGR": a biologist is looking for reports on the use of experimental technology called "phage display" where an AA sequence motif *IGR* was identified. This motif has a particular biological significance.

   In response to the second query three different databases might return the following fragments concerning bibliography references:

```
PIR:
<refinfo refid="A56634">
 <authors>
  <author>Curry, W.J.</author>
  <author>Shaw, C.</author>
 </authors>
 <title>Neuropeptide</title>
</refinfo>
SwissProt:
<citation type="journal article">
 <title>Neuropeptide</title>
 <authorList>
  <person name="Curry W.J." />
  <person name="Shaw C." />
 </authorList>
</citation>
```

```
PubMed:
<Article>
 <ArticleTitle>Neuropeptide</ArticleTitle>
  <AuthorList CompleteYN="Y">
   <Author>
    <LastName>Curry</LastName>
    <ForeName>W J</ForeName>
   </Author>
   <Author>
    <LastName>Shaw</LastName>
    <ForeName>C</ForeName>
   </Author>
 </AuthorList>
</Article>
```

3. In answer to the query, a set of items from different data sources (databases or XML documents) will be obtained. The answers can be syntactically and semantically heterogeneous and overlap each other. Some of them can be exact while others are approximate. In the merging phase, all partial answers are merged, restructured and ranked, and a reconciled result is produced [15]. In performing these operations, a repository of *mappings* is used. Mappings are created and maintained dynamically in the system. To create mappings a lot of methods based on linguistics, ontologies, statistics, data mining and machine learning can be used [6].

3 Discovering keys in XML data

As will be seen later, in our method for XML merging keys play an important role. Two XML trees can be merged if theirs key paths are equivalent, i.e. the trees represent information about the same entity from the application domain of interest. Work on keys has been presented by Buneman *et al.* [4,3]. In [7] Grahne and Zhu discussed a data mining scenario for discovering keys in XML data when its schema is not available. We will adopt these approaches. However, we restrict ourselves to so-called 1-keys, i.e. to keys where the target path is uniquely determined by a single key path rather than by a set of key paths.

A path (or path pattern) conforms to the syntax: $P ::= \epsilon \mid l \mid l/P$, where ϵ is the empty path, and l is a label (element tag or attribute name). If n is a node of a tree T, then $n[\![P]\!]$ denotes the set of nodes in T that are reachable from n by following path P. We shall use $[\![P]\!]$ as an abbreviation for $r[\![P]\!]$, where r is the root node of T. By $n_1 =_v n_2$ we will denote *value equality* between nodes meaning that values assigned to nodes n_1 and n_2 are equal.

Following [4], we assume the following definition of absolute keys:

Definition 1. An absolute key for XML is a pair of paths (Q, P), where Q is a target path and Q/P is a key path. An XML tree T satisfies an absolute key (Q, P), if for any two nodes $n_1, n_2 \in [\![Q]\!]$, if $n_1[\![P]\!] =_v n_2[\![P]\!]$ then $n_1 = n_2$. It means, that a subtree denoted by Q is uniquely identified by the (value of) key path Q/P. □

For example, key expressions `(article,title)` and `(article/author, lname)` might be satisfied by an XML document having elements similar to t_1 depicted in Figure 2, see page 286.

Theoretically, a tree must satisfy a key in 100%. In practice, however, especially in a case when data are taken from heterogeneous sources, such expectations are unrealistic. Instead we are looking for keys that are satisfied in a subset very close to the whole XML tree, and violated only in a very small part of it [7]. Another problem is that some keys appear only a few times, and though they are satisfied in 100% are not very interesting. As measures for keys with respect to an XML tree we define the *support* and *confidence* of the key.

Definition 2. Given an absolute key (Q, P) and an XML tree T, we define:

$$support(Q,P) = \frac{\sum_{n \in [\![Q]\!]} exists(n,P)}{count([\![Q]\!])}$$
$$confidence(Q,P) = \frac{count(\text{value-distinct}([\![Q/P]\!]))}{count([\![Q/P]\!])} \quad (1)$$

$$exists(n,P) = \begin{cases} 1, \text{ if there is a P-branch in the subtree rooted at node } n, \\ 0, \text{ otherwise,} \end{cases}$$

value-distinct(X) is a subset $X' \subseteq X$of nodes having distinct values.

Value of $support(Q,P)$ is a measure of interestingness of the key expression with respect to a tree T, and defines the fraction of all subtrees rooted at nodes from $[\![Q]\!]$ that are candidates for being identified by the key path (i.e. it says how many subtrees of this kind have at least one P-branch); $confidence(Q,P)$ describes how many subtrees of T rooted in nodes from $[\![Q]\!]$ satisfy the condition from Definition 1. A key is *interesting* if its support is greater then a given threshold *min_supp* and the key is *accurate* if its confidence is greater then a given threshold *min_conf*. Interesting and accurate keys we call *candidate keys*.

In our system described in [9], candidates for absolute keys are searched by the following SQL query:

```
select   PathID, (count(distinct L.Value)/count(L.Value))
from     Data D, Leaf L
where    L.LeafID = D.LeafID
group by PathID
```

where `Data` and `Leaf` are among the tables used for representation of XML trees [9], i.e. `Leaf(LeafID,Value)` - represents text contained in the leaves, and `Data(DbID,KeyId,PathID,LeafID,order)` - associates each leaf with its data source, the key of the record the leaf belongs to, the path leading from the root to the leaf, and order of the leaf in the document.

As a result, a set of candidate keys will be obtained. In fact, every candidate for absolute key has the form (Q, ϵ), and using inference rules from [4] we can infer (Q_1, Q_2), if $Q = Q_1/Q_2$. The result can be stored in the table

`CandAbsPath(PathID,Value)`.

Values in the `Value` column are further analyzed (using statistics, taxonomies, thesauri or user provided information), and some rows could be removed from the table while some others – as for example *accession numbers*, identifiers from PubMed, or other commonly used identifiers in life science data sources – are good candidates to remain in the table. Finally, a set of accepted absolute keys will be obtained. This set is represented by the table:

`AbsKeys(AbsPathID,Value)`.

4 Subsumptions on XML data trees

Our focus in data integration is the merging of XML data trees constituting partial answers (obtained from different sources) in such a way that subsuming data (i.e. more general or less informative) will be removed. The aim of a merging phase is to produce a variant of integrated data that is "minimal" in structure but "maximal" with respect to information content.

To discuss the problem more precisely, we will use a simple tree language TL, to express both tree patterns (consisting of element tags and attribute names) and trees (consisting of patterns and values):

$$
\begin{array}{ll}
T ::= P \mid P/(T,...,T) & \textit{(tree patterns)} \\
P ::= l \mid l/P & \textit{(path patterns)} \\
t \ ::= v \mid T{:}v \mid P/(t,...,t) & \textit{(trees)} \\
v \ ::= c \mid (v,...,v) & \textit{(values)}
\end{array}
$$

where l is a node label, c is a constant string value, and "/" denotes the XPath axis `child`. Note, that a tree pattern is a set of path patterns with a common prefix.

To define semantics for TL, we will use the approach used in description logic [1]. Let Δ be a non-empty set of *individuals*, and $\texttt{child} \subseteq \Delta \times \Delta$ be a transitively closed binary relation over Δ. *Interpretation* of TL is a function $\cdot^{\mathcal{I}}$ defined as follows:

$$
\begin{array}{ll}
c^{\mathcal{I}} & \subseteq \Delta, \\
l^{\mathcal{I}} & \subseteq \Delta, \\
(v_1,...,v_n)^{\mathcal{I}} & = v_1^{\mathcal{I}} \cap ... \cap v_n^{\mathcal{I}}, \\
(l/P)^{\mathcal{I}} & = (l^{\mathcal{I}} \bowtie \texttt{child} \bowtie P^{\mathcal{I}}).2, \text{ where} \\
 & \quad (X \bowtie \texttt{child} \bowtie Y).2 = \{y \in Y \mid \exists x (x \in X \wedge (x,y) \in \texttt{child})\}. \\
(T_1,...,T_n)^{\mathcal{I}} & = T_1^{\mathcal{I}} \cap ... \cap T_n^{\mathcal{I}}, \\
(P/(T_1,...,T_n))^{\mathcal{I}} & = (P^{\mathcal{I}} \bowtie \texttt{child} \bowtie (T_1,...,T_n)^{\mathcal{I}}).2, \\
(t_1,...,t_n)^{\mathcal{I}} & = t_1^{\mathcal{I}} \cap ... \cap t_n^{\mathcal{I}}, \\
(P/(t_1,...,t_n))^{\mathcal{I}} & = (P^{\mathcal{I}} \bowtie \texttt{child} \bowtie (t_1,...,t_n)^{\mathcal{I}}).2, \\
(T:v)^{\mathcal{I}} & = (T^{\mathcal{I}} \bowtie \texttt{child} \bowtie v^{\mathcal{I}}).2,
\end{array}
$$

We say that an expression E_1 *is subsumed by* an expression E_2, or that E_2 *subsumes* E_1, written $E_1 \sqsubseteq E_2$, if $E_1^{\mathcal{I}} \subseteq E_2^{\mathcal{I}}$. If both $E_1 \sqsubseteq E_2$ and $E_2 \sqsubseteq E_1$, then E_1 is equivalent to E_2, written $E_1 \equiv E_2$.

Theorem 1. *The following rules hold:*

R1. $(v_1,...,v_n) \sqsubseteq (v_1,...,v_i)$, $(T_1,...,T_n) \sqsubseteq (T_1,...,T_i)$, $(t_1,...,t_n) \sqsubseteq (t_1,...,t_i)$, *for any* $1 \leq i \leq n$;
R2. $P/t \sqsubseteq t$,
R3. $P/T{:}v \sqsubseteq P{:}v$,
R4. if P_1/t_1 *and* P_2/t_2 *are valid trees, then* $P_1 \sqsubseteq P_2 \wedge t_1 \sqsubseteq t_2 \Rightarrow P_1/t_1 \sqsubseteq P_2/t_2$,
R5. $P/(T_1{:}v_1,...,T_n{:}v_n) \sqsubseteq P{:}(v_1,...,v_n)$.

Proof. (R1) follows from the property of sets intersection; to prove (R2) note that $(P/t)^{\mathcal{I}} = (P^{\mathcal{I}} \bowtie \texttt{child} \bowtie t^{\mathcal{I}}).2 \subseteq t^{\mathcal{I}}$; in proof of (R3) we use the fact that the `child` relation is transitively closed, thus we have $(P/T{:}v)^{\mathcal{I}} = ((P^{\mathcal{I}} \bowtie \texttt{child} \bowtie T^{\mathcal{I}}).2) \bowtie \texttt{child} \bowtie v^{\mathcal{I}}).2 \subseteq (P^{\mathcal{I}} \bowtie \texttt{child} \bowtie v^{\mathcal{I}}).2$; (R4) is a standard property of partial ordering relations. (R5) follows from the definition and from (R1) and (R3): $(P/(T_1{:}v_1, ..., T_n{:}v_n))^{\mathcal{I}} = (P/T_1{:}v_1, ..., P/T_n{:}v_n))^{\mathcal{I}} \subseteq (P{:}v_1, ..., P{:}v_n))^{\mathcal{I}} = (P{:}(v_1, ..., v_n))^{\mathcal{I}}$. □

5 Data merging in data integration

5.1 Ontology- and key-based data merging

The main challenge in data integration is to find methods for merging partial answers into a one, duplicate-free, well constructed answer. These functions should be performed be the `Merge` component (Fig.1). In order to realise the expected functionality of the `Merge`, we can use:

- definitions of source data schemas given by means of DTD or XML Schemas, if they are available;
- domain ontologies both for names and tags (in the schema level) and for values (in the instance level),
- any other resources which can be used to understand and classify data correctly, such as dictionaries, taxonomies, thesauri, user provided match and mismatch information as well as knowledge discovered in data, e.g. keys and statistical characteristics.

Using these resources and methods, we can classify XML tree fragments – such as values and paths and tree patterns – into equivalence classes with respect to the synonymy relation. As the result the following set of indexes can be created:

```
ValueClass(DbId,Constraint,Value,ValueClassId)
ValueClassRep(ValueClassId,ValueRep)
PatternClass(DbId,Constraint,Pattern,PatternClassId)
PatternClassRep(PatternClassId,PatternRep),
```

where `DbId` is a database identifier being the source for `Value` or `Pattern`; and `Constraint` is an additional condition used to classify the `Value` or `Pattern`. A value equivalence class is identified by `ValueClassId` and is represented by a chosen value `ValueRep`. Thus, the value representing the class of semantically equivalent values resolves such issues as diversity of currencies, measures, and representation formats, in order to overcome difficulties in duplicate elimination and value comparison. For text values there is a problem with synonyms, different languages, jargon and so on. To solve these problems, methods from information retrieval can be used [2,5].

Next, subsumption on these classes can be defined, where $v_1 \sqsubseteq v_2$ means that v_1 is *more desirable* than v_2, because v_1 is more *informative*, more *reliable*

(one database may be considered to be more reliable than others) or has higher *precision*. Correct definition of this relation is crucial because it is used to define subsumption relation over complex expressions.

In order to define subsumption on patterns, we start with establishing it on individual labels. As for values, patterns with different syntax may have the same meaning, e.g. `fname`, `first-name`, and `firstname` belong to the same equivalence class. The path pattern `author/name` and the tree pattern `author/name(fname,lname)` will belong to different, but somehow related equivalence classes. Again, identification of such patterns can be supported by ontologies, statistics and machine learning methods [8,15]. For complex patterns, the subsumption relation can be inferred from atomic patterns by means of rules proved in Theorem 1. Subsumptions will be indexed to enable efficient subsumption testing. Indexes will be used to test whether two given values or patterns are in subsumption relation or not:

```
ValueSubsumption(SubsumedValue,SubsumingValue)
PatternSubsumption(SubsumedPattern,SubsumingPattern)
```

The following inference rules (2) and (3) follow from Theorem 1 and are of special importance for data merging during data integration:

$$P_1 \sqsubseteq P_2 \wedge v_1 \sqsubseteq v_2 \Rightarrow P_1{:}v_1 \sqsubseteq P_2{:}v_2, \tag{2}$$

$$\begin{aligned} P/(P_1, ..., P_n) \sqsubseteq P'/(P'_1, ..., P'_m) \wedge (v_1, ..., v_n) \sqsubseteq (v'_1, ..., v'_m) \Rightarrow \\ \Rightarrow P/(P_1{:}v_1, ..., P_n{:}v_n) \sqsubseteq P'/(P'_1{:}v'_1, ..., P'_m{:}v'_m) \end{aligned} \tag{3}$$

It follows from Theorem 1 that it is sufficient to inspect subsumptions between trees and paths, rather than between trees and trees. So, for some paths and trees, precomputed index table could be maintained in the system:

```
TreePathSubsumption(SubsumedTree,SubsumingPath)
```

Example 1. For data trees t_1, t_2, and t_3 from Figure 2, we have:

Patterns:
`article` $\equiv$ `paper`,
`author/fname` $\sqsubseteq$ `author`,
`author/lname` $\sqsubseteq$ `author`,
`author/(fname,lname)` $\sqsubseteq$ `author`.

Values:
`"John Smith"` $\sqsubseteq$ `"John"`,
`"John Smith"` $\sqsubseteq$ `"Smith"`.

Trees:
`author/(fname:"John",lname:"Smith")` $\sqsubseteq$ `author:"John Smith"`,
$t_3 \sqsubseteq t_1$, $t_3 \sqsubseteq t_2$. □

Note that if we restricted ourselves to paths only, we would not be able to construct the expected minimal result tree t_3 from t_1 and t_2, because neither `author/fname:"John"` nor `author/lname:"Smith"` is subsumed by the path `author:"John Smith"`.

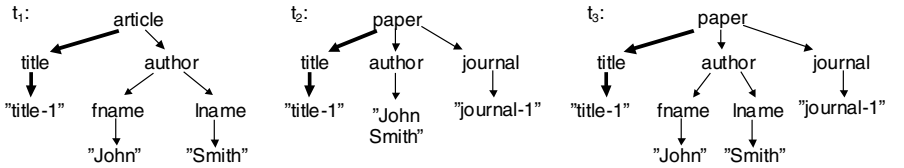


Fig. 2. Source data trees t_1 and t_2, and their join, $t_3 = join(\{t_1, t_2\})$, fat arrows denote equivalent key paths, i.e. all trees represent the same real world entity

Trees t_1 and t_2 from Example 1 can be joined because there are two absolute keys holding in t_1 and t_2, respectively, which are equivalent and have equivalent values, i.e. `article/title:"title-1"` $\equiv$ `paper/title:"title-1"`. Thus, these trees could be treated as describing the same entity from the semantic domain of interest. When trees describe different entities they are non-joinable. Non-joinable trees are merged in such a way that a new root label is created and all trees under consideration become the highest-level subtrees of the newly created root (Fig. 3).

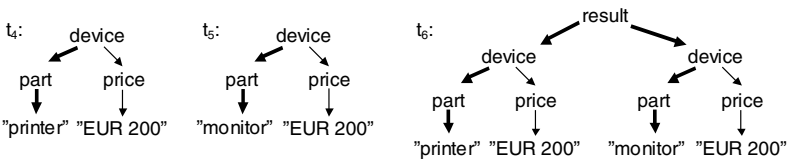


Fig. 3. Data trees t_4 and t_5, and their merge, $t_6 = merge(result, \{t_4, t_5\})$, fat arrows denote non-equivalent key paths, as t_4 and t_5 represent different real world entities

5.2 Merging algorithms

Given a set T of trees constituting partial answers to a query, and a label L for labelling the result tree, the following formula describes the merging of trees from T into the result tree:

$$\begin{aligned} merge(L, T) &= L/(join(G_1), ..., join(G_k)), \\ \{G_1, ..., G_k\} &= T|_{\equiv_{keypath}} \end{aligned} \tag{4}$$

The result tree, $merge(L, T)$, is constructed in such a way that a new root labelled L is created. All the subtrees immediately following L (the highest-level subtrees) are joins over sets G_i, $1 \leq i \leq k$, of trees. The family $\{G_1, ..., G_k\}$ consists of all equivalence classes in T determined be in the equivalence relation $\equiv_{keypath}$. To one equivalence class belong all trees that have equivalent key paths.

The merging process can be performed using the following two algorithms:

Algorithm 1 *(merge of a set of minimal trees)*
Input: A set $T = \{t_1, ..., t_n\}$ of trees; each tree is a minimal partial answer to a query returned from one of data sources.
L – a root label for the result tree.
Output: A tree t^* such that $t^* \sqsubseteq \{t_1, ..., t_n\}$, and t^* is minimal.
Steps:

1. Divide T into groups $\{G_1, ..., G_k\}$ of trees with equivalent key paths.
2. For each group G_i compute the join $g_i = join(G_i)$, $1 \leq i \leq k$.
3. Construct the tree $L/(g_1, ..., g_p)$ and denote it by t^*.
4. Tree t^* is the expected merge of $\{t_1, ..., t_n\}$, denoted $t^* = merge(L, \{t_1, ..., t_n\})$.

Algorithm 2 *(join of a set of joinable trees)*
Input: A set $T = \{t_1, ..., t_n\}$ of joinable trees, i.e. trees with equivalent key paths.
Output: A tree t^* such that $t^* \sqsubseteq \{t_1, ..., t_n\}$, and t^* is minimal.
Steps:

1. Create a tree t as a sum of trees from T; t has a root label representing the equivalence class from T which is being joined, and each highest-level subtree (i.e. a subtree immediately following the root) of t_i is a highest-level subtree of t.
2. Denote: Π – a set of all paths from t; Σ – a set of all highest-level subtrees from t.
3. *Reduction*:
   for each $p \in \Pi$
   for each $S \in \Sigma$ **such that** $p \notin S$, and p and S have the same key path
   if $S \sqsubseteq p$ **then** mark p for deletion
   remove from t all paths marked for deletion and denote the result by t^*
4. Tree t^* is the expected join of $\{t_1, ..., t_n\}$, denoted $t^* = join(\{t_1, ..., t_n\})$.

6 Conclusions

In this paper we proposed a new approach to data merging in data integration systems. We previously introduced the problem using an example of integrating heterogeneous biological databases [9], where the existence of some domain oriented taxonomies and ontologies makes the problem more feasible than in the case of "deep Web integration" [8]. We believe, however, that our method is applicable in other setting as well.

The main novelty of our approach is:

- Apart from traditional methods rooted in information analysis [2,5], we use constraints (keys) discovered in data sets (by means of statistic and data mining methods) to support data classification and data merging.
- We introduce a simple formal language to specify data patterns and data instances and propose semantics for it. We use it to define subsumption between data formally, and to show that our approach is both expressive and sound.

The method presented in the paper is a part of our research on semi-structured data integration [9,14] and XML data transformation [13].
Acknowledgement: EH's research is funded by the Medical Research Council, UK.

References

1. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Petel-Schneider, P., Eds.: *The Description Logic Handbook: Theory, Implementation and Applications*, Cambridge University Press, 2003.
2. Baeza-Yates, R., Ribeiro-Neto, B.: *Modern Information Retrieval*, Addison Wesley, New York, 1999.
3. Buneman, P., Davidson, S. B., Fan, W., Hara, C. S., Tan, W. C.: Keys for XML, *Computer networks*, **39**(5), 2002, 473–487.
4. Buneman, P., Davidson, S. B., Fan, W., Hara, C. S., Tan, W. C.: Reasoning about keys for XML, *Information Systems*, **28**(8), 2003, 1037–1063.
5. Carvalho, J. C. P., da Silva, A. S.: Finding similar identities among objects from multiple web sources, *Fifth ACM CIKM International Workshop on Web Information and Data Management, WIDM 2003*, ACM, 2003, 90–93.
6. Doan, A., Domingos, P., Halevy, A.: Reconciling Schemas of Disparate Data Sources: A Machine-Learning Approach, *ACM SIGMOD 2001*, ACM, 2001, 509–520.
7. Grahne, G., Zhu, J.: Discovering approximate keys in XML data, *ACM CIKM International Conference on Information and Knowledge Management*, ACM, 2002, 453–460.
8. He, B., Chang, K. C.-C., Han, J.: Discovering complex matchings across web query interfaces: a correlation mining approach, *Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2004*, ACM, 2004, 148–157.
9. Hunt, E., Pafilis, E., Tulloch, I., Wilson, J.: Index-Driven XML Data Integration to Support Functional Genomics In: Workshop on Data Integration in Life Sciences, DILS'04, *Lecture Notes in Computer Science*, **2994**, 2004, 95–109.
10. Jagadish, H., Olken, F.: Data Management for life science research, *SIGMOD Record*, **33**(2), 2004, 15–20.
11. Lacroix, Z., Boucelma, O., Essid, M.: The Biological Integration System, *Fifth ACM CIKM International Workshop on Web Information and Data Management, WIDM 2003*, ACM, 2003, 45–49.
12. Lacroix, Z., Critchlow, T., Eds.: *Bioinformatics: Managing Scientific Data*, Morgan Kaufman, 2003.
13. Pankowski, T.: A High-Level Language for Specifying XML Data Transformations, In: Advances in Databases and Information Systems, ADBIS 2004, *Lecture Notes in Computer Science*, **3255**, 2004, 159–172.
14. Pankowski, T.: Processing XPath expressions in relational databases, In: Theory and Practice of Computer Science, SOFSEM 2004, *Lecture Notes in Computer Science*, **2932**, 2004, 265–276.
15. Theobald, A., Weikum, G.: The Index-Based XXL Search Engine for Querying XML Data with Relevance Ranking, In: Advances in Database Technology - EDBT 2002, *Lecture Notes in Computer Science*, **2287**, 2002, 477– 495.

Approximation Quality of the RBS Ranking Algorithm*

Marcin Sydow

Polish-Japanese Institute of Information Technology
Koszykowa 86, 02-008 Warsaw, Poland
e-mail: msyd@pjwstk.edu.pl

Abstract. The RBS algorithm is a novel link-based algorithm for ranking results of a search engine. RBS may be viewed as an extension of PageRank by a parameterized "back button" modeling. RBS is based on the "random surfer with back step" model [7] similarly as PageRank is based on the simpler "random surfer" model [4]. To scale to real Web RBS computes merely a fast probabilistic approximation of the ranking induced by the "random surfer with back step" model [6].

In this paper we experimentally measure the quality of this approximation using a high quality synthetic Web evolution model [5] of our own implementation.

The results demonstrate that RBS is a very good approximation to the "ideal" ranking. Furthermore, as the experiment shows, RBS clearly outperforms PageRank in "back step" modeling even if we try to parameterize the latter.

1 Introduction

The RBS algorithm is a novel link-based algorithm for ranking results of a search engine. RBS may be viewed as an extension of Google's PageRank by a parameterized "back button" modeling, which was omitted in an original PageRank, despite the fact that "back button" is reportedly heavily used by surfers of the Web. Link analysis of the *Web graph*[1] turned out to be a powerful approach in the context of exponentially growing Web with documents of extremally varying type, quality and size, where traditional Text Information Retrieval techniques are not robust enough.

Link-based ranking algorithms are used for ordering the results of search queries passed to search engines. More precisely, each document is assigned some *ranking score* which is computed by ranking algorithm according to some intuitive model, and then documents are presented in non-increasing order of their ranking scores. Since the number of documents returned by a search engine may easily be far too large to be seen by a user, such an ordering is extremely important. Link-based ranking algorithms proved their value in practice. The most famous example is perhaps Google's PageRank [4] which has been successfully applied in this leading world search engine since its origin.

* This paper is based on fragments of the author's PhD thesis [6]
[1] Each document is represented as a node and each hyperlink as a directed edge

1.1 Traditional Random Surfer Model

PageRank is based on the Random Surfer Model [4].The model has a parameter $0 \leq a \leq 1$, which influences the overall behavior of a modeled surfer. More precisely, in each time step t a surfer, currently visiting a document $p(t)$ randomly performs *one* of the two possible actions, accordingly to a model parameter a. With probability a follows uniformly picked out-link or with the remaining probability $1 - a$ randomly jumps to another page with uniform distribution. The value of $1 - a$ is called a *decay factor*.

PageRank computes the stationary distribution of the Markov chain defined above.Thus, each document's ranking is a limit probability of visiting it according to the random surfer model.

Unfortunately, the traditional Random Surfer model *does not* reflect some very common aspect of Web surfing i.e *reverting* to a page visited in a previous step. Reverting to a previous page constitutes a substantial part of Web surfing behaviour, and in our opinion, it *should* be reflected in ranking schemes.

1.2 Random Surfer with Back Step

We extend traditional Random Surfer model by introducing another parameter $0 \leq b \leq 1$, satisfying $a + b \leq 1$, which represents the *back-step* probability. Random surfer is visiting pages in discrete time steps, so that in each step while visiting a page p they may perform one of the following actions:

1. with probability a follow uniformly picked outgoing link of the page p. In the case, there are no out-links on the page, the surfer performs random jump described in the last point.
2. with probability b *reverts* to the previously visited page ("back button" modeling), assuming the previous step was of type 1 (follow link) – otherwise ignores reverting.
3. with the remaining probability $1 - (a + b)$, randomly jumps to another page according to the uniform distribution.

1.3 Previous Work

Theoretical foundations of incorporating back-button usage to Web surfing models may be found in [1] by Fagin et al.

Very recently, idependently on our work, there were proposed two variants of modeling back-button usage in [3]. The first variant assumes that two consecutive uses of the "back-button" cancell each other, which, unfortunately, is *in contradiction* with back-button implementation in real browsers. The second approach, more realistic, is similar to the one discussed in this paper but has two drawbacks, compared to ours: is not parameterized (assumes that back-button usage probability is always the same as following a link), assumes that back-button *cannot* be used twice in a row.

Neither of these drawbacks is present in our model.

2 Motivation

In [6] we have presented efficient RBS algorithm, which is based on the probabilistic approximation of our random surfer model with back step. The algorithm is convergent, efficient, scales to real Web and returns the set of results which are generally similar to that of PageRank, what somehow "guarantees" that RBS does not produce insensible results.

In this paper we will answer the two important questions:

1. Is RBS a good approximation of our "ideal" random surfer with back step model?
2. Is RBS a really *novel* ranking scheme?

The first question is posed in the context of the situation that RBS computes its ranking according to the probabilistic approximation of the random surfer model with back step rather than to the model itself.

The second question is motivated as follows. The random surfer model proposed in [1], although defined in the way which considered back steps, led to the same stationary distribution as would have been obtained without back step[2] (this phenomenon is also briefly commented in [6]). Despite the fact, that we have seen that RBS produces results being different than those produced by PageRank with *some* value of the decay factor, we still may be not convinced, that the results of RBS cannot be obtained by the traditional PageRank with some *appropriately set* decay parameter. In other words, we will ensure whether our model cannot be simulated by PageRank.

3 Preliminaries

3.1 RBS-walk Simulator and RBS

The straightforward, practical way of computing ranking according to the "random surfer with back step" model is to simulate the random surfing of sufficiently large number of artifificial surfers according to the model rules and record, for each page, the total number of visits made. We have implemented this approach and experimentally observed in [6], that relative numbers of visits made to each page converge to the desired disribution with arbitrary precision as the function of simulation duration. We call this approach as "RBS-walk simulator" [6].

However, as we also observe in [6], RBS-walk simulator, although being capable of producing the desired ranking among the documents, is too slow for working on large collections of Web documents[3].

[2] In a few words, our ranking differs from Fagin's because, in RBS model, back step is not allowed directly after random jump. This important detail does not reflect browsers implementation, but rather the user behavior

[3] In practice, the time complexity of RBS-walk simulator is approximately $\Theta(n^2)$

Thus, we developed a fast probabilistic approximation of the ideal ranking, which we call *RBS*. RBS is very similar to the power method of computing PageRank [4] with the addition of an extra "back-step flow" of rank score which each page x gets, due to potential back-step, from any page y, being linked by x. We call this quantity $R_B(x, y)$, and suggest it to be proportional to *the conditional probability* of visiting the page x in the previous step assuming that in the current step y is visited:

$$R_B(x, y) = \beta(y) \frac{aR(x)}{outDeg(x)R(y)}, \tag{1}$$

where $\beta(y)$ is some proportionality factor dependent on y, which is introduced to normalize the probabilities. After some computations, the final equation for the total amount of rank a page x gets through "back-step" flow is:

$$R_B(x) = ab \frac{R(x)}{outDeg(x)} \sum_{y \in OUT(x)} \frac{R(y)}{R_{IN}(y)}, \tag{2}$$

Where $R_{IN}(y)$ denotes the amount of rank flow which y receives due to "follow link" flow. Keeping other things similarily as in PageRank computation, we can adapt a power method to compute the resulting ranking vector R efficiently. We call our method RBS algorithm. In [6] we derive from the results obtained in [1], that a vector R exists and is unique under assumption $b < 1/2$. Due to space limitations, for more details we refer the reader to [6].

3.2 The Hybrid Model of Web Graph Evolution

The Web graph is dynamically evolving and it has many special properties, which may be appropriately explored to make link-based algorithms more efficient. Because of this, many researchers have been working on increasingly sophisticated synthetic Web graph models (for a survey, see [6], ch. 5).

One of the best models proposed is the *Hybrid Model* [5]. In this model, the evolution of the graph is modeled in epochs. In each epoch, there is a constant factor of new nodes added to the existing ones. The model has two parameters a, b, where $0 \leq a, b \leq 1$ and $a + b \leq 1$. For each node, there are edges randomly added. The probability of choosing the destination of new edge is either proportional to its in-degree with probability a, or with probability b proportional to its PageRank, or with the remaining probability choosen uniformly from all the existing nodes.

The author implemented his own version of the model generator [6], which is used in the experiment described in this paper.

3.3 Comparing Rankings

The ranking generated by a ranking scheme may be viewewd as a list of document identifiers (ranging from 1 to n, the number of considered documents)

sorted non-increasingly by a ranking score. In practice, only the *top-k* sublist, for some moderate value of k, is important, because the average user checks only the limited number of top-ranked documents.

To compare rankings in this paper we use two distance measures described below. The both measures are *distance metrics* (proofs may be found in [2]).

For any k-list σ, let us assume that D_σ denote the *domain* of this list (i.e. the set of all items present in the list) and $\sigma(i)$ denote the *position* of an item $i \in D_\sigma$. Let us assume that the compared k-lists are denoted σ_1 and σ_2.

`Spearman footrule with location parameter` $k+1$`:`

$$F(\sigma_1, \sigma_2) = \sum_{i \in D_{\sigma_1} \cup D_{\sigma_2}} |\sigma_1'(i) - \sigma_2'(i)|, \tag{3}$$

where σ_1' is σ_1 extended in that way that the items belonging to $D_{\sigma_2} \setminus D_{\sigma_1}$ are *all* put on the $(k+1)$-th position. The definition of σ_2 is analogous.

`Weighted sum of symmetric differences:`

$$\delta(\sigma_1, \sigma_2) = \frac{1}{k} \Sigma_{i=1}^{k} \frac{card(D_{\sigma_1^{(i)}} \oplus D_{\sigma_2^{(i)}})}{2i}, \tag{4}$$

where $\oplus$ denotes the *symmetric difference* between two sets (intersection of the arguments subtracted from their sum) and $\sigma_1^{(i)}$ denotes the i-list consisting of the top i items of σ_1 (the $\sigma_2^{(i)}$ is defined in an analogous way).

Note that the second metric is a little more sophisticated than the first, since it is more *aware* of the fact that changes on lower positions are *more important* than changes on higher positions in rankings. This property is extremely important in the case of comparing rankings.

4 Experiment Design and Implementation

To answer the questions mentioned in the section 2 we designed the following experimental scheme: 1) Take a directed graph $G(V, E)$, 2) Produce the "ideal" ranking on G, according to the random surfer model with back step for some parameters a, b. 3) Compute the fast RBS approximation ranking on G with the same parameters. 4) Compute PageRank on G for a wide range of values of the decay parameter. 5) Compute the *distance* between the "ideal" ranking and the RBS ranking. 6) Compute the distance between the ideal ranking and those obtained with PageRank. 7) Compare the above distances.

We made the following decisions in order to refine the above scheme.

The RBS-walk simulator, described before, seemed to be the ideal tool to obtain the "ideal" ranking according to our model, thus we decided to use it in the experiment.

About the graph to take, we had access to large, 50- and 80- million-page real Web samples (used in [7,6]), but we could not use them here, because of the square time complexity of the RBS-walk simulator. Even if we had

about 10GB of memory needed to use RBS-walk simulator on 80-million-node graph, we assessed that the process of generating a single ranking vector with the least acceptable accuracy would take hours. Of course, with 2GB of the memory being at our disposal, we would have to change the implementation of the RBS-walk simulator to keep the graph and histories on disk. Therefore the actual time of generating a single ranking would have increased couple of times. Since we had to generate the "ideal" ranking multiple times (for various parameter settings) to increase the reliability of the results, using our large real Web graph sample here, turned out to be practically impossible.

To overcome this problem, we had to input much smaller graphs to the RBS-simulator. We decided to use a high quality artificial Web graph model – the *hybrid* model [5], of the appropriate size. We used our own implementation of the hybrid generator described in [6]. We also used other graphs to perform similar experiments – e.g. the *uniform* model ([6]).

To measure distance between rankings we produced top-k lists for various values of k and for each value we used: normalized Spearman's footrule with location parameter $k + 1$ and normalized, weighted sum of symmetric differences.

To be sure that the traditional PageRank is not capable of simulating random surfer with back step, for each set of parameters we have computed PageRank for a wide range of decay parameter values and choose the ranking which was the *closest* to the ideal ranking.

Now, we precisely report details of one particular experiment we performed according to the above decisions.

1. We used 103248-node graph generated by our hybrid generator described in [6] with the following parameters $\alpha = \beta = 0.33, growthFactor : 0.1$, PageRank decay factor: 0.1, number of epochs: 101.
2. We generated the "ideal" ranking on this graph with RBS-walk simulator for 3 different sets of a, b, each time with 3 000 000 surfers and 200 turns. The generation process took about 5 minutes each time.
3. We computed the RBS ranking on the graph with the same 3 sets of parameters.
4. We computed PageRank on the graph for a wide range of the decay parameter values.
5. For each set of parameters and each ranking we computed top-k lists for various values of k ($k \in \{20, 50, 100, 500, 1000\}$).
6. For each "ideal" top-k list we computed its distance to the corresponding top-k list of RBS. We used two different distance metrics mentioned previously.
7. For each "ideal" top-k list we computed the distance to *the closest* PageRank k-list among the wide range of decay parameter values for both distance measures separately.

5 Experimental Results

The results of this experiment left no doubts about both: the quality of our approximation and PageRank's disability of simulating back step. Below we demonstrate three tables for the three different sets of the model parameters: 1) a=0.85, b=0.085, 2) a=0.85, b=0.1, 3) a=0.85, b=0.05, respectively.

In the tables below, k is the number of top considered items, $RBSn$ and PRn are the distances[4] from RBS- and PageRank- generated (respectively) rankings to the "ideal" ranking and dn is the value of PageRank's decay factor, for which the smallest distance was observed [5].

k	RBS1	PR1	d1	RBS2	PR2	d2
20	0.0	0.0190	0.09	0.0	0.0230	0.09
50	0.0015	0.0329	0.05	0.0012	0.0320	0.06
100	0.0021	0.0235	0.12	0.0018	0.0265	0.05
500	0.0022	0.0074	0.1	0.0020	0.0115	0.1
1000	0.0028	0.0045	0.09	0.0025	0.0075	0.1

k	RBS1	PR1	d1	RBS2	PR2	d2
20	0.0	0.0238	0.08	0.0	0.0266	0.08
50	0.0023	0.0407	0.05	0.0018	0.0381	0.05
100	0.0019	0.0287	0.05	0.0018	0.0317	0.05
500	0.0021	0.009	0.1	0.0020	0.014	0.1
1000	0.0029	0.0052	0.08	0.0025	0.009	0.08

k	RBS1	PR1	d1	RBS2	PR2	d2
20	0.0047	0.0095	0.06	0.01	0.0075	0.06
50	0.0039	0.0188	0.05	0.0064	0.0154	0.05
100	0.0021	0.0128	0.13	0.004	0.0134	0.12
500	0.002	0.0045	0.12	0.0023	0.0064	0.12
1000	0.0027	0.0034	0.12	0.0026	0.0047	0.12

6 Conclusion

We can clearly see in the above tables that:

1. Our fast RBS algorithm approximates the "ideal" back-step ranking very well (low distance).
2. RBS definitely outperforms PageRank in back-step modeling, even if we take the minimal distance of the latter among the wide range of decay factor. PageRank is not capable of modeling back step.

[4] where $n = 1$ stands for Spearman footrule distance and $n = 2$ for weighted sum of symmetric difference

[5] for PageRank we present the *smallest possible* distance among a wide range of decay values

It is important to note that there does not seem to exist any simple rule of choosing the decay value to best approximate back step with use of PageRank, since the best decay values above vary from case to case.

We obtained similar results for other graphs and parameter settings.

7 Future Directions

Some further possible directions of RBS evaluation are: user evaluation of RBS-generated ranking (compared to PageRank) and comparison of RBS ranking with Web servers clickthrough data (i.e. the actual visit frequencies to Web documents). Both these directions, however, need time consuming processes of collecting the data from users and Web servers.

References

1. R. Fagin, A.Karlin, J.Kleinberg, P.Raghavan, S.Rajagopalan, R.Rubinfeld, M.Sudan, and A.Tomkins. Random walks with back buttons. *Annals of Applied Probability*, 11(3), 2001.
2. R. Fagin, R.Kumar, and D.Sivakumar. Comparing top k lists. *SIAM J. Discrete Mathematics*, 17(1):134–160, 2003.
3. F. Mathieu and M. Bouklit. The effect of the back button in a random walk (poster): Application for pagerank. In *Proceedings of the 13th WWW Conference. Alternate Track. Papers and Posters*. ACM Press, 2004.
4. L. Page, S.Brin, R.Motwani, and T.Winograd. The pagerank citation ranking: Bringing order to the web. In *Stanford Digital Library Working Paper*, 1998.
5. G. Pandurangan, P.Raghavan, and E.Upfal. Using pagerank to characterize web structure. In *Proceedings of the 8th Annual International Computing and Combinatorics Conference*, 2002.
6. M. Sydow. *Link Analysis of the Web Graph. Measurements, Models and Algorithms for Web Information Retrieval*. PhD dissertation., Polish Academy of Sciences, Institute of Computer Science, Warsaw, 2004.
7. M. Sydow. Random surfer with back step (poster). In *Proceedings of the 13th International WWW Conference, (Alternate Track. Papers and Posters)*, pages 352–353. ACM press, 2004.

Part IV

Regular Sessions: Biologically Motivated Algorithms and Systems

Effects of Versatile Crossover and Mutation Operators on Evolutionary Search in Partition and Permutation Problems

Zbigniew Kokosiński

Cracow University of Technology, Faculty of Electr. & Computer Eng.,
Dept. of Control Eng., ul. Warszawska 24, 31-155 Kraków, Poland

Abstract. The paper investigates the influence of versatile crossover and mutation operators on the efficiency of evolutionary search in solving two important classes of hard optimization problems. Chromosome representations of set partitions and permutations defined in the paper are not problem–oriented and are described together with their versatile variation operators. The proposed representations are tested in evolutionary programs on standard partitions and permutation problems i.e. graph coloring (GCP) and traveling salesman (TSP). The optimization results vary depending on the problem class. They are relatively positive with respect to GCP and negative for TSP.

1 Introduction

In evolutionary optimization programs the crucial issue is the proper choice of chromosome representation and corresponding genetic operators for the problem at hand : the selection of chromosome representations and operators determines the overall performance of evolutionary algorithm for given problem [14,15]. The opposite is also true: a versatile representation supported by efficient variation operators is of interest for many potential applications.

Classical crossover and/or mutation operators produce invalid instances when applied to some combinatorial objects in common representations [13]. Set partitions and permutations are representative classes of this kind of objects. In order to resolve the problem of invalid offspring one may use a conventional representation and operators, and then apply elimination or repair strategies for chromosomes, eventually try to design representation-specific genetic operations for selected representation which do not cause such anomalies. Alternative approach is to find a different representation (versatile or problem-specific).

In [17] a general method of handling permutation chromosomes is described in which permutations are represented by their ranks and genetic operators like crossover and mutation are performed on objects' ranks in binary form. This approach can be easily generalized for other combinatorial objects. Main disadvantages of the method are: 1. individuals are represented by huge numbers (object ranks) 2. object ranks do not contain problem-related ge-

netic information (no inheritance) 3. genetic operators are relatively complex (ranking and unranking algorithms are required).

In the present paper we describe versatile recombination operators for set partitions [10]. A representation of set permutations is also presented in which conventional crossover and mutation on integer vectors do perform correctly [9]. The main aim of the paper is to verify applicability of such representations in genetic search with no particular restrictions of the search space known a priori. That is the case of many intractable partition/permutation problems. The efficiency of both representations and their variation operators is verified by computer experiments on two standard optimization problems i.e. graph coloring (GCP) and traveling salesman (TSP).

The rest of the paper is organized as follows. Section 2 is devoted to a partition representation and operators. In the next section a permutation representation and its operators is described. Section 4 reports selected experimental results and their discussion. Section 5 contains conclusions.

2 Partition chomosome and its operators

Let us introduce two basic representations of partitions used in graph coloring. In set representation each block of partition does correspond to a single color. In assignement representation available colors are assigned to an ordered sequence of graph vertices/edges. It is well known that edge coloring of a given graph is equivalent to vertex coloring of the corresponding edge graph.

2.1 Sum–Product Partition Crossover

The recombination operator called Sum–Product Partition Crossover (SPPX) employs for offspring generation simple set sum and set product operations on block of partitions and a random mechanism of operand selection from randomly determined 4 parental chromosomes. SPPX is composed of two procedures PRODUCT and SUM which are applied to the pair of chromosomes p=$\{V_1^p, \ldots, V_k^p\}$, r=$\{V_1^r, \ldots, V_l^r\}$ and produce a pair of chromosomes s=$\{V_1^s, \ldots, V_m^s\}$ and t=$\{V_1^t, \ldots, V_n^t\}$ with probabilities of elementary operations satisfying: $0 \leq \text{Prob(PRODUCT)} < \text{Prob(SUM)} \leq 1$. A pseudocode of the procedure SPPX is presented in Fig. 1.

Example 1

Two parents represent different 5–colorings of a graph with 10 vertices: p=p1=p2={1,8}{2,7}{3,10}{4,6} and r=r1=r2={1}{2,8}{3,7,10}{4,9} {5,6}. Let us assume Prob(SUM)=0.7, Prob(PRODUCT)=0.5 . Let rand2=0.3 and SUM is computed with h=3, j=4. Thus, V_3^p ={3,10} and V_4^r ={4,9}. We obtain $V_1^s = V_1^t$ = {3,4,9,10} and after lexicographic reordering of blocks the resulting partitions are s1={1,8}{2,7}{3,4,9,10}{5}{6} and t1={1}{2,8}{3,4,9,10}{5,6}{7}, respectively.

```
procedure: SPPX (p,r,s1,s2,t1,t2,Prob(SUM),Prob(PRODUCT))
begin
  s1=t1=s2=t2=∅;
  generate random numbers rand1, rand2 : 0 ≤rand1,rand2≤1;
  if rand1 ≤ Prob(SUM) then SUM(p,r,s1,t1);
  if rand2 ≤ Prob(PRODUCT) then PRODUCT(p,r,s2,t2);
end SPPX;
SUM (p,r,s,t)
begin
  select random h (1≤h≤k) and j (1≤j≤l);
  V_1^s = V_1^t = (V_h^p ∪ V_j^r);
  for i = 1 to k do
    if i ≠ h do if (V_i^p \ V_j^r) nonempty then
      add next block V_i^p \ V_j^r to s;
  for i = 1 to l do
    if i ≠ j do if (V_i^r \ V_h^p) nonempty then
      add next block V_i^r \ V_h^p to t;
end SUM;
PRODUCT(p,r,s,t)
begin
  select random h (1≤h≤k) and j (1≤j≤l);
  V_1^s = V_1^t = (V_h^p ∩ V_j^r);
  for i = 1 to k do
    if i ≠ h do if (V_i^p \ V_1^s) nonempty then
      add next block V_i^p \ V_1^s to s;
  for i = 1 to l do
    if i ≠ j do if (V_i^r \ V_1^t) nonempty then
      add next block V_i^r \ V_1^t to t;
end PRODUCT;
```

Fig. 1. The recombination operator SPPX

2.2 Mutation operators

Two types of mutation operators are used. Transposition is a classical type of mutation that exchange colors of two randomly selected vertices in the assignment representation. The second mutation operation called First Fit is designed for colorings in partition representationand and is well suited for GCP. In First Fit mutation one block of the partition is selected at random and we try to make a conflict–free assignment of its vertices to other blocks using the heuristic First Fit. Vertices with no conflict–free assignment remain in the original block. Thus, as a result of the mutation First Fit the color assignment is partially rearranged and the number of partition blocks is often reduced by one.

Example 2

In chromosome p=<5,2,**3**,4,1,4,2,**5**,1,3> that represents a 5–coloring of a graph with 10 vertices, Transposition mutation exchanges colors of 2 randomly selected vertices 3 and 8. The resulting chromosome is s=<5,2,5,4,1,4,2,3,1,3> which is another 5–coloring of the given graph in assignment representation.

In chromosome r={1}{2,8}{3,7,10}{4,9}{5,6} that represents a 5–coloring of a graph with 10 vertices, the mutation First Fit performs a conflict–free assignment of vertices from the maximum partition block i.e. {3,7,10} to all remaining blocks. The resulting chromosome is t={1,3}{2,7,8}{4,9,10}{5,6} which is a 4–coloring of the given graph in partition representation.

3 Coset permutation chromosome and its operators

In this paper two versatile representations with classical variation operators are investigated and compared to others known from literature. In both representations standard crossover and mutation operators always produce a legal offspring.

In ordinal representation [13] the representations of tours in TSP are normalized by reference list of items and allow the use of classical crossover which makes this representation a versatile one.

The second representation of permutations is derived from an iterative decomposition of symmetric permutation group S_n into cosets [9]. In this representation set permutations are described by integer sequences called choice functions of indexed families of sets.

Let $< A_i >_{i \in I}$ denote an indexed family of sets $A_i = A$, where: $A = I =$ {1, ... , n}. Any mapping f which "chooses" one element from each set $A_1, ..., A_n$ is called a choice function (or a system of representatives, or a transversal) of the family $< A_i >_{i \in I}$. If for every i $\neq$ j a suplementary condition: $a_i \neq a_j$ is satisfied then any choice function $\alpha =< a_i >_{i \in I}$ that belongs to the indexed family $< A_i >_{i \in I}$ is called n-permutation of the set A. Set of all such choice functions represents the set of all permutations of the n–element set.

Let us denote any permutation π of n-element set $A = \{1, ..., n\}$ by the sequence $< \pi(1), \pi(2), \ldots, \pi(n) >$. The set of all permutations of A is called the symmetric group S_n.

The symmetric group of permutations S_n on the finite set $A = \{1, ..., n\}$ can be decomposed recursively into right and left cosets of S_{n-1} in S_n :

$$S_n = S_{n-1}\tau_n^1 + S_{n-1}\tau_n^2 + \ldots + S_{n-1}\tau_n^n, \tag{1}$$

$$S_n = \tau_n^1 S_{n-1} + \tau_n^2 S_{n-1} + \ldots + \tau_n^n S_{n-1}, \tag{2}$$

where + denotes the union of disjoint sets and τ_i^j denotes the transposition (ij) (τ_i^i is the identity permutation).

Table 1. Coset permutation chromosomes in lexicographic order (n=4)

No.	α	π_1	π_2
1	$<1,1,1,1>$	$<2,3,4,1>$	$<4,1,2,3>$
2	$<1,1,1,2>$	$<4,3,1,2>$	$<3,4,2,1>$
3	$<1,1,1,3>$	$<2,4,1,3>$	$<3,1,4,2>$
4	$<1,1,1,4>$	$<2,3,1,4>$	$<3,1,2,4>$
5	$<1,1,2,1>$	$<3,4,2,1>$	$<4,3,1,2>$
6	$<1,1,2,2>$	$<3,1,4,2>$	$<2,4,1,3>$
7	$<1,1,2,3>$	$<4,1,2,3>$	$<2,3,4,1>$
8	$<1,1,2,4>$	$<3,1,2,4>$	$<2,3,1,4>$
9	$<1,1,3,1>$	$<4,1,3,2>$	$<2,4,3,1>$
10	$<1,1,3,2>$	$<2,4,3,1>$	$<4,1,3,2>$
11	$<1,1,3,3>$	$<2,1,4,3>$	$<2,1,4,3>$
12	$<1,1,3,4>$	$<2,1,3,4>$	$<2,1,3,4>$
13	$<1,2,1,1>$	$<3,2,4,1>$	$<4,2,1,3>$
14	$<1,2,1,2>$	$<3,4,1,2>$	$<3,4,1,2>$
15	$<1,2,1,3>$	$<4,2,1,3>$	$<3,2,4,1>$
16	$<1,2,1,4>$	$<3,2,1,4>$	$<3,2,1,4>$
17	$<1,2,2,1>$	$<4,3,2,1>$	$<4,3,2,1>$
18	$<1,2,2,2>$	$<1,3,4,2>$	$<1,4,2,3>$
19	$<1,2,2,3>$	$<1,4,2,3>$	$<1,3,4,2>$
20	$<1,2,2,4>$	$<1,3,2,4>$	$<1,3,2,4>$
21	$<1,2,3,1>$	$<4,2,3,1>$	$<4,2,3,1>$
22	$<1,2,3,2>$	$<1,4,3,2>$	$<1,4,3,2>$
23	$<1,2,3,3>$	$<1,2,4,3>$	$<1,2,4,3>$
24	$<1,2,3,4>$	$<1,2,3,4>$	$<1,2,3,4>$

Let $< P_i >_{i\in I}$be indexed family of sets $P_i \subseteq A$, where $P_i = \{1, ...i\}$. Any choice function $\alpha =< p_i >_{i\in I}$, that belongs to Carthesian product $\times_{i\in I} P_i$ represents a permutation of A.

Any set $P_i = \{\tau_i^1, \ldots, \tau_i^j, \ldots, \tau_i^i\}$, for $i \in I = \{1, \ldots, n\}$ is a system of representatives of right cosets (left cosets) and is called the complete right (left) transversal of S_i in S_{i-1} . Moreover, $|P_i| = i$ and $P_k \cap P_l = \emptyset$, for every $k \neq l$, $k, l \in I$. By substituting $\tau_i^j = j$, we receive : $P_i = \{1, \ldots, i\}$. Depending on the decomposition scheme any choice function $\alpha =< p_i >_{i\in I}$,

$\alpha \in \times_{i \in I} P_i$ correspondes to one of the two sequences: $\pi_1 =< \pi_1(1), \ldots, \pi_1(n) >$ or $\pi_2 =< \pi_2(1), ..., \pi_2(n) >$. The sequences π_1 and π_2 are obtained by performing on the elements of the set $\{1, \ldots, n\}$ the transposition sequence $\tau_2^{p_2}, \ldots, \tau_n^{p_n}$ (according to the decomposition scheme (1) or $\tau_n^{p_n}, \ldots, \tau_2^{p_2}$ (according to the decomposition scheme (2) , respectively.

Two exemplary permutation sequences $\pi_1 =< \pi_1(1), \ldots, \pi_1(n) >$ and $\pi_2 =< \pi_2(1), \ldots, \pi_2(n) >$ (represented by choice functions α for $n = 4$ are shown in Table 1.

3.1 Standard crossover operator

Choice functions α shown in previous section are chromosome representation permutations for which appropriate genetic operators have to be chosen. In oposition to other commonly used representations the crossover and mutation operations for our representation are straightforward.

Illustration of exemplary crossover operator is shown in Fig. 2.

1-point crossover
parents:
$\alpha_1 =< 2, 2, 4 \mid 1, 3, 3 >$, $\quad \alpha_2 =< \mathbf{1}, \mathbf{3}, \mathbf{2} \mid \mathbf{4}, \mathbf{2}, \mathbf{5} >$
$\pi_1 =< 5, 6, 2, 4, 1, 7, 3 >$, $\quad \pi_2 =< 7, 1, 3, 6, 4, 2, 5 >$
children:
$\alpha_3 =< 2, 2, 4 \mid \mathbf{4}, \mathbf{2}, \mathbf{5} >$, $\quad \alpha_4 =< \mathbf{1}, \mathbf{3}, \mathbf{2} \mid 1, 3, 3 >$
$\pi_3 =< 1, 3, 6, 7, 4, 2, 5 >$, $\quad \pi_4 =< 4, 5, 6, 2, 1, 7, 3 >$

Fig. 2. The permutation crossover operator

3.2 Standard mutation operator

Similarly, standard 1–point mutation is possible with our representation. At the chromosome position $\alpha(i)$ only alleles $\{1, \ldots, i\}$ are valid mutations what follows directly from the definition of choice function α. Exemplary mutation operator is depicted in Fig. 3.

1-point mutation at $\alpha(7)$
chromosome 1:
$\alpha_1 =< 2, 2, 4, 1, 3, \mathbf{3} >$, $\quad \pi_1 =< 5, 6, 2, 4, 1, 7, 3 >$
chromosome 2:
$\alpha_2 =< 2, 2, 4, 1, 3, \mathbf{2} >$, $\quad \pi_2 =< 5, 6, 7, 4, 1, 3, 2 >$

Fig. 3. The permutation mutation operator

4 Experimental verification

4.1 Graph edge coloring problem

Graph coloring problem (GCP) belongs to the class of NP–hard combinatorial optimizations problems [5,11]. GCP is defined for an undirected graph as an assignment of available colors to graph vertices/edges providing that adjacent vertices/edges are assigned different colors and the number of colors is minimal.

Evolutionary algorithms (EA) are metaheuristics often used for GCP [1–4,8,10,12]. For computer experiments we used edge coloring version of the problem and graph coloring instances available in the web archive [20].

We performed tests on two standard DIMACS graphs queen5-5 (25 vertices, 160 edges) and myciel5 (47 vertices, 236 edges). Initial population of size=50 was randomly generated. We assumed standard probabilities of crossover and mutation operators i.e. 0.7 and 0.1, respectively. The program was executed 10 times with constant number of iterations =1000.

In our experiments (see Table 2) variable program parameters included type of crossover and mutation. We tested all ten combinations of five crossovers: SPPX, CEX [10], Grouping (GX) [8], Standard (Std), UISX [2], and two mutations: Transposition (T), First Fit (FF). In SPPX operator probability of sum was 0.7 while probalility of product 0.3. Both selections mechanisms Roulette (RS) and Tournament (TS) were tested.

For the graph queen5-5 we assumed optimal edge coloring with 12 colors what corresponded to the fittness function =288. Set, Ass — denote Set or Assignment representations of partitions, respectively.

The best results provide combination of Grouping crossover with any type of mutation. CEX crossover with mutation First Fits is equally good as Grouping. The next operator is versatile SPPX which works well (especially with Tournament Section) in finding conflict-free colorings but produces much less opimal colorings. SPPX is always better then standard crossover on assignement representation of partitions. In general the mutation First Fit is superiour to Transposition. It was observed that in some cases setting combinations (UISX,T), (SPPX,FF) and (SPPX,T) can lead to invalid colorings when the number of colors is reduced below optimum.

4.2 Traveling Salesman Problem

The TSP [5] and many of its variants is one of the most popular permutation problems in combinatorial optimization. Evolutionary algorithms (EA) are metaheuristics often used for TSP [13,15,19].

We tested randomly generated problems with N=100,200 and 500 cities. Initial population of size=100 was also randomly generated. We assumed standard probabilities of crossover and mutation operators i.e. 0.7 and 0.1, respectively, and the fitness function $1/lenght$. Scalling and copying of the

Table 2. Comparison of average solutions of edge GCP for various evolutionary program settings (graph queen5-5)

Represen –tation	Cross –over	Muta –tion	Average Fitness		Conflict–free colorings		Optimal colorings	
			RS	TS	RS	TS	RS	TS
Set	SPPX	FF	272,5	269,7	9	9	1	8
Set	SPPX	T	234,1	202,2	10	8	0	0
Ass	Std	FF	191	242,4	4	7	0	5
Ass	Std	T	182,5	182,2	4	5	0	0
Ass	CEX	FF	288	288	10	10	10	10
Ass	CEX	T	267,5	265,8	10	10	0	0
Set	GX	FF	288	288	10	10	10	10
Set	GX	T	288	269,3	10	10	10	0
Set	UISX	FF	264,3	288	10	10	6	10
Set	UISX	T	208,2	212,6	6	10	0	0

best individual to next population was applied. The program was stopped after 1000 iterations without improvement. Computations were repeated 10 times. Analogous simulations were performed starting with populations from Nearest Neighbour (NN) algorithm (they are not reported here).

In our experiments variable program parameters included a coding scheme, type of crossover and mutation. We tested two codings with standard recombination operators (Ordinal (Ord), Coset) and Path codings with all eight combinations of four crossovers: OX, CX, PMX, Grefenstette Heuristic (GH) [6] and two mutations: Inversion (I) and Transposition (T).

Standard operators work better with Coset representation then with Ordinal, however both are inefficient. The problem–oriented Path representation is always superiour. The fastest and most efficient operators for large N are simple CX with I when the number of iteratios is enough high. Complex GH is very slow but under our assumptions gives best results for relatively small values of N. OX becomes slow with large N. Inversion mutation is usually better then transposition. Results of experiments are shown in Table 3.

We used the following additional conventions:
NN - Nearest Neighbour tour (typical),
B - optimal/suboptimal tour computed in additional experiments:
for N=100 with settings=(Path,H,T,random initial population)
for N=200 with settings=(Path,PMX,I,initial population from NN algorithm)
for N=500 with settings=(Path,CX,I,initial population from NN algorithm).

Table 3. Comparison of average solutions of TSP for various evolutionary program settings (random initial population)

Repre	Cross	Muta	Average tour lenght			Average # iterations		
-senta	-over	-tion	N=100	N=200	N=500	N=100	N=200	N=500
-tion			NN=8665	NN=13014	NN=20119			
			B=7638	B=11020	B=17684			
Coset	Std	Std	22510	44143	112053	4938	9722	18945
Ord	Std	Std	26190	56031	159375	7051	15908	26544
Path	OX	I	8182	12703	31737	11714	33113	83508
Path	OX	T	8629	13448	35785	13296	32603	87662
Path	GH	I	7964	11940	20783	2580	4921	9190
Path	GH	T	7862	12063	20498	2767	3764	6338
Path	PMX	I	8529	12065	20672	7718	22435	69268
Path	PMX	T	13610	23622	50656	10755	33766	113787
Path	CX	I	8270	11932	19967	7513	22597	70113
Path	CX	T	13140	23360	50606	12202	34080	118997

5 Conclusions

The results presented in this paper lead to conclusion that versatile representations with classical crossover and mutation reveal different properties in GA application to exemplary partition and permutation problems. They can be efficiently used for solving partition problem GCP, where general purpose operators SPPX and First Fit with Tournament Selection can work almost as efficiently as dedicated operators. On the other hand similar approach for permutation problem TSP is inefficient although the proposed Coset representation outperforms the well known Ordinal representation. In TSP problem the dedicated Path representation is always superior.

Generalization of the above conclusion on other partition and permutation optimization problems requires confirmation by further research. Problems that do not have specific operators are of particular interest.

6 Acknowledgements

In conducting computer experiments the author used software tools developed at CUT by Paweł Tuszyński and Marcin Kołodziej . Valuable remarks of anonymous referee resulted in improving presentation of the paper and its experimental part.

References

1. Croitoriu C., Luchian H., Gheorghies O., Apetrei A. (2002) A new genetic graph coloring heuristic. Computational Symposium on Graph Coloring and Generalizations COLOR'02. Proc. Int. Conf. Constraint Programming CP'02
2. Dorne R., Hao J-K. (1998) A new genetic local search for graph coloring. Parallel Problem Solving from Nature 1998, LNCS 1498, 745–754
3. Filho G. R., Lorena L. A. N. (2000) Constructive genetic algorithm and column generation: an application to graph coloring, Proc. Asia Pacific Operarions Research Symposium APORS'2000
4. Galinier P., Hao J-K. (1999) Hybrid evolutionary algorithms for graph coloring. J. Combinatorial Optimization, 374–397
5. Garey R., Johnson D. S. (1979) Computers and intractability. A guide to the theory of NP-completeness. Freeman, San Francisco
6. Grefenstette J. J. et al. (1985) Genetic algorithm for the TSP. Proc. 1st Int. Conf. on Genetic Algorithms, 160–168
7. Johnson D. S., Trick M. A. (1996) Cliques, coloring and satisfiability: Second DIMACS Implementation Challenge. DIMACS Series in Discr. Math. and Theor. Comp. Sc. **26**
8. Khuri S., Walters T. Sugono Y. (2000) Grouping genetic algorithm for coloring edges of graph. Proc. 2000 ACM Symposium on Applied Computing, 422–427
9. Kokosiński Z. (1999) A chromosome representation of permutations for genetic algorithms. Proc. International Conference on Artificial Intelligence ICAI'1999, Las Vegas, CSREA Press, 66–69
10. Kokosiński Z., Kolodziej M., Kwarciany K. (2004) Parallel genetic algorithm for graph coloring problem. Proc. International Conference on Computational Science ICCS'2004, LNCS **3036**, 215–222
11. Kubale, M. (ed.)(2004) Graph colorings. American Mathematical Society
12. Lorena L. A. N., Filho G. R. (1997) Constructive genetic algorithm for graph coloring. Proc. Asia Pacific Operarions Research Symposium APORS'1997
13. Michalewicz Z.: Genetic Algorithms + Data Structures = Evolution Programs, Springer-Verlag (1996)
14. Murata T., Ischibuchi H. (1996) Positive and negative combination effect of crossover and mutation operators in sequencing problems. Proc. Int. Conf. on Evolutionary Computation ICEC'96, IEEE Computer Society Press, 170–175
15. Oliver M., Smith D. J., Holland J. R. C. (1987) A study of permutation crossover operators on the travelling salesman problem, Proc. 2nd Int. Conf. on Genetic Algorithms and Their Application, 224–230
16. Syswerda G. (1991) Scheduling optimization using genetic algorithms. [in:] Davis L. (ed.) Handbook of Genetic Algorithms. Van Nostrand Reinhold, 332–349
17. Üçoluk G. (1997) A method for chromosome handling of r–permutations of n–element set in genetic algorithms. Proc. 4th Int. Conf. on Evolutionary Computing ICEC'97, IEEE Computer Society, 55–58
18. de Werra D. (1990) Heuristics for graph coloring. [in:] Tinhofer G. et all. (eds.) Computational graph theory. Springer–Verlag, 191–208
19. Whitley D., Starweather T., Fuquay D. A. (1989) Scheduling problems and travelling salesman: The genetic edge recombination operator. Proc. 3rd Int. Conf. on Genetic Algorithms, 133–140
20. http://mat.gsia.cmu.edu/COLOR/instances.html

Global Induction of Oblique Decision Trees: An Evolutionary Approach

Marek Krętowski and Marek Grześ

Faculty of Computer Science, Białystok Technical University, Wiejska 45A, 15-351 Białystok, Poland

Abstract. A new evolutionary algorithm for induction of oblique decision trees is proposed. In contrast to the classical top-down approach, it searches for the whole tree at the moment. Specialized genetic operators are developed, which enable modifying both the tree structure and the splitting hyper-planes in non-terminal nodes. The problem of over-fitting can be avoided thanks to suitably defined fitness function. Experimental results on both synthetical and real-life data are presented and compared with obtained by the state-of-the-art decision tree systems.

1 Introduction

There exists dozens of decision tree induction algorithms [13]. However, a wide diversity among decision tree systems is somehow seeming. Almost all approaches are based on top-down strategy, where the learning set is associated with the root node and recursive procedure of optimal split searching, sub-nodes creation and feature vectors redistribution is applied until the stop condition is met. This greedy search technique is fast, easy to implement and, what is probably the most important, efficient in practical situations. On the other hand, it is evident that for many problems, the top-down induction fails to find the optimal solution and more sophisticated methods can be indispensable. However, it should be clearly stated that more complex algorithms are generally more time consuming.

In this paper, a global approach to induction of decision trees is investigated. In contrast to the classical step-wise manner of tree building, the whole tree (its structure and all splits) is searched at the time. Moreover, the top-down approach is often combined with the post-pruning applied in order to prevent the over-fitting problem. In the proposed method, any restructuring procedure is not necessary, because the assurance of the optimal tree size is embedded into the search process. As a result, the global approach allows to find suitable trees, both in terms of the classification power and the complexity.

The proposed method consists in designing a specialized evolutionary algorithm for decision tree building. Evolutionary algorithms (EA) are stochastic, search techniques, which were inspired by the process of biological evolution [11]. Their main advantage is ability to avoid the local optima, which

is especially important in such a difficult optimization problem as induction of decision trees.

The simplest type of decision trees is called *univariate*, because tests in non-terminal nodes are based on single attributes. Such a test is equivalent to partitioning the feature space with an axis-parallel hyper-plane. In case of non-axis parallel decision border, applying only univariate test can lead to their approximation by a very complicated stair-like structure. The aforementioned problem is eliminated by *oblique* (perceptron, linear) trees, where linear combinations of attributes are utilized in tests. The first such a system was CART [6], but it had a strong preference for univariate tests. One of the most well-known oblique tree system is *Oblique Classifier 1* (OC1) [12], which combines deterministic and randomized techniques in search for optimal splits. Other interesting top-down based oblique tree systems were proposed by Gama *et al.* [9] and Bobrowski *et al.* [3]. APDT (*Alopex Perceptron Decision Tree*) [16] can be seen as a first step toward more global induction of decision trees, because it evaluates goodness of a split based also on the degree of linear separability of sub-nodes.

The first application of evolutionary approach to linear tree induction was done in BTGA (*Binary Tree-Genetic Algorithm*) system [8], where standard genetic algorithm (SGA) with binary representation was used to find a splitting hyper-plane in each non-terminal node. In [7], the original OC1 system was successfully extended by using two standard algorithms: (1+1) evolution strategy and SGA. The system described in [10] utilized the specialized evolutionary algorithm for optimization of hyper-plane locations based on the dipolar criteria. After the greedy recursive partitioning the potentially over-specialized decision tree is post-pruned. That system can be treated as a top-down predecessor of the approach proposed in this paper.

A global approach to decision tree induction was investigated in genetic programming (GP) community. Nikolaev *at el.* applied [14] standard GP framework with specialized fitness function to evolve univariate decision tree-like programs. In [5] GP allows to induce classification trees with limited oblique splits.

The rest of the paper is organized as follows. In the next section the proposed evolutionary algorithm is detailed. Its experimental validation on both artificial and real-life problems is described in the section 3. In the last section, conclusions and plans for future work are presented.

2 Evolutionary Algorithm for Global Induction of Oblique Decision Trees

General structure of the proposed evolutionary algorithm follows the typical framework [11] and only application-specific issues are described in this section.

2.1 Preliminaries

A learning set is composed of M objects: N-dimensional feature vectors $\mathbf{x}^j = [x_1^j, ..., x_N^j]^T$ $(j = 1, ..., M)(\mathbf{x}^j \in R^N)$ belonging to one of K classes. The feature space could be divided into two regions by a hyper-plane:

$$H(\mathbf{w}, \theta) = \{\mathbf{x} : \langle \mathbf{w}, \mathbf{x} \rangle = \theta\}, \tag{1}$$

where $\mathbf{w} = [w_1, ..., w_N]$ $(\mathbf{w} \in R^N)$ is a weight vector, θ is a threshold and $\langle \mathbf{w}, \mathbf{x} \rangle$ represents an inner product. If $\langle \mathbf{w}, \mathbf{x}^i \rangle - \theta > 0$, it can be said that the feature vector $\mathbf{x}^i$ is on the positive side of the hyper-plane $H(\mathbf{w}, \theta)$.

A *dipole* [4] is a pair $(\mathbf{x}^i, \mathbf{x}^j)$ of feature vectors. A dipole is called *mixed* if and only if feature vectors constituting it belong to different classes and a pair of the vectors from the same class constitutes *pure* dipole. Hyper-plane $H(\mathbf{w}, \theta)$ splits the dipole $(\mathbf{x}^i, \mathbf{x}^j)$ if and only if:

$$(\langle \mathbf{w}, \mathbf{x}^i \rangle - \theta) \cdot (\langle \mathbf{w}, \mathbf{x}^j \rangle - \theta) < 0 \tag{2}$$

It means that the input vectors $\mathbf{x}^i$ and $\mathbf{x}^j$ are situated on the opposite sides of the dividing hyper-plane.

2.2 Representation, Initialization and Termination Condition

An oblique decision tree is a binary tree with splitting hyper-planes in non-terminal nodes and class labels in leaves. Each hyper-plane in the tree can be represented by a fixed-size $N + 1$-dimensional table of real numbers corresponding to the weight vector $\mathbf{w}$ and the threshold θ. However, the size and the structure of the classifier for a given learning set cannot be known in advance of induction. In such a situation, a variable-length representation is indispensable. Furthermore, during the induction process, additional information concerning, among other things, the learning vectors associated with each node are necessary. As a result, decision trees are not especially encoded in the form of traditional (binary or real-valued) chromosomes and they are represented in their actual form in the presented system.

An initial population is created by applying for each individual the following simple top-down algorithm combined with selection of optimal tree size according to the fitness function. An effective test in non-terminal nodes is searched based on randomly chosen mixed dipole $(\mathbf{x}^i, \mathbf{x}^j)$. The hyper-plane $H_{ij}(\mathbf{w}, \theta)$ is placed to split it:

$$\mathbf{w} = \mathbf{x}^i - \mathbf{x}^j \qquad \text{and} \qquad \theta = \delta \cdot \langle \mathbf{w}, \mathbf{x}^i \rangle + (1 - \delta) \cdot \langle \mathbf{w}, \mathbf{x}^j \rangle, \tag{3}$$

where $\delta \in (0, 1)$ is a randomly drawn coefficient, which determines the distance to the opposite ends of the dipole. $H_{ij}(\mathbf{w}, \theta)$ is perpendicular to the segment connecting dipole ends.

The EA terminates if the fitness of the best individual does not improve during the fixed number of generations (default value is equal to 1000) or the maximum number of generations is reached (default value: 10000).

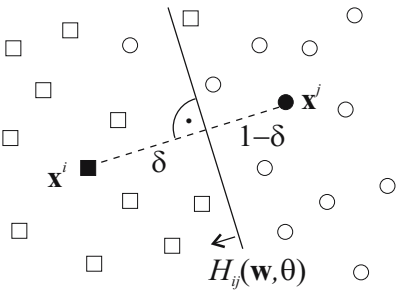


Fig. 1. Hyper-plane initialization based on randomly chosen mixed dipol

2.3 Fitness Function

The ultimate goal of any classification system is correct prediction of classes for new unlabelled feature vectors, which were not accessible during the learning phase. It is obvious that such a target function cannot be directly defined. Instead the classification quality on the training data is often applied as an estimate measure of the predictive power of the classifier. However, it should be underlined that optimizing only the re-classification quality leads to the over-fitting problem. In classical systems this problem is usually mitigated by post-pruning techniques. In our approach another solution is proposed. A complexity term is introduced into the fitness function. This term works as a certain type of penalty, which is proportional to the size of the tree. This way more compact trees are promoted and it allows avoiding the over-specialization.

Finally, the fitness function, which is maximized, has the following form:

$$Fitness(T) = Q_{Reclass}(T) - \alpha \cdot S(T), \tag{4}$$

where $Q_{Reclass}(T)$ is the re-classification quality, $S(T)$ is the size of the tree T expressed as a number of nodes and α - is a relative importance of the complexity term (default value is 0.005) and a user supplied parameter. It is rather obvious that there is no one, optimal value of α for all possible dataset. When the concrete problem is analyzed, tuning this parameter may lead to the improvement of the results (in terms of accuracy or classifier complexity).

2.4 Genetic Operators

It is now commonly accepted opinion, that task specific operators are highly useful in improving optimization process. In our system two complex genetic operators are currently employed: *MutateNode* and *CrossoverTrees*. Both of them can alter the tree structure as well as splitting hyper-planes in non-terminal nodes.

The first composite operator can be seen as a combination of the typical mutation-like operator with the dipolar operator introduced in [10]. *MutateNode* is applied to every single node of the tree with the given probability

(default value is equal to 0.1). The operator can cause with equal probability a modification of the test or a change of the node role. If a non-terminal node is concerned it can be pruned to a leaf or the corresponding hyper-plane can be altered. The hyper-plane position can be modified due to an application of the dipolar operator or by standard mutation. The dipolar operator chooses at random one dipole from the set of not divided mixed and divided pure dipoles. Then it shifts the hyper-plane by modifying only one randomly chosen weight w_i in such a way that the chosen mixed dipole is divided or division of pure one is avoided. When a leaf is concerned it can be only replaced by a new non-terminal node unless it is not reasonable.

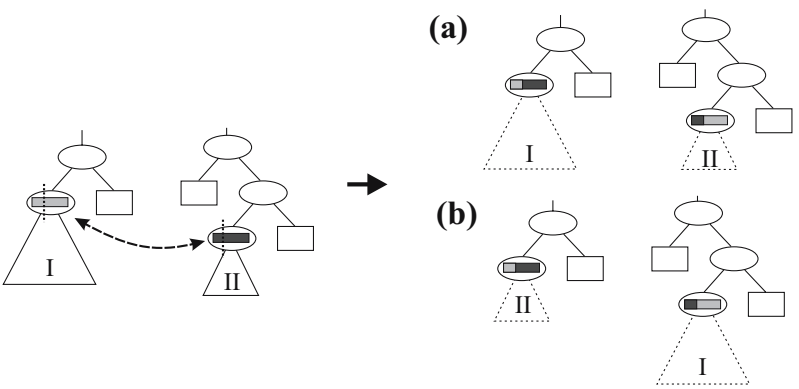


Fig. 2. *CrossoverTrees* operator: (**a**) exchange limited only to tests (**b**) exchange of the whole sub-trees

The second operator is an equivalent of the standard crossover operator and alters two individuals by replacing parts of the input trees. The exchange can be limited only to nodes or can encompass also sub-trees (see Fig. 2). At the beginning, the type of the exchange is randomly drawn (two variants are equally probable) and then, regardless of the type, one node in each tree is chosen also at random. If both nodes are non-terminal ones, the typical one-point crossover is applied on weights vectors and thresholds. In other cases nodes are just substituted. After the exchange concerning the nodes, depending on range of the operator, sub-tree starting from the altered nodes can be also replaced.

It should be underlined that trees modified by genetic operators require renewed determination of locations of all input feature vectors in the affected parts. As a result of this process some parts of the tree can be even pruned, because they do not contain any input feature vectors.

As a selection mechanism the proportional selection with linear scaling is applied. Additionally *elitist* strategy is used, which means that the best

tree in terms of the fitness function in each iteration is copied to the next population.

It was observed by Bennett *et al.* [1] that in oblique trees enlarging the margin, defined as the distance between decision boundary and the input feature vectors, is profitable in term of classifcation accuracy. In the presented system, a simple mechanism based on this observation was introduced. In each non-terminal node, two the closest feature vectors ($\mathbf{x}^+$ and $\mathbf{x}^-$) to the splitting hyper-plane $H(\mathbf{w}, \theta)$ are determined on the opposite sides of it. If the found dipole is mixed, the hyper-plane is centered by modifying the threshold:

$$\theta' = \frac{1}{2}[\langle \mathbf{w}, \mathbf{x}^+ \rangle + \langle \mathbf{w}, \mathbf{x}^- \rangle]. \tag{5}$$

It should be noted that such an operation does not change the fitness function.

3 Experimental Results

In this section experimental validation of the proposed approach on both artificial and real-life datasets is described. All results presented in the tables correspond to averages of 5 runs and were obtained by using test sets (mainly in case of synthetic datasets) or by 10-fold stratified cross-validation. Average number of nodes is given as a complexity measure. For the purpose of the comparison, results obtained by C4.5 (release 8) [15] and OC1 [12] with default parameters are also presented. If not otherwise stated, the proposed system (described as GEA-ODT in tables) is run with default values of parameters.

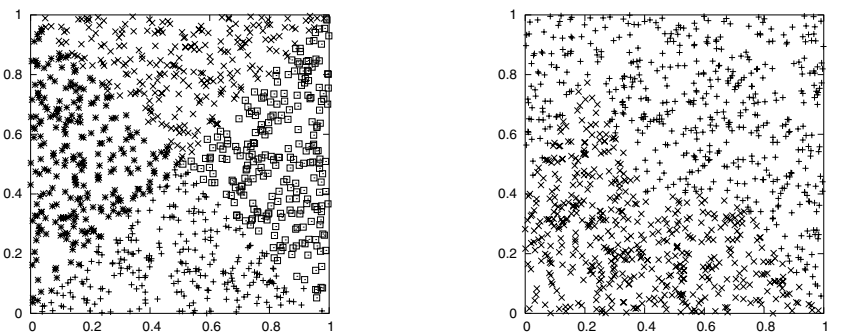


Fig. 3. Examples of artificial datasets (rotated *chessboard4* and *house*)

3.1 Artificial Datasets

In this subsection, synthetical datasets with analytically defined decision borders are analyzed. Analogous experiments are described in [12] and [16],

but original datasets are not available, hence similar configurations were generated by using random number generator. All these datasets are two-dimensional, except LS10 problem which is defined with 10 features. Number of feature vectors in the learning sets is 1000. Examples of synthetical datasets are depicted in Fig. 3.

Before presenting the definitive results, two experiments with parameters of the proposed system are presented. In the first one, usefulness of introducing the dipolar operator is investigated on the 2-class rotated chessboard problem. In Fig. 4 two learning curves (in fact, their initial parts) corresponding to induction without and with the use of dipolar operator are illustrated. As it could be expected applying the dipolar-based approach allows to speed up the search process.

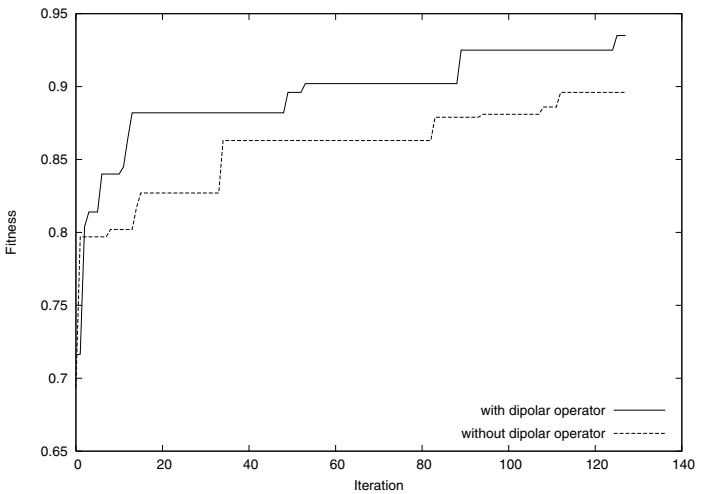


Fig. 4. Results of initial experiments on *chessboard2* problem: comparison of learning curves with and without dipolar operator

Sore point of any decision tree induction method is finding appropriate balance between re-classification quality and generalization power related to the tree complexity. In the global approach parametr α from the fitness function seems crucial for this problem. In the next experiment, we investigated how the classification quality and the tree complexity are influenced by changes of α. The relationships obtained on *chessboard2* dataset are presented in Fig. 5.

It can be observed that for relatively broad range of values (0.05-0.001) optimal trees were found. Further decrease of a parameter results in performance deterioration in terms of tree simplicity.

The final results of experiments concerning synthetical datasets are gathered in Table 1. For all domains GEA-ODT perform very well, both in accu-

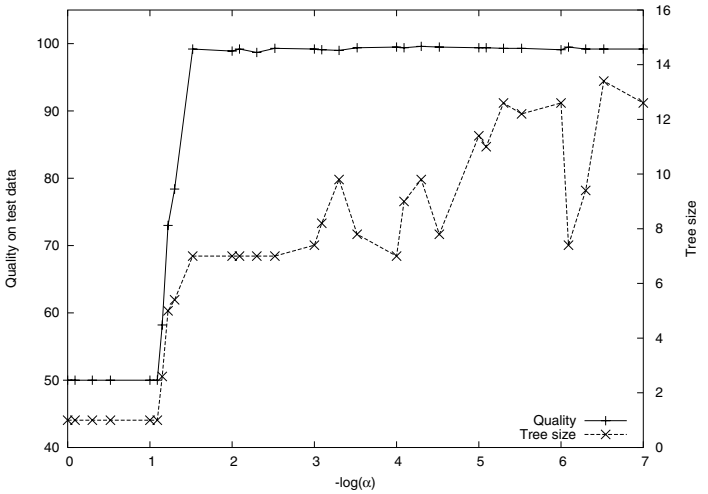


Fig. 5. Results of initial experiments on *chessboard2* problem: impact of α on the classification quality and the tree complexity

Table 1. Experimental results obtained for artificial datasets

Dataset	GEA-ODT Quality	GEA-ODT Size	OC1 Quality	OC1 Size	C4.5 Quality	C4.5 Size
chessboard2	99.1	7	99.3	11	93.3	47
chessboard4	99.3	7	99.5	7	94.6	87
zebra1	98.5	7	98.2	15	50	1
zebra2	99.0	5	99.8	7	96.9	55
zebra3	98.2	15.8	95.1	29	91.6	99
house	95.8	5.4	95.9	5	98	39
LS10	92.6	4.2	96	7	76.2	295

racy and tree complexity. It could be observed that the global method was able to find slightly less complicated trees than generated by other methods.

3.2 Real-life Datasets

In the second series of experiments a few well-known real-life datasets taken from UCI Machine Learning Repository [2] were analyzed and obtained results are collected in Table 2.

The proposed system performed very well on almost all analyzed datasets. Only for vehicle domain it was significantly worse that OC1 and C4.5. For this

dataset EA is converging very slowly and it was observed that the maximum number of iteration was used to stop the algorithm. When this constraint is relaxed the quality raise to more than 69%, but the computation time is really long (a few hours). Concerning complexities of the trees, it can be found that the global EA approach is generally more efficient than other systems.

Table 2. Results obtained for UCI datasets

Dataset	GEA-ODT Quality	GEA-ODT Size	OC1 Quality	OC1 Size	C4.5 Quality	C4.5 Size
breast-w	96.7	3	95.3	5	94.9	26
bupa	67.7	4.9	67.5	12.9	64.7	44.6
iris	97.0	5	96.6	5	94.7	8.4
page-blocks	94.6	3.7	97	23	97	82.2
pima	73.5	3.2	72.6	9.1	74.6	40.6
sat	83.1	11	85.4	45	85.5	435
vehicle	65.4	14.7	70.2	30.4	72.3	129
waveform	81.5	6.2	78	5	73.5	107

4 Conclusions

In the paper, new evolutionary algorithm for induction of oblique decision trees is presented. The greedy top-down technique is replaced by the global approach, where the whole tree is searched at the moment. The experimental validation indicates that the accuracy of the proposed method is at least comparable with the results obtained by leading decision tree systems. In terms of the tree complexity it seems that global algorithm is able to find more compact classifiers than the competitors.

The presented system is constantly improved and currently feature selection is embedded into the algorithm, which will allow eliminating redundant and noisy features at each non-terminal node. Furthermore several directions of possible future research exist. One of them is designing more robust fitness function, which has a critical influence on the performance of the system. We also want to incorporate into the induction process the variable misclassification cost and feature's cost.

The proposed approach is not the fastest one now but hopefully it is known that evolutionary algorithms are well suited for parallel architecture. We plan to speed up our system by re-implementing it in the distributed environment.

Acknowledgments This work was supported by the grant W/WI/05 from Białystok Technical University.

References

1. Bennett, K., Cristianini, N. et al. (2000) Enlarging the margins in perceptron decision trees. Machine Learning **41**, 295–313
2. Blake, C., Keogh, E. et al. (1998) UCI repository of machine learning databases,[http://www.ics.uci.edu/˜mlearn/MLRepository.html]. Irvine, CA: University of California, Dept. of Computer Science
3. Bobrowski, L., Krętowski, M. (2000) Induction of multivariate decision trees by using dipolar criteria. In: Proc. of PKDD'00. Springer LNCS 1910, 331–336
4. Bobrowski, L. (1996) Piecewise-linear classifiers, formal neurons and separability of the learning sets, In: Proc. of 13^{th} Int. Conf. on Pattern Recognition 224–228
5. Bot, M., Langdon, W. (2000) Application of genetic programming to induction of linear classification trees. In: EuroGP 2000. Springer LNCS 1802, 247–258
6. Breiman, L., Friedman, J., Olshen, R., Stone C. (1984) Classification and Regression Trees. Wadsworth Int. Group
7. Cantu-Paz, E., Kamath, C. (2003) Inducing oblique decision trees with evolutionary algorithms. IEEE Trans. on Evolutionary Computation **7**(1), 54–68
8. Chai, B., Huang, T. et al. (1996) Piecewise-linear classifiers using binary tree structure and genetic algorithm. Pattern Recognition **29**(11), 1905–1917
9. Gama, J., Brazdil, P. (1999) Linear tree. Inteligent Data Analysis **3**(1), 1–22
10. Krętowski, M. (2004) An evolutionary algorithm for oblique decision tree induction, In: Proc. of ICAISC'04. Springer LNCS 3070, 432–437
11. Michalewicz, Z. (1996) Genetic Algorithms + Data Structures = Evolution Programs. 3^{rd} edn. Springer
12. Murthy, S., Kasif, S. et al. (1994) A system for induction of oblique decision trees. Journal of Artificial Intelligence Research **2**, 1–33
13. Murthy, S. (1998) Automatic construction of decision trees from data: A multidisciplinary survey. Data Mining and Knowledge Discovery **2**, 345–389
14. Nikolaev, N., Slavov, V. (1998) Inductive genetic programming with decision trees. Intelligent Data Analysis **2**, 31–44
15. Quinlan, J. (1993) C4.5: Programs for Machine Learning. Morgan Kaufmann
16. Shah, S., Sastry, P. (1999) New algorithms for learning and pruning oblique decision trees. IEEE Trans. on Systems, Man, and Cybernetics - Part C **29**(2), 494–505

Nature-Inspired Algorithms for the TSP

Jarosław Skaruz[1], Franciszek Seredyński[1,2,3], and Michał Gamus[4]

[1] Institute of Computer Science, University of Podlasie, Sienkiewicza 51, 08-110 Siedlce, Poland
[2] Polish-Japanese Institute of Information Technology, Koszykowa 86, 02-008 Warsaw, Poland
[3] Institute of Computer Science, Polish Academy of Sciences, Ordona 21, 01-237 Warsaw, Poland
[4] Warsaw University of Technology, Nowowiejska 15/19, 00-665 Warsaw, Poland

Abstract. Three nature-inspired algorithms are applied to solve Travelling Salesman Problem (TSP). The first originally developed Multi-agent Evolutionary Algorithm (MAEA) is based on multi-agent interpretation of TSP problem. An agent is assigned to a single city and builds locally its neighbourhood — a subset of cities, which are considered as local candidates to a global solution of TSP. Creating cycles — global solutions of TSP is based on Ant Colonies (AC) paradigm. Found cycles are placed in Global Table and are evaluated by genetic algorithm (GA) to modify a rank of cities in local neighbourhood. MAEA is compared with two another algorithms: artificial immune — based system (AIS) and a standard AC — both applied to TSP. We present experimental results showing that MAEA outperforms both AIS and AC algorithms.

1 Introduction

TSP has many applications and has been studied for years in the literature. TSP solutions could be useful for various domains such as airlines, post offices, drilling circuit boards, scheduling, VLSI chip fabrication [5]. TSP belongs to NP hard problem family. Therefore, algorithms of artificial intelligence are used to receive approximate results. During the last few years we could observe the increase of interest in TSP that is reflected in published articles representing various approaches to this problem.

Some researches emphasise GA and AC approaches as being very promising for the TSP. To the most known results obtained using GA approach are those obtained by [13] due to developing EAX (Edge Assembly Crossover) operator. This approach gives results with very low error margin in comparison with Kyobashis's algorithm [9]. In [7] the authors improved GLS algorithm for GA receiving very good results for various problems taken from TSPLIB library. On the other side Gambardella and Dorigo [3] proposed Ant-Q algorithm based on the Q-learning algorithm [12]. In particular they drew attention in their approach to the importance of cooperation between the agents. In [11] the authors proposed MAX-MIN Ant system, which is a modification of AC. In that algorithm they only allow the best ant in each

iteration to update the trial intensity. They also set minimum and maximum value for an edge. The results obtained by them are very promising and are also much better than previously received results which made use of other algorithms like AC.

In this paper we emphasise on multi-agent approach to solve the TSP, and we use both AC and GA approaches. We developed an original MAEA, where agents initially build local neighbourhoods, used to build cycles. We use modified AC algorithm to build cycles and GA to evaluate the rank of cities in local neighbourhoods. We compare our algorithm with both AC and AIS-based approaches.

The paper is organized as follows. In the next section we present TSP. Section 3 contains presentation of MAEA. In section 4 two other algorithms AIS and AC are presented in the context of TSP. Section 5 contains experimental results. Last section summarizes obtained results.

2 Travelling Salesmen Problem

What exactly is Travelling Salesman Problem? Let us consider a person who starts a tour from his home city and visits *n-1* cities only once and then returns home. Since the person is interested in finding the shortest route, this problem corresponds to the shortest Hamilton's cycle in a complete graph. Formally speaking, the tour length [6]

$$l(\pi) = \sum_{i=1}^{n-1} d_{\pi(i),\pi(i+1)} + d_{\pi(n),\pi(1)}, \tag{1}$$

has to be minimized, where d_{ij} is the distance between cities i and j and π is a permutation of <1, 2, ..., n> while the solution is a vector π with $j = \pi$(i) denoting city j to visit at *i-th* step.

Several modifications of TSP exists in the literature. There are two main varieties: asymmetric travelling salesman problem (ATSP) [4,14] and symmetric travelling salesman problem (STSP) [8,10], which is considered in our research. The difference between two approaches is that in the first one the distance between one city and another is not the same as in the opposite direction. Fekete and Fleischer [2] considered another modification of TSP, namely Competing Salesman Problem (CSP), which is a 2-player version of STSP. Each player must visit more places while both know their current positions. A player wins when he reaches more cities then the other.

3 Multi-agent Evolutionary Algorithm

3.1 The Algorithm Concept

Our algorithm uses a number of agents, each of them assigned to a city. Next, an agent builds its own neighbourhood. This is a subset of cities which

are considered to be appropriate to the global solution. The main task of each agent is to build a cycle choosing a city from its neighbourhood, with respect to the weight of the city. Then local solution is put on the Global Table, where rank of cities is evaluated. The algorithm employs also 3-opt and crosselimination algorithms, which improve local solutions. The 3-opt cuts randomly a tour in three points and then three fragments are linked in the best way in terms of the length of a tour. The second algorithm assures that there are not any route between two cities which crosses another route. The algorithm is completed when all agents visited edges in the same order. Description of the algorithm is presented in a pseudo-code below.

```
FOR every agent
     Choose cities for neighbourhood
END
N:=number of steps
FOR step:=1 TO N
     Create a cycle
     IF crosselimination enabled THEN
          Run Crosselimination
     Put a cycle on the Global Table
     Modify weight of agents
     Improve local solution by GA
     IF 3-opt enabled THEN
          Run 3-opt
END
```

3.2 Building Local Neighbourhood

Neighbourhood setup is a critical point of the algorithm. The first problem is a number of agents that should constitute a neighbourhood. On the one hand, the less agents the lesser complexity of algorithm. However, it could lead to elimination of edges that could be better than others. The key to success is to define such a neighbourhood which is of quite small size and includes the best agents. Since we have no information about geometrical position of the cities, evaluations must be based only on distances between cities. To solve the above problem "shadow zone" notion comes to its aid. Its definition is stated as follows: "shadow zone is an area containing all these agents which have to be eliminated from neighbourhood". In other words, if the direct distance between a city, which is considered as a candidate and a city whose neighbourhood is being created now and is a bit shorter or equal to the distance through the just determined city, then neighbourhood will not include this considered city.

A candidate (C city) is discarded according to the following formula:

$$\frac{d_{CB} + d_{BA}}{d_{CA}} < p * d_{CB} < d_{CA}, \tag{2}$$

where: A — a city which neighbourhood is being created, B — neighbouring city, C — candidate city, d_{CB} — the distance between cities C and B, d_{BA} — the distance between cities B and A, d_{CA} — the distance between cities C and A, p — parameter.

The size of the area is controlled by p parameter. The grater p is, the more cities are eliminated. The process of neighbourhood setup can be described as follows: for each city sort remaining cities by distance, the city at the top of the list belongs to the neighbourhood and eliminates these cities which are included in its shadow zone, the previous step is repeated starting with the first not eliminated city until all cities have been considered. The number of cities of just created neighbourhood may be restricted. If only one city belongs to the neighbourhood then shadow zone must be narrowed until it adds another city. The process of neighbourhood setup is presented in the figure Fig. 1.

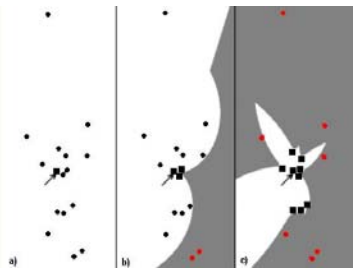


Fig. 1. The process of neighbourhood setup.

The square-shaped point in the part a) of the figure Fig. 1 means a city for which neighbourhood is created. In the next step two cities (square shaped points) constitute the neighbourhood and two other cities (in the dark area) are included in the shadow zone. Part c) of the figure Fig. 1 presents final point of the process — seven cities (in the dark area) were excluded from the neighbourhood.

3.3 Creating Cycles

Building a cycle is quite an easy task. At the beginning each agent inserts itself into an empty list at the first position. In the next steps of the algorithm lists are sent to their neighbours, which are attached to the list and this process is repeated until the last city has been reached. Choosing an edge is made according to the following formula:

$$p_i = \frac{T_i * \left(\frac{1}{d_{ki}}\right)^{\alpha}}{\sum_{z \notin P \wedge z \in n_k} T_z * \left(\frac{1}{d_{kz}}\right)^{\alpha}}, \tag{3}$$

where: p_i — probability of choosing i city, P — the list containing just visited cities, T_i — quality of *i-th* neighbour (see 3.4), d_{ki} — the distance between k city and i neighbour, d_{kz} — the distance between k city and z neighbour, α — parameter, n_k — set of neighbours of k city.

Agents are able to cooperate with each other. This can be understood as considering additional criterion, which is a quality of a neighbour chosen by each agent from its own point of view. Probability of the choice of the edge in respect to cooperation between agents is defined below:

$$p_i = \frac{T_i * \eta_i \left(\frac{1}{d_{ki}}\right)^{\alpha}}{\sum_{z \notin P \wedge z \in N} T_i * \eta_i \left(\frac{1}{d_{kz}}\right)^{\alpha}}, \tag{4}$$

where η — the distance quality (weight of a route) in *i-th* neighbour point of view.

3.4 Global Table

The Global Table includes some completed cycles. Given a set of temporary solutions, edges quality modification must be done. We can estimate all agents in one of its point of view and examine this relationship in two ways: is there any cycle which includes the given neighbour; the ratio of average distance of all cycles to the average distance of these cycles where an agent was used? Usefulness of an agent can be concluded from the answer on the first question and it is the basis of agent quality increase or decrease. Weight modification is made according the formula:

$$T_i^{mod} = T_i * \beta, \tag{5}$$

where: T_i^{mod} — quality of *i-th* neighbour, T_i — quality of *i-th* neighbour before modification, β — parameter range from 0 to 1. The second dependence constitute another possibility to find out how suitable an agent is and can be described as follows:

$$T_i^{mod} = T_i * \frac{L_{sr}}{L_{sr}^i} * \beta, \tag{6}$$

where: T_i^{mod} — quality of *i-th* neighbour, T_i — quality of *i-th* neighbour before modification, β — parameter range from 0 to 1, L_{sr} — the average distance of cycle, L_{sr}^i — the average distance of cycle that used *i-th* neighbour.

We employ GA to improve results obtained from the table. We used ranking selection, mutation inversion and crossover with edges recombination. Set of new evaluated cycles can be changed fully or partially with those which are in the table.

4 Artificial Immune System and Ant Colony Algorithm

As a point of reference AIS takes advantage of a human immune system. In comparison with other artificial intelligence algorithms such as GA or neural network (NN) it is a quite new approach. There are a lot of applications of AIS: pattern recognition, anomaly detection, optimisation problems. Solving a problem involves creating its model and finally applying to AIS. Probably, the most useful algorithm for AIS is a well-known CLONALG [1]. In this paper we show a few modifications of CLONALG, which have been made to make it the most appropriate to TSP.

In our implementation each antibody represents a potential solution of the problem in the form of a list of n ordered cities. To find a shorter route (better solution) these lists are modified according to the algorithm steps. Meaning of coefficients is the following: v — cloning multiplier (best Antibody is cloned v times), m — mutation multiplier (worst Antibody is mutated on m % positions), dg — number of iterations. Description of the algorithm is presented in pseudo-code below.

```
FOR number of iterations
    Initialise population (at random the N of antibodies);
    FOR every antibody
        Count the affinity; (measure of the route length)
    END
    Sort antibodies by affinity
    Choose the n the best antibodies of population to new
population C
    FOR every antibody C
        Cloning(v);
        Hipermutation(m);
    END
    IF shortest path enabled THEN
        Shortest path;
    FOR every antibody C
        Increase its age(f);
    END
    IF crossover enabled THEN
        Crossover;
    IF disrupting enabled and number of step iteration
 MOD dg = 0 THEN
        Save ds best antibodies
    IF short mutation enabled THEN
        Short mutation;
    IF exclusive mutation enabled THEN
        Exclusive mutation
    IF local mutation enabled THEN
```

```
        Local Mutation
    IF Standard mutation enabled THEN
        Standard mutation
    Replace d antibodies with the new ones
END
```

The algorithm allows to choose various types of mutations, but the results with only standard mutation are reported in the paper. Other modifications are disrupting option which save *ds* best antibodies every *dg* iterations and shortest path option, which very quickly results in much better routes. It gives the greatest effects but sometimes leads to local optimum instead of global.

There exists three groups of AC: ANT-density, ANT-quantity and ANT-cycle. In all these algorithms pheromone can be placed on all edges. Probability of choosing an edge depends on: length of the edge, value of pheromone on this edge, list of visited cities and is evaluated according the following formula for all kinds of AC:

$$p_{ij} = \frac{T_{ij}^{\alpha}\left(\frac{1}{d_{ij}}\right)^{\beta}}{\sum_{z \notin P} T_{iz}^{\alpha}\left(\frac{1}{d_{iz}}\right)^{\beta}}, \tag{7}$$

where: p_{ij} — probability of choosing ij edge, T_{ij} — value of pheromone on ij edge, d_{ij} — length ij edge, α and β — parameters. AC algorithms were classified for the sake of different way of pheromone modifying. For ANT-density this is done according to the formula:

$$\delta_{ij}^{k} = \begin{cases} Q & \text{if ij edge belongs to a cycle created by k ant;} \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

where: Q-parameter, δij^k — increase of pheromone for k ant and ij edge. In this approach each ant increases pheromone on all visited edges with the same value. For ANT-quantity each ant increases pheromone inversely proportionally to the length of the edge. Only for ANT-cycle pheromone modification is made after a cycle has built. If an ant visits an edge then pheromone is increased of quotient a parameter by the length of a created cycle.

5 Experimental Results

All experiments were performed using *berlin52, kroA100, lin105, tsp225* and *st70* problems accessible at http://www.iwr.uni-heidelberg.de/iwr/comopt/software/TSPLIB95. Firstly, we investigated what values of parameters are the best for chosen problems in terms of effectiveness of the algorithm. Knowing this, we did some experiments to find out which algorithm is the best.

At first we tuned the parameters of algorithms and then compare the obtained results. The best values of parameters of immune system are d=10, f=10, n=50, v=24, m=20, ds=10, dg=10, enabled shortest path option. The greater the v value the better results would be obtained but more time was needed. Each experiment consisted of 20 tests and 200 iterations in each.

Testing multi-agent evolutionary algorithm we checked the results in five different cases. For each problem we did 20 tests with the following options enabled: improvement temporary solutions in the table, improvement temporary solutions in the table and 3-opt, improvement temporary solutions in the table, 3-opt and crosselimination, crosselimination, and without additional algorithms, but only results for improvement temporary solutions in the table, 3-opt and crosselimination are shown. Ant colony algorithm was run with α=0.6, β=1, ro = 0.5, and the number of ants was the same as the number of cities. All values on graphs presented in the figures Fig. 2 and Fig. 3 are average of 20 tests of each problem. It is easy to notice that especially parts a) and c) of the figure Fig. 2 show big difference between results of MAEA and results of two other algorithms.

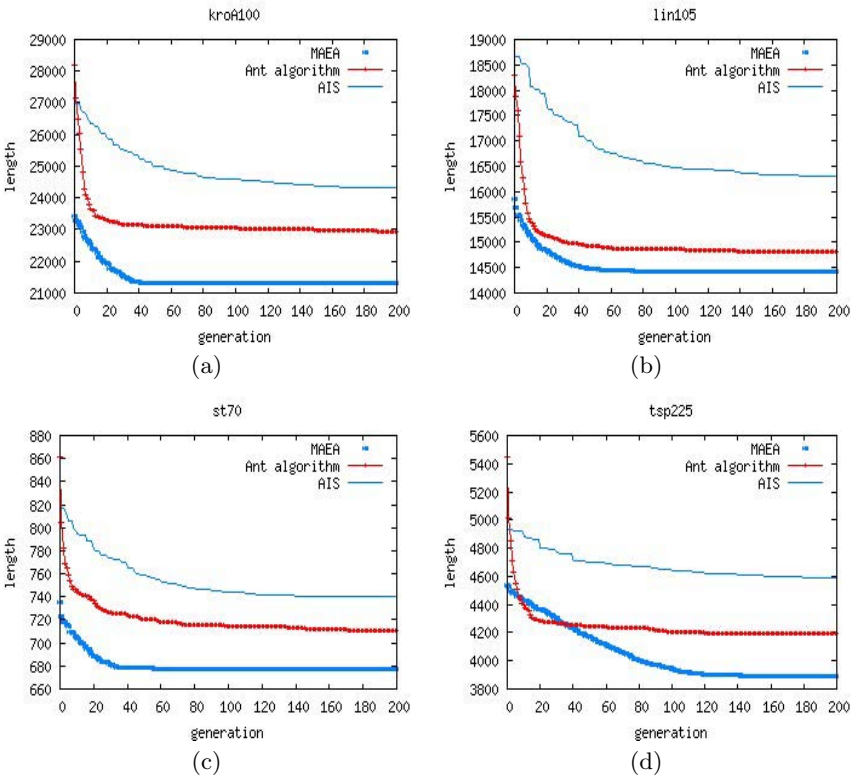


Fig. 2. kroA100 (a), lin105 (b), st70 (c), tsp225(d)

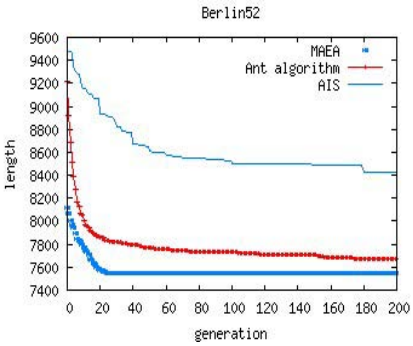


Fig. 3. berlin52

To compare algorithms presented in this paper we assembled the shortest routes from all experiments and presented them in Table 1. Values in brackets are the best one, while others are average.

Table 1. Algorithms results comparison

Problem	MAEA	AC	AIS	Optimum	Best -optimum (%)
berlin52	**7544,36** [**7544,36**]	7678,49 [7548,99]	8270,27 [8060,44]	7542	0,031
st70	**677,42** [**677,1**]	711,54 [701,5]	733,17 [707,82]	675	0,31
kroA100	**21293,69** [**21285,4**]	22930,79 [22490,42]	24177,43 [22586,91]	21282	0,015
tsp225	**3897,67** [**3866,7**]	4192,91 [4123,93]	4590,41 [4524,57]	3859	0,19
lin105	**14427,21** [**14382,99**]	14811,61 [14688,49]	16146,11 [15808,31]	14379	0,027

6 Conclusions

New algorithm for the TSP — MAEA has been presented in this paper. It makes use of both multi-agent concept and evaluating paradigms of GA and AC. We did some research and compared results obtained from MAEA with results from AC and AIS. It turned out that MAEA outperforms other algorithms. It generates much better results than results obtained from two other algorithms and the difference between them and the optimum is very small. We reckon that MAEA is a very promising method to solve TSP.

During the research it turned out that functioning of AIS is not so efficient as that of two other algorithms. Much better results could be obtained if shortest path option was enabled. We think that this algorithm should be exactly analysed to find its weak and strong points.

References

1. de Castro L., N., Von Zuben F., J. (2000) The Clonal Selection Algorithm with Engineering Applications. Proc. of the Genetic and Evolutionary Computation Conference, Workshop on Artificial Immune Systems and Their Applications, 36-37
2. Fekete P., S., Fleischer R., Fraenkel A., Shmitt M. Traveling Salesman in the Presence of Competition. Theoretical Computer Science, Vol. **303**, No. 3, 377-392
3. Gambardella L. M., Dorigo M. (1995) Ant-Q: A Reinforcement Learning approach to traveling salesman problem. Proceedings of ML-95, Twelfth International Conference on Machine Learning, 252-260
4. Glover F., Gutin G., Yeo A., Zverovich A. (2001) Construction heuristics for the asymmetric TSP. European Journal of Operational Research **129** 555-568
5. Korte B. (1988) Applications of Combinatorial Optimization. in Talk at the 13th International Mathematical Programming Symposium, Tokyo
6. Merz P. Memetic Algorithms for Combinatorial Optimization Problems: Fitness Landscapes and Effective Search Strategies. PhD thesis, Department of Electrical Engineering and Computer Science, University of Siegen, Germany
7. Merz P., Freisleben B. (1997) Genetic Local Search for the TSP: New Results. Proceedings of The IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence
8. Merz P., Freisleben B. (2001) Memetic Algorithms for the Traveling Salesman Problem. Tech. Rep., Department of Computer Science, University of Siegen, Germany. Accepted for publication in Complex Systems
9. Nagata Y., Kobayashi S. (1997) Edge assembly crossover: A high-power genetic algorithm for the traveling salesman problem. In T. Black, editor, Proc. Of the 7th Int'l. Conf. on GAs, 450-457. Morgan Kaufmann
10. Shin S., Zhang B., Jun S. (1999) Solving Traveling Salesman Problems using Molecular Programming. Proceedings of the Congress on Evolutionary Computation, Washington, DC, Vol. **2**, 994–1000
11. Stutzle T., Hoos H. (1997) Improvements on the Ant-System: Introducing the MAX-MIN Ant System. ICANNGA97 — Third International Conference on Artificial Neural Networks and Genetic Algorithms, University of East Anglia, Norwich, UK
12. Watkins C. Learning from Delayed Rewards. PhD thesis, Psychology Department, Cambridge University, Cambridge, England
13. Watson J., Ross C., Eisele V., Denton J., Bins J., Guerra C., Whitley D., Howe A. (1998) The Traveling Salesrep Problem, Edge Assembly Crossover, and 2-opt. Parallel Problem Solving from Nature — PPSN V, 5th International Conference, Amsterdam, The Netherlands, September 27-30
14. Zhang W. (2004) Phase Transitions and Backbones of the Asymmetric Traveling Salesman Problem. Journal of Artificial Intelligence Research **21**, 471-497

Graph-Based Analysis of Evolutionary Algorithm

Zbigniew Walczak

Warsaw University of Technology, Department of Electronics and Information Technology, ul. Nowowiejska 15/19, Warsaw, Poland,
email: Z.Walczak@elka.pw.edu.pl

Abstract. Evolutionary algorithms work in an algorithmically simple manner but produce a huge amount of data. The extraction of useful information to gain further insight into the state of algorithm is a not-trivial task. In the paper, we propose a method of analysis of evolutionary algorithm by means of a graph theory. The method is inspired by latest results on scale-free network and small world phenomena. The paper presents visualization of evolutionary process based on network visualization software. The properties of such network are analyzed and various research possibilities are discussed.

1 Introduction

Evolutionary algorithms work in an algorithmically simple manner. However, when put to work they produce a vast amount of data. Apart from simple convergence information it is a non trivial task to extract useful information from those data to provide insight into the state, progress of evolutionary algorithms. Several methods for extracting and visualizing important data have been proposed [9], [11], [5], [6], [4], [3]. Most of commonly used techniques for visualization are limited to representing data on one or two variables. Problems solved by evolutionary algorithms are more complicated.

The most popular visualization technique for evolutionary algorithms is to visualize the quality of solutions found in subsequent populations. The best, average and worst solution in each population may be drawn on a single plot. This method could be extended to draw a values of all chromosomes in the population in 3D fitness distribution graph.

Another class of methods is to visualize the best chromosome in a given population. For example the Euclidean Travelling Salesman may be represented on the plane by points and lines connecting them. The disadvantage of this technique is the fact that every problem requires the special visualization software.

Due to large number of dimensions in practical problems, anther popular technique is proposed, i.e. the transformation of higher dimensional search space into the smaller one. This may be done by Principal Component Analysis or Sammon Mapping [9], [5]. The transformation should provide the lower-dimension picture of the search space where the dissimilarities between

data points of multidimensional domain corresponds with the dissimilarities of the lower-dimensional domain. After transformation the solutions may be visualized in two or three dimensions.

All these methods explore the properties of evolutionary algorithms at hand in order to provide same clues for designer of the algorithm. What the parameters should be (crossover probability, mutation probability etc.)? Is the chromosome structure good for the problem considered? How the search space is explored? They may offer the user the opportunity to interactively explore the evolutionary computing and the properties of the problem considered, this may lead to better understanding of Evolutionary Computing and the additional benefits of "Human-EA Interaction" [5] may be achieved. Moreover some educational tasks may be easily archived.

In the paper, we propose a new method of analysis of evolutionary algorithm based on a graph theory. We claim that such an approach may give us additional information and important knowledge about the evolutionary processes in general. The visualization by means of graph theory do not rely on the dimension of search space. Instead only relation between individuals are considered. Thus it gives the insight into the algorithm progress from other perspective than currently used methods.

Let us notice that exhaustive search usually organizes the search space as a tree, and then the following exhaustive search technique is applied: depth-first search method. This is not the case in evolutionary computation. The node may occur again by crossover or mutation in subsequent populations. Therefore, we may analyze and visualize the structure of the graph describing the search process of the algorithm.

The paper is organized as follows. Section 2 contains the description of evolutionary algorithm used in the paper and evolutionary graph construction process. Next section contains description of similar graph for local search optimization technique. Section 4 describes the optimization problem we consider as example (Travelling Salesman Problem). Section describes the results of visualization of evolutionary and local search. The next two sections study properties of obtained graphs. Finally, the paper is summarized and possible future research directions are given.

2 Evolutionary algorithms and graph construction

Fig. 1 presents the evolutionary algorithm used in our studies [8]. During the process the graph $G(V, E)$ is built. The nodes of this graph correspond to chromosomes of all populations of the algorithm. The identical chromosomes (in terms of their structure) are represented by the same single node. So if the same solution occurs again during the search process, it is represented by the same graph node.

The edges are between parents and their children. We join by an edge a chromosome and its offspring (either obtained by mutation or crossover). In

our implementation of the algorithm the crossover or mutation is performed in a single iteration, however the same technique of graph construction may be used in any other possible combination. We employ elite selection i.e. we select, the best chromosomes from previous population and its offspring.

```
initialization λ members of population P_0
initialize empty graph G(V, E)
for each chromosome in P_O add node to V
t = 0
while (not stop condition)
    O_{t+1} = ∅
    divide P_t into λ/2 two-elements subsets
            K_m  (m = 1, .., λ/2)
    for each (x_i, x_j) ∈ K_m(m = 1, .., λ/2)
        if u_(0.1) < p_cross than
            (x̂_i, x̂_j) = crossover(x_i, x_j)
            add edge (x_i, x̂_i) to E
            add edge (x_j, x̂_i) to E
            add edge (x_i, x̂_j) to E
            add edge (x_j, x̂_j) to E
            O_{t+1} = O_{t+1} ∪ {x̂_i, x̂_j}
        else
            x̂_i = mutate(x_i)
            x̂_j = mutate(x_j)
            add edge (x_i, x̂_i) to E
            add edge (x_j, x̂_i) to E
            O_{t+1} = O_{t+1} ∪ {x̂_j, x̂_i}
        end
    end
    evaluate chromosomes from O_{t+1}
    P_{t+1} = the best λ chromosomes from P_t ∪ O_{t+1}
    t = t + 1
end
```

Fig. 1. The evolutionary algorithm used in simulations. $u_{(0,1)}$ gives random number with uniform distribution in $(0, 1)$.

The in-degree of a node is the number of times it has been generated as an offspring of a mutation or crossover, and the out-degree of a node is the number of times the node has been used as a parent for other chromosomes.

3 Local search

One of the basic approaches to combinatorial optimization is local search. Instead of searching the entire space of all possible solutions, we focus attention on a local neighborhood of some particular solution. For comparison, we

also build a graph of local search. We use the simplest possible local search method. At each iteration a new point from neighborhood of current solution is considered. If this new point gives better fitness function, the process is repeated with this new solution. Let us notice that resulting graph may not be acyclic. If the neighborhood is large the algorithm may resample points that were already tried.

```
start with empty graph G(V, E)
x = random solution
while (not stop )
     if exists y ∈ N(x) such that f(x) < f(y)
          add edge (x, y) to E
          x = y
     else
          stop
     end
end
```

Fig. 2. The local search algorithm used in simulations.

4 The TSP problem

All tests were performed on the Travelling Salesman Problem. We select this problem due to its conceptual simplicity and to the fact that it is well-known. The problem is NP-hard, and heuristic methods need to be used. We use standard permutation encoding [8]. To ensure that all chromosomes encode different TSP cycles, we use the following simple trick. Instead of using permutation of length n we use permutation of length $n-1$. The city 1 is never included in this permutation. Thus the permutation:

$$2-3-4-5-6$$

encodes the path

$$1-2-3-4-5-6$$

Similarly, the permutation:

$$3-4-5-6-2$$

encodes the path

$$1-3-4-5-6-2$$

This method allows us to encode all the possible TSP paths. However, the search space is reduced to $(n-1)!$ and only two different nodes of evolutionary graph may denote the same solution (this is resolved during graph construction process). We implement mutation as inversion i.e. we select two

points along the length of the permutation, which is then cut at these points, and the substring between these points is reversed. Standard PBX crossover is used [8].

There is a huge variety of local search algorithms for the TSP [8]. In our work we used the simplest possible method. The neighborhood of path x is defined by all permutations which may be obtained from x through the single inversion operation. This is equivalent to 2-opt optimization procedure.

Test problems are generated randomly. The cities of TSP problem are uniformly distributed in a unit square and the Euclidean distance metric is assumed [8].

5 Visualization of evolutionary graphs

We use Pajek software to visualize the obtained graphs [12], [7]. The gray level of the node in the picture denotes the time the chromosome was first generated, i.e. chromosomes generated in first population are white, whereas chromosomes generated in the last population are black. In local search graph we use black color for the nodes used for search, and white color for a node which dies immediately after born (Fig. 3).

The evolutionary search graph without crossover is close to the tree, whereas evolutionary graph with crossover is more dense. It could easily be seen that chromosomes generated in first iteration are not used in search in final permutations. This is strongly connected with time of life of a chromosome. In final population the best chromosomes have bigger out-degree than at the beginning of the search.

The local search has a complectly different structure (Fig. 3). It is very close to a tree, however, most of the nodes have out-degree equal to one.

6 Complex networks

Complex networks are currently being studied across many fields of science. Many systems in nature can be described by models of complex networks, which are structures consisting of nodes or vertices connected by links or edges. There are numerous examples of such systems: social networks, the internet, food webs, distribution networks, metabolic and protein networks, and citation networks [2], [1],[13],[10]. Most of these networks share the following three important features:

- The average shortest path length L is small. In order to connect the two edges of the graph, typically only a few edges need to be passed. Similarly, a graph diameter is small. A diameter of a graph is a maximum length of the shortest path between two nodes in the graph [1].

[1] diameter is defined as $\max_{u,v \in V} d(u,v)$, where $d(u,v)$ is graph distance between nodes u,v.

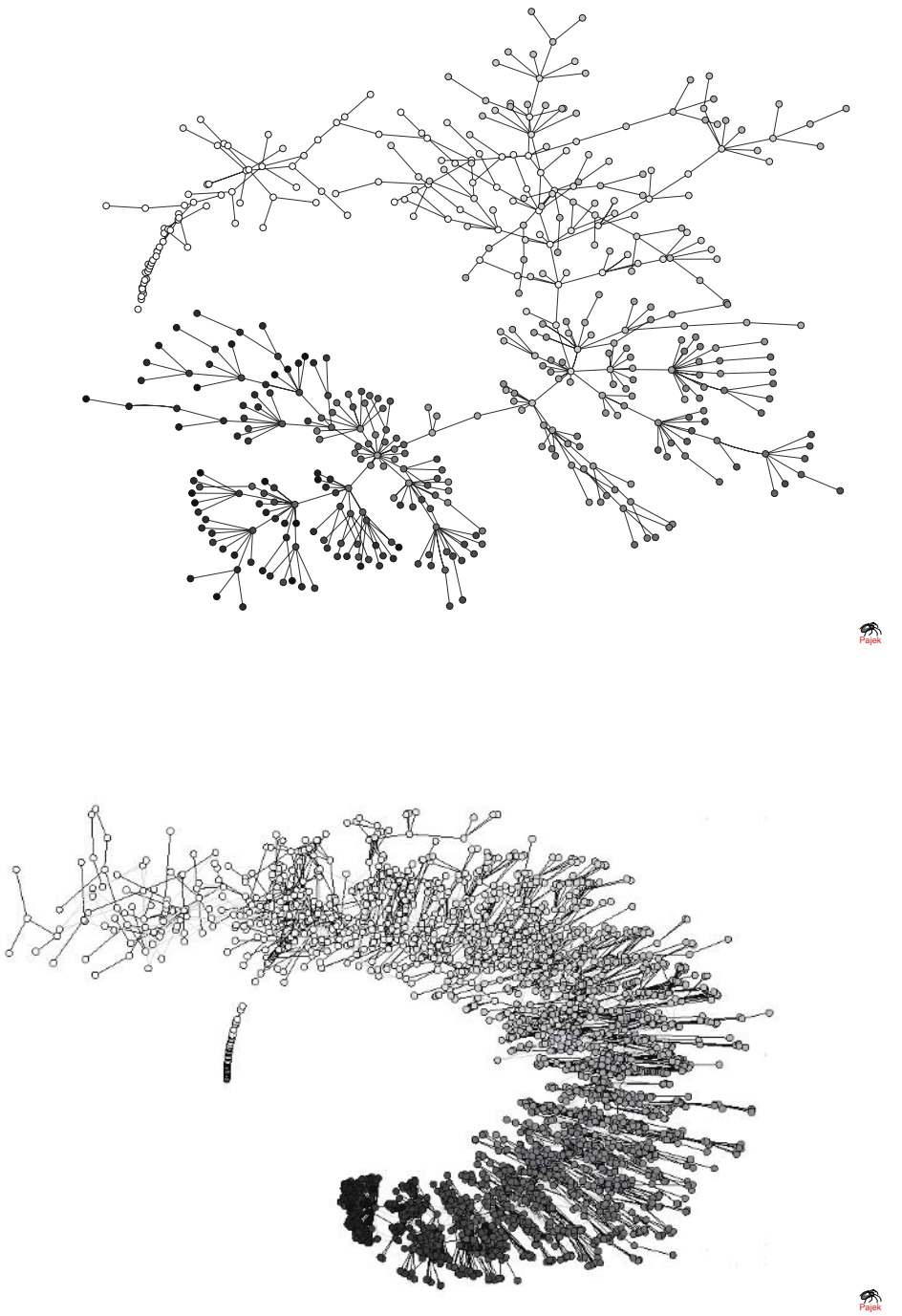


Fig. 3. Evolutionary graph with $p_{\text{cross}} = 0$ i.e only mutation is used and with $p_{\text{cross}} = 0.5$. Gray edges are obtained using crossover whereas black ones denote mutations. The nodes' gray level corresponds to the time (population) the chromosome was generated first time. $\lambda = 25$ TSP problem with 25 cities.

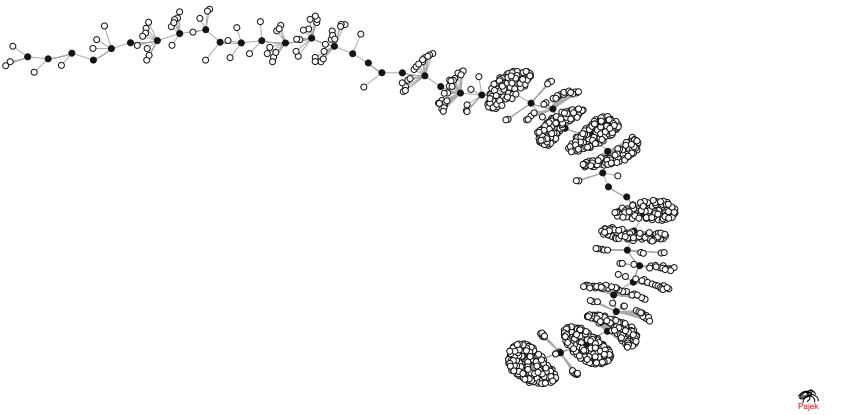


Fig. 4. Local search graph, TSP problem with 25 cities.

- The clustering coefficient C is large. The clustering coefficient C is an average fraction of pairs of neighbors of a node that are also neighbors for each other. Suppose the node i has k_i edges and they connect this node to k_i other edges. The clustering coefficient C_i of node is defined as the ratio between the number of edges E_i that actually exists between those k_i nodes and the total possible number:

$$C_i = E_i/(k_i(k_i - 1))$$

  If the clustering coefficient is large, two nodes having a common neighbor are far more likely to be connected to each other than are two nodes picked at random.
- The distributions of degree is scale-free i.e., it behaves as a power law of the form $P(k) \sim k^{-\gamma}$, where $P(k)$ is the probability that a randomly selected node has probability k . The degree exponent γ varies from 1.13 for food webs to 2.7 for language networks [13].

The third property is especially interesting as until recently the random models of complex network were assumed. In random networks the distribution of degree of nodes obeys the Poisson distribution. The discovery that many real networks can be described by a power law $P(k) \sim k^{-\gamma}$ raise the question about the model which can explain the behavior of many complex systems.

In this paper we analyze the properties of evolutionary graphs. We want to check which of these three properties occur in analysis of evolutionary computation.

7 Properties of evolutionary graph

We check the following evolutionary graph properties defined in the previous section:

- Diameter. The resulting graph is directed, so instead of checking the average path length, we decide to check the diameter of a graph. We found out that the diameter of the graph is approximately 5 times smaller than the number of populations. It is much larger than in any other networks present in nature.
- Clustering coefficient. The clustering coefficient of evolutionary graphs is close to 0. Although there are cycles in the graph, it is unlikely that two neighbors of a parent are connected.
- Nodes degrees. We found out the the distribution od nodes degrees in the evolutionary graph follows the power law $P(k) \sim k^{-\gamma}$. The only exception are nodes with out-degree equal to 1 and in-degree equal to 0. The number of such nodes exceeds the number predicted by power law. We have obtained $\gamma \sim 2$ for out-degree and $\gamma \sim 3$ for in-degree. The results for $n = 100$, size of population of 100 chromosomes are presented in Fig. 7. The degrees of local search graph do not follow the power law.

Let us emphasis the third property. The better nodes i.e. the nodes with better fitness function are most often used as a parents and generate offsprings. This is not the case in local search which do not have this property.

8 Summary

In this paper we present possibilities of graph-based analysis of evolutionary computation. We present the possibility of visualizing the evolutionary process using graph drawing software. We compare and visualize the search graph of evolutionary algorithm with and without crossover and the process of local search. We found out, the nodes' degrees follow the power law which is present in many networks present in nature i.e. WWW, protein interaction, etc. However, clustering coefficient of evolutionary graph is very small.

In the future other questions may be addressed:

- Is it possible to use the results of such simulation to improve the performance of evolutionary computation?
- Are there any connections between evolutionary graphs and other graphs present in nature?

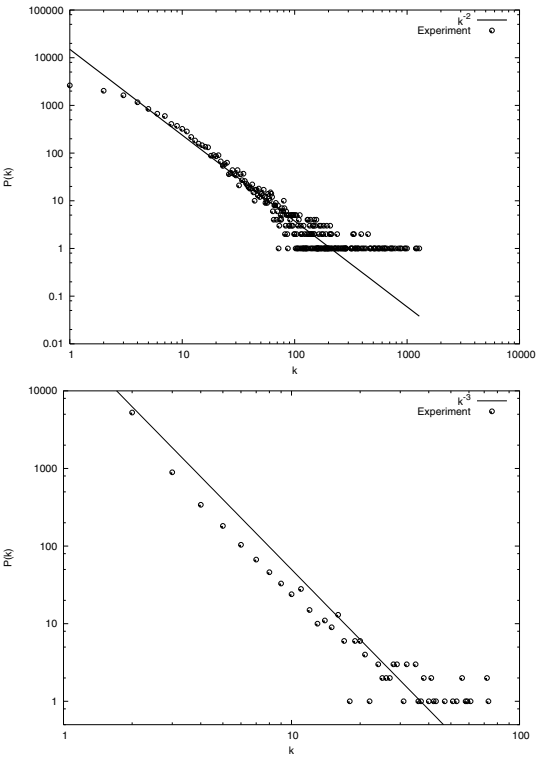


Fig. 5. Out and in degree histograms for evolutionary graph. TSP with 100 cities. $\lambda = 100$, $p_{\text{cross}} = 0$

- What parameters of evolutionary computation determine the structure of evolutionary graph?
- Are there any parameters of evolutionary graphs which describe well-designed algorithm?

More sophisticated data mining techniques may also be used to analyze the graph of evolutionary search process and other visualization techniques may be combined with the proposed one. We hope, that the analysis of this graph may lead to better understanding of the evolutionary process in general.

References

1. Reka Albert and Albert-Laszlo Barabasi. Statistical mechanics of complex networks. *Review of Modern Physics*, pages 47–97, January 2002.
2. Albert-Laszlo Barabasi and Zoltan N. Oltvai. Network biology: Understanding the cell's functional organization. *Nature*, pages 101–113, February 2004.

3. A. Chorazyczewski and R. Galar. Visualization of evolutionary adaptation in R^n. In *Evolutionary Programming*, pages 659–668, 1998.
4. T. D. Collins. *Advances in evolutionary computing: theory and applications*, chapter Visualizing evolutionary computation, pages 95–116. Springer-Verlag New York, Inc., 2003.
5. T.D. Collins. *The Application of Software Visualization Technology to Evolutionary Computation: A case study in genetic algorithms.* PhD thesis, The Open University, Knowledge Media Institute, Milton Keynes, UK, 1998.
6. T.D. Collins. Understanding evolutionary computing: A hands on approach. In *In The Proceedings of the International Conference on Evolutionary Computation (ICEC'98)*, 1998.
7. Guy Melancon Ivan Herman and M. Scott Marshall. Graph visualization and navigation in information visualization: a survey. *IEEE Trans. on Visualization and Computer Graphics*, pages 47–97, January 2000.
8. Zbigniew Michalewicz and David Fogel. *How to solve it: Modern Heuristcs.* Springer, 1999.
9. H. Polheim. Visualization of evolutionary algorithms - set of standard techniques and multidimensional visualization. *Proceedings of The Genetic and Evolutionary Computation Conference*, pages 533–540, 1999.
10. Albert-Laszlo Barabasi Reka Albert, Hawoong Jeong. Diameter of the world-wide web. *Nature*, page 130, September 1999.
11. T.W. Routen. Techniques for the visualization of genetic algorithms. *Proceedings of The First IEEE Conference on Evolutionary Computation*, pages 846–851, 1994.
12. A. Mrvar V. Batagelj. Pajek - analysis and visualization of large networks. *Jnger, M., Mutzel, P., (Eds.) Graph Drawing Software, Springer Berlin*, pages 77–103, 2003.
13. Xiao Fan Wang and Guanrong Chen. Complex networks: Small-word, scale-free and beyond. *IEEE Circuicts and Systems Magazine*, pages 6–20, First quarter 2003.

Part V

Regular Sessions: Statistical and Database Methods in AI

Probability of Misclassification in Bayesian Hierarchical Classifier

Robert Burduk

Chair of Systems and Computer Networks, Wrocław University of Technology, Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland

Abstract. The paper deals with the probability of misclassification in a multistage classifier. This classification problem is based on a decision-tree scheme. For given tree skeleton and features to be used, the Bayes decision rules at each non-terminal node are presented. Additionally the information on objects features is fuzzy or non-fuzzy. The upper bound of the difference between probability of misclassification for the both information's is presented. In the paper we use the maximum likelihood estimator for fuzzy data.

1 Introduction

Knowledge discovery is a nontrivial process of identifying potentially useful, novel, valid and ultimately understandable patterns from large collections of data [6]. Data mining is one of the knowledge discovery steps. Data mining is the step that is concerned with the actual extraction of knowledge from data, in contrast to the knowledge discovery process that is concerned with many other thinks like preparation and understanding of the data, verification and application of the discovery knowledge. Clustering, classification, regression, time series are, among others the tasks of data mining. This paper describes hierarchical classifier problem which is characterized by the fact, that an unknown pattern is classified into a class using several decision in a successive manner. The mechanics of multistage classification can be described by means of a tree, in which the terminal nodes represent labels of classes and the interior nodes denote the groups of classes.

Classical statistical techniques assume that both the data and the probability functions are represented by know numerical values thus knowledge of data is precise. In practice, we are often faced with two types of uncertainty – randomness and fuzziness. Randomness is described and investigated using probability theory methods which satisfy statistical laws. In this type of uncertainty subjective influences are not taken into account. Fuzziness is characterized by nonstatistical properties and subjective influences. It is based on the fuzzy set theory. There are many cases where the available information is a mixture of randomness and fuzziness. A simple example of such situation is classification where the observations of the features are fuzzy-valued, but the probabilities of classes are numerical. In order to manage to treat both types of uncertainties, it is necessary to incorporate fuzzy concept into statistical

technique. This problem has been studied by many authors. Following papers describe statistical point estimation in the fuzzy environment [5,7,17], fuzzy decision trees [4,10], testing of fuzzy hypotheses [3,8], fuzzy Bayesian statistics [1,15] and other combinations of statistical methods and fuzzy set theory. In this paper, we consider the problem of probability of misclassification in multistage classifier when observations of the features are fuzzy-valued.

2 Bayesian hierarchical classifier

In the paper [12] the Bayesian hierarchical classifier is presented. The synthesis of multistage classifier is a complex problem. It involves specification of the following components:

- the decision logic, i.e. hierarchical ordering of classes,
- feature used at each stage of decision,
- the decision rules (strategy) for performing the classification.

The present paper is devoted only to the last problem. This means that we shall deal only with the presentation of decision algorithms, assuming that both the tree skeleton and feature used at each non-terminal node are specified.

The procedure in the Bayesian hierarchical classifier consist of the following sequences of activities. At the first stage, there are measured some specific features x_0. They are chosen from among all accessible features x, which describe the pattern that will be classified. These data constitute a basis for making a decision i_1. This decision, being the result of recognition at the first stage, defines a certain subset in the set of all classes and simultaneously indicates features x_{i_1} (from among x) which should be measured in order to make a decision at the next stage. Now at the second stage, features x_{i_1} are measured, which together with i_1 are a basis for making the next decision x_2. This decision – like i_1 – indicates features x_{i_2} necessary to make the next decision (at the third stage) and – again as at the previous stage – defines a certain subset of classes, not in the set of all classes, however, but in the subset indicated by the decision i_1, and so one. The whole procedure ends at the last N-th stage, where the decision made i_N indicates a single class, which is the final result of multistage recognition. Thus multistage recognition means a successive narrowing of the set of potential classes from stage to stage, down to a single class, simultaneously indicating at every stage features which should be measured to make the next decision in more precise manner.

3 Decision problem statement

Let us consider a pattern recognition problem, in which the number of classes is equal to M. Let us assume that classes were organized in a $(N+1)$ horizontal decision tree. Let us number all nodes of the constructed decision-tree

with consecutive numbers of $0, 1, 2, \ldots$, reserving 0 for the root-node and let us assign numbers of classes from the $\mathcal{M} = \{1, 2, \ldots, M\}$ set to terminal nodes so that each one of them is labelled with the number of the class which is connected with that node. This allows the introduction of the following notation:

- $\mathcal{M}(n)$ – the set of numbers of nodes, which distance from the root is n, $n = 0, 1, 2, \ldots, N$. In particular $\mathcal{M}(0) = \{0\}$, $\mathcal{M}(N) = \mathcal{M}$,
- $\overline{\mathcal{M}} = \bigcup_{n=0}^{N-1} \mathcal{M}(n)$ – the set of interior node numbers (non terminal),
- $\mathcal{M}_i \subseteq \mathcal{M}(N)$ – the set of class labels attainable from the i-th node ($i \in \overline{\mathcal{M}}$),
- $\mathcal{M}^i$ – the set of numbers of immediate descendant nodes ($i \in \overline{\mathcal{M}}$),
- m_i – number of direct predecessor of the i-th node ($i \neq 0$).

We will continue to adopt the probabilistic model of the recognition problem, i.e. we will assume that the class label of the pattern being recognized $j_N \in \mathcal{M}(N)$ and its observed features x are realizations of a couple of random variables $\boldsymbol{J}_N$ and $\boldsymbol{X}$. Complete probabilistic information denotes the knowledge of a priori probabilities of classes:

$$p(j_N) = P(J_N = j_N), \quad j_N \in \mathcal{M}(N) \tag{1}$$

and class-conditional probability density functions:

$$f_{j_N}(x) = f(x/j_N), \quad x \in X, \quad j_N \in \mathcal{M}(N) \ . \tag{2}$$

Let

$$x_i \in X_i \subseteq R^{d_i}, \quad d_i \leq d, \quad i \in \mathcal{M} \tag{3}$$

denote vector of features used at the i-th node, which have been selected from the vector x.
Our target now is to calculate the so-called multistage recognition strategy $\pi_N = \{\Psi_i\}_{i \in \overline{\mathcal{M}}}$, that is the set of recognition algorithms in the form:

$$\Psi_i : X_i \to \mathcal{M}^i, \quad i \in \overline{\mathcal{M}} \ . \tag{4}$$

Formula (4) is a decision rule (recognition algorithm) used at the i-th node, which maps observation subspace to the set of immediate descendant nodes of the i-th node. Equivalently, decision rule (4) partitions observation subspace X_i into disjoint decision regions $D^k_{x_i}$, $k \in \mathcal{M}^i$, such that observation x_i is allocated to the node k if $k_i \in D^k_{x_i}$, namely:

$$D^k_{x_i} = \{x_i \in X_i : \Psi_i(x_i) = k\}, \ k \in \mathcal{M}^i, \quad i \in \overline{\mathcal{M}}. \tag{5}$$

Our aim is to minimizes the mean risk function (the probability of misclassification) denoted by:

$$R^*(\pi^*_N) = \min_{\Psi_{i_n}, \ldots, \Psi_{i_{N-1}}} R(\pi_N) = \min_{\Psi_{i_n}, \ldots, \Psi_{i_{N-1}}} E[L(I_N, J_N)]. \tag{6}$$

The π_N^* strategy we will call the globally optimal N-stage recognition strategy. In the next chapter we will calculate globally optimal strategy for the zero-one loss function.

4 The recognition algorithm

A fuzzy information $\tilde{A}_i$ from X_i is a fuzzy event characterized by membership function $\mu_{\tilde{A}_i}(x_i)$. Fuzzy observation is described by fuzzy number or fuzzy trapezoidal interval [13] represented by the following simple notation: fuzzy triangular number – $\tilde{A} = (a_1,\ a_2,\ a_3,\ a_4)$, fuzzy trapezoidal interval – $\tilde{A} = (a_1,\ a_2,\ a_3,\ a_4)_{Tr}$. In fuzzy triangular number $a_2 = a_3$. In addition, assume that for each component k of observation subspace X_i the set of all available fuzzy numbers $\tilde{\mathcal{A}}_i = \{\tilde{A}_i^1,\ \tilde{A}_i^2, \ldots, \tilde{A}_i^k\}$ satisfies the orthogonality constraint [14]. Probability of fuzzy event assume in Zadeh's form [18] $P(\tilde{A}) = \int\limits_{supp(\tilde{A})} \mu_{\tilde{A}}(x) f(x) dx$, where $supp(\tilde{A})$ denotes carrier of the $\tilde{A}$ set.

Applying procedure similar to [2] and use zero-one loss function we obtain searched globally optimal strategy with decision algorithms as follows:

$$\Psi_{i_n}^*(\tilde{A}_{i_n}) = i_{n+1} \quad \text{when} \tag{7}$$

$$\sum_{j_N \in \mathcal{M}_{i_{n+1}}} p(j_N) q_F^*(j_N/i_{n+1}, j_N) \int\limits_{supp\, \tilde{A}_{i_n}} \mu_{\tilde{A}_{i_n}}(x_{i_n}) f_{j_N}(x_{i_n}) dx_{i_n} =$$

$$= \max_{k \in \mathcal{M}^{i_n}} \sum_{j_N \in \mathcal{M}_k} p(j_N) q_F^*(j_N/k, j_N) \int\limits_{supp\, \tilde{A}_{i_n}} \mu_{\tilde{A}_{i_n}}(x_{i_n}) f_{j_N}(x_{i_n}) dx_{i_n}$$

for $i_n \in \mathcal{M}(n),\ n = 0, 1, 2, \ldots, N-1$, where $q_F^*(j_N/i_{n+1}, j_N)$ denotes the probability of accurate classification of the object of the class j_N in further stages using π_N^* strategy rules on condition that on the n-th stage the i_{n+1} decision has been made.

Using maximum likelihood estimation method [5] we obtain the following decision rules for hierarchical classifier with fuzzy observations of the features:

$$\Psi_{i_n, S_m}^{(MNW)}(\tilde{A}_{i_n}) = i_{n+1} \quad \text{when} \tag{8}$$

$$\sum_{j_N \in \mathcal{M}_{i_{n+1}}} \frac{\nu_{j_N}}{\sqrt{s_{i_n, j_N}^2 + \lambda_{i_n, j_N}^2}} \hat{q}_F(j_N/i_{n+1}, j_N) \times$$

$$\times \int\limits_{supp\, \tilde{A}_{i_n}} \mu_{\tilde{A}_{i_n}}(x_{i_n}) \exp(-(x_{i_n} - \bar{a}_{i_n, j_N})^2 / 2(s_{i_n, j_N}^2 + \lambda_{i_n, j_N}^2)) dx_{i_n} =$$

$$= \max_{k \in \mathcal{M}^{i_n}} \sum_{j_N \in \mathcal{M}_k} \frac{\nu_{j_N}}{\sqrt{s_{i_n, j_N}^2 + \lambda_{i_n, j_N}^2}} \hat{q}_F(j_N/k, j_N) \times$$

$$\times \int\limits_{supp\,\tilde{A}_{i_n}} \mu_{\tilde{A}_{i_n}}(x_{i_n}) \exp(-(x_{i_n} - \bar{a}_{i_n,j_N})^2/2(s^2_{i_n,j_N} + \lambda^2_{i_n,j_N}))dx_{i_n},$$

where

$$\bar{a}_{i,j} = \frac{1}{\nu_j} \sum_{k=1}^{\nu_j} \frac{(a^k_{3,i,j}+a^k_{4,i,j})^2-(a^k_{1,i,j}+a^k_{2,i,j})^2-a^k_{3,i,j}a^k_{4,i,j}+a^k_{1,i,j}a^k_{2,i,j}}{3(a^k_{4,i,j}-a^k_{1,i,j}-a^k_{2,i,j}+a^k_{3,i,j})},$$

$$s^2_{i,j} = \frac{1}{\nu_j} \sum_{k=1}^{\nu_j} \left(\frac{(a^k_{3,i,j}+a^k_{4,i,j})^2-(a^k_{1,i,j}+a^k_{2,i,j})^2-a^k_{3,i,j}a^k_{4,i,j}+a^k_{1,i,j}a^k_{2,i,j}}{3(a^k_{4,i,j}-a^k_{1,i,j}-a^k_{2,i,j}+a^k_{3,i,j})} - \bar{a}_{i,j} \right)^2,$$

$$\lambda^2_{i,j} = \frac{1}{\nu_j} \sum_{k=1}^{\nu_j} \left(\frac{1}{18} \left(\left(a^k_{4,i,j} - a^k_{3,i,j}\right)^2 + \left(a^k_{2,i,j} - a_{1,i,j^k}\right)^2 + \left(a^k_{2,i,j} - a^k_{1,i,j}\right) \times \right.\right.$$
$$\left. \times \left(a^k_{4,i,j} - a^k_{3,i,j}\right) \right) + \frac{1}{12} \left(\left(a^k_{3,i,j} - a^k_{2,i,j}\right)^2 + \left(a^k_{3,i,j} - a^k_{2,i,j}\right) \times \right.$$
$$\left.\left. \times \left(\left(a^k_{4,i,j} - a^k_{3,i,j}\right) + \left(a^k_{2,i,j} - a^k_{1,i,j}\right) \right) \right) \right).$$

The $\hat{q}_F(j_N/i_{n+1}, j_N)$ denotes the empirical joint probability of correct classification at the next stage. It can be calculated, the so-called "leave-one-out" method.

When we use fuzzy information on object features instead of exact information we deteriorate the classification accuracy. The upper bound of the difference between probability of misclassification for the both information's is the following:

$$Pe_F(\pi^*_N) - Pe(\pi^*_N) \leq \sum_{j_N \in \mathcal{M}(N)} p(j_N) \sum_{i_k \in s(j_N)-\{0\}} \varepsilon_{m_{i_k}} \tag{9}$$

where

$$\varepsilon_i = \sum_{\tilde{A}_i \in X_i} \left| \int\limits_{supp\,\tilde{A}_i} \mu_{\tilde{A}_i}(x_i) \max_{k \in \mathcal{M}^i} \{f_k(x_i)\} dx_i - \max_{k \in \mathcal{M}^i} \left\{ \int\limits_{supp\,\tilde{A}_i} \mu_{\tilde{A}_i}(x_i) f_k(x_i) dx_i \right\} \right|.$$

Let us illustrate the obtained results with the following example.

5 Illustrative example

Let us consider the two-stage binary classifier. Four classes have identical a priori probabilities which are equal 0.25. The data for the experiments were computer generated 3-dimensional random variables $x = [x^{(1)},\ x^{(2)},\ x^{(3)}]$. For performing the classification at the root-node 0 the first coordinate was used, components $x^{(2)}$ and $x^{(3)}$ were used at the node 5 and 6 respectively. The data were Gaussian random variables with covariance matrices equal for every class $\sum_{j_2} = 4I,\ j_2 \in \mathcal{M}(2)$, and with the following expected values $\mu_1 = [0,\ 0,\ 0]$, $\mu_2 = [0,\ 4,\ 0]$, $\mu_3 = [3,\ 0,\ 1]$, $\mu_4 = [3,\ 0,\ 8]$. In experiments were used the following sets of fuzzy numbers:

case A

$\tilde{\mathcal{A}} = \{\tilde{A}_1 = (\infty, -9, -8),\ \tilde{A}_2 = (-9, -8, -8, -7), \ldots,$
$\tilde{A}_{25} = (14, 15, 15, 16),\ \tilde{A}_{26} = (15, 16, \infty)\}.$

case B
$\tilde{\mathcal{B}} = \{\tilde{B}_1 = (\infty, -9, -8.5)\,,\ \tilde{B}_2 = (-9, -8.5, -8.5, -8), \ldots,$
$\tilde{B}_{50} = (15, 15.5, 15.5, 16)\,,\ \tilde{B}_{51} = (15.5, 16, \infty)\}.$
case C
$\tilde{\mathcal{C}} = \{\tilde{C}_1 = (\infty, -9.25, -8.75)\,,\ \tilde{C}_2 = (-9.25, -8.75, -8.25, -7.75)_{Tr}, \ldots,$
$\tilde{C}_{26} = (14.75, 15.25, 15.75, 16.25)_{Tr}\,,\ \tilde{C}_{27} = (15.75, 16.25, \infty)\}.$
and in case D we use non-fuzzy data.

The sizes of the training sets m: 10, 20, 50, 100, 250, 500, 750, 1000, the size of the testing set: 500.

Fig. 1 summarizes the results of simulations and shows influence of fuzzy numbers who describe features space on classification accuracy.

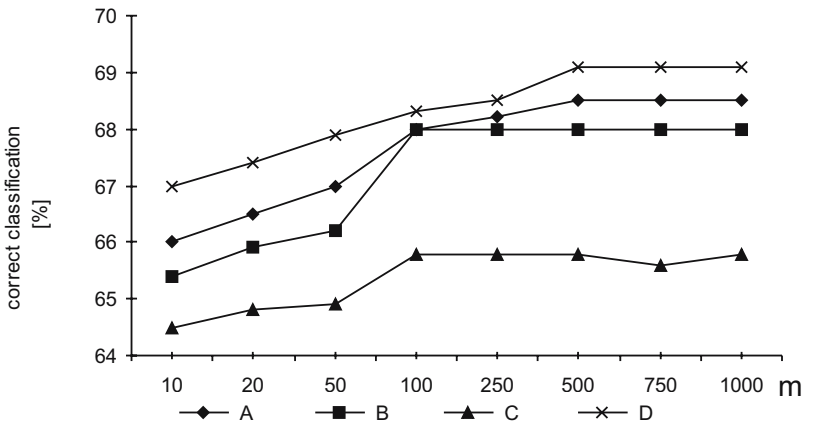


Fig. 1. Results of simulation

Below we present the upper bound of the difference between probability of misclassification for fuzzy and non fuzzy data calculated form (9).

Table 1. Upper bound of misclassification

Case	$Pe_F(\pi_N^*) - Pe(\pi_N^*)$
A	0,9
B	1,6
C	2,0

6 Conclusion

In the present paper we have concentrated on the optimal (Bayes) strategy for the hierarchical classifier based on a decision tree scheme. Assuming that both the tree skeleton and the features, used at each interior node, are specified, we derive the decision rules of the multistage Bayes classifier with fuzzy observations of the features. We use maximum likelihood method for estimation conditional density function The obtained algorithm is a generalization of the result presented in [12]. Additionally the upper bound of the difference between probability of misclassification for fuzzy and non fuzzy information about features is presented.

Obtained results shows that choice of fuzzy numbers sets who describe features space is very important. Above the sizes of the training sets equal 100 we don't have higher classification accuracy for fuzzy data. Always algorithm for non-fuzzy data have higher classification accuracy.

Further research should concern combining techniques [9,16] to improve of performance of presented classifier.

References

1. Berger J. O. (1985) Statistical Decision Theory and Bayesian Analysis. Springer-Verlag, Berlin Heidelberg
2. Burduk R. (2004) Decision Rules for Bayesian Hierarchical Classifier with Fuzzy Factor. Soft Methodology and Random Information Systems, Advances in Soft Computing. 519–526
3. Casals M. R., Gil M. A., Gil P. (1986) On the Use of Zadeh's Probabilistic Definition for Testing Statistical Hypotheses from Fuzzy Information. Fuzzy Sets and Systems **20**, 175–190
4. Chang R. L., Pavlidis T. (1977) Fuzzy Decision Tree Algorithms. IEEE Trans. Systems, Man, and Cybernetics **1**, 28–35
5. Corral N., Gil M. A. (1984) The Minimum Inaccuracy Fuzzy Estimation : an Extension of the Maximum Likelihood Principle. Stochastica **8**, 63-81
6. Fayyad U. M., Piatetetsky-Shapiro G., Smyth P., Uthurusamy R.(1996) Advances in Knowledge Discovery and Data Mining. AAAI Press, Cambridge
7. Gertner G. Z., Zhu H. (1996) Bayesian Estimation in Forest Surveys when Samples or Priori Information Are Fuzzy. Fuzzy Sets and Systems **77**, 277–290
8. Grzegorzewski P. (2000) Testing Statistical Hypotheses with Vague Data. Fuzzy Sets and Systems **112**, 501–510
9. Hung W. L. (2001) Bootstrap Method for Some Estimators Based on Fuzzy Data. Fuzzy Sets and Systems **119**, 337–341
10. Janickow C. Z. (1998) Fuzzy Decision Trees: Issues and Methods. IEEE Trans. Systems, Man, and Cybernetics B: Cybernetics **28**, 1–14
11. Kurzyńsk M. (1983) Decision Rules for a Hierarchical Classifier. Pattern Recognition Letters **1**, 305–310
12. Kurzyńsk M. (1988) On the Multistage Bayes Classifier. Pattern Recognition **21**, 355–365

13. Möller B., Beer M. (2004) Fuzzy Randomness. Springer-Verlag, Berlin Heidelberg
14. Pedrycz W. (1990) Fuzzy Sets in Pattern Recognition: Methodology and Methods. Pattern Recognition **23**, 121–146
15. Viertl R. (1996) Statistical Methods for Non-Precise Data. CRC Press, Boca Raton
16. Woźniak M. (2004) Proposition of Boosting Algorithm for Probabilistic Decision Support System. LNCS **3036**, 675–678
17. Yao J. S., Hwang C. M. (1996) Point Estimation for the n Sizes of Random Sample with One Vague Data. Fuzzy Sets and Systems **80**, 205–215
18. Zadeh L. A. (1968) Probability Measures of Fuzzy Events. Journal of Mathematical Analysis and Applications **23**, 421–427

A study on Monte Carlo Gene Screening

Michał Dramiński[1], Jacek Koronacki[1], and Jan Komorowski[2]

[1] Institute of Computer Science, Polish Academy of Sciences,
Ordona 21, 01-237 Warsaw, Poland
[2] The Linnaeus Centre for Bioinformatics, Uppsala University,
SE-751 24 Uppsala, Sweden

Abstract. In the paper, three conceptually simple but computer-intensive versions of an approach to selecting informative genes for classification are proposed. All of them rely on multiple construction of a tree classifier for many training sets randomly chosen from the original sample set, where samples in each training set consist of only a fraction of all of the genes. It is argued that the resulting ranking of genes can then be used to advantage for classification via a classifier of any type.

1 Introduction and Problem Statement

It has been evident from the outset that the advent of microarray technology would open vast new areas of research with a potentially revolutionary impact on the society. At the same time, it has opened new, and fascinating, challenges for researchers, including data analysts of whatever methodological background. Regarding analysis of microarray gene expression data, much outstanding work has already been done, to mention only a fundamental book-length treatment of the field [5] and the references there, as well as, e.g., numerous papers in *J. Computational Biol.*, *Biostatistics*, *Bioinormatics*, *Science* and presentations at many prestigious conferences.

In particular, very much has been done concerning class prediction or supervised classification (see [5], the chapter by Sandrine Dudoit and Jane Fridlyand). The major challenge here comes from typical sizes of data matrices: very small number of records (samples), of the order of tens, versus thousands of attributes or features for each record (in our context the features are genes or, more precisely, their expressions).

As Dudoit and Fridlyand argue convincingly and in detail in [5], cf. also Simon et al. [4], *The importance of taking feature selection into account when assessing the performance of a classifier cannot be stressed enough.* Even more than that, one definitely would like to have genes which are differentially expressed, and hence informative or "relatively important", for a given classification problem **regardless** of a classifier used (or to be used). To put it otherwise, one would like to have a kind of objective measure of relative importance of genes for the particular classification problem.

This study is a continuation of our earlier work (see [3]) whose aim was to provide such a measure. In this paper, an important modification of that measure is proposed, which again rests on constructing millions of trees for

randomly selected subsets of genes. Our new measure, to be termed modified relative importance or mRI, rewards genes that have been used to build any given tree proportionally to the tree's classification result on a testing set (in [3], the classification results on testing sets were not taken into account when calculating relative importance of genes).

As previously, our proposal is scrutinized in the context of one exemplary data set, namely the lymphoma data of Alizadeh et al. [1], available from the authors' web site. The data consist of measurements on 4,026 genes from 62 patients. The patients are classified into three classes: lymphoma and leukemia (DLCL; 42 samples), follicular lymphoma (FL; 9 samples) and chronic lymphocytic leukemia (CLL; 11 samples).

In the next section, we present the screening procedure in detail and propose three versions of constructing the trees, thus providing three ways of obtaining mRI's for the genes. In effect, three final sets of most informative genes are found.

In Sec. 3, we compare classification results (sets of genes) from the three versions using just 45 genes earlier found to be relatively most important for each of the versions. This is done by constructing again (for each version) thousands of trees on the 45 most important genes, where for each set of genes many training subsets of samples are drawn at random from the 62 original samples and trees are grown (and pruned) on these subsets.

Classification results are measured, roughly speaking, by accuracy obtained on a test set (for each tree constructed, the original set of samples is randomly split into the training and test sets).

In Sec. 4 we conclude with a few remarks.

2 Gene screening

As already stated in the previous section, for each of the versions of our approach, measuring relative importance of genes consists in constructing millions of trees for randomly selected subsets of genes. For each subset of genes we run training and testing a couple of times (this we call the inner loop) for randomly chosen sets of examples.

In all the experiments C4.5 trees (version 8) were constructed, as implemented in WEKA 3-4-1 [8] (j48 tree, with the same parameters throughout the whole study, in particular with ConfidenceFactor for pruning equal to 0.25). Trees were grown on the original data of Alizadeh et al., i.e., with missing data left intact (not imputed).

Modified relative importance (mRI) of a gene was defined as a sum of (partial) relative importances for separate trees. For a given tree, the partial relative importance of a gene was determined as the product of the squared weighted accuracy for that tree, $wAcc^2$, and the number of splits made on that gene in all the nodes of the tree. As was mentioned already, in [3] the relative importance was determined as equal to the overall number of splits made

on the gene in all nodes of all trees. Thus, each tree was treated in the very same way, no matter if a given tree proved good or not on a testing set. Using mRI we add penalty for inaccuracy of classification. Squaring wAcc proved important as it better differentiates between trees of different classification capability.

Before we give results for the modified relative importance of genes, let us summarize classification results from all the trees (see [3] for some more details).

Classification results from all the trees were measured by accuracy and a more objective performance index, weighted accuracy, which takes into account sizes of the classes in such a way as to prevent undue influence of a majority class on the performance index. For the confusion matrix,

$$\begin{array}{c} \text{Predicted} \\ \begin{array}{|ccc|} \hline T_1 & F_{12} & F_{13} \\ F_{21} & T_2 & F_{23} \\ F_{31} & F_{32} & T_3 \\ \hline \end{array} \text{Observed} \end{array}$$

where, say, $T_1 \equiv T_{DLCL}$ is the number of DLCL samples correctly classified, $T_2 \equiv T_{FL}$, $T_3 \equiv T_{CLL}$, F_{12} is the number of DLCL samples classified as FL, etc., accuracy is defined as

$$Acc = (T_{DLCL} + T_{FL} + T_{CLL})/N,$$

with N denoting the overall number of samples, and weighted accuracy is defined as

$$wAcc = \frac{1}{3}\left(\frac{T_1}{T_1 + F_{12} + F_{13}} + \frac{T_2}{T_2 + F_{21} + F_{23}} + \frac{T_3}{T_3 + F_{31} + F_{32}}\right),$$

i.e., as the mean of the three true positive rates. Both Acc and $wAcc$ are, of course, measured for each tree on a test set. Their values for all the trees combined (i.e., the values corresponding to the sum of all confusion matrices) are 0.846 and 0.761, respectively (combined true positive rates are: 0.93 for DLCL, 0.61 for FL and 0.74 for CLL).

Modified relative importance (mRI) of a gene was defined as a sum of relative importances for separate trees. For a given tree, determined by sum of $wAcc^2$ trees that contained given gene. If gene occurred in two nodes of the tree we added $2 * wAcc^2$ to mRI. In classic RI gene that have been chosen as a node in a tree many times had high RI no matter if tree was good or not on testing set. Using mRI we add penalty if classification is not so good. Squared wAcc is important because influences on differential of mRI.

2.1 Three-class classification

The first version of calculating mRI is based on the usual approach to classification, i.e., on considering our exemplary classification task as a usual

3-class problem. Out of 4026 genes 60,000 subsets of 100 genes were selected at random. For each subset obtained, 30 trees were constructed and their performance was assessed. Each of the 30 trees in the inner loop was trained and evaluated on a different, randomly selected training and test samples. Each time 40 samples were drawn at random for training (in such a way as to preserve proportions of classes from the full set of data) and the remaining 22 samples were used for testing. A relatively small subset size of 100 of genes out of all 4026 genes was used to prevent informative genes to be masked too severely by those relatively most important ones (we shall comment more on the issue of masking in the sequel; cf., e.g., Breiman et al. [2] for a general discussion of the masking effect). The final set of the best genes, given below, includes 45 genes with the highest mRI.

GENE1553X, GENE1602X, GENE1605X, GENE1606X, GENE1610X
GENE1611X, GENE1613X, GENE1622X, GENE1647X, GENE1661X
GENE1672X, GENE1673X, GENE2097X, GENE2368X, GENE2373X
GENE2391X, GENE2402X, GENE2404X, GENE2426X, GENE2553X
GENE2668X, GENE454X, GENE464X, GENE524X, GENE530X
GENE537X, GENE542X, GENE563X, GENE584X, GENE586X
GENE588X, GENE598X, GENE622X, GENE639X, GENE640X
GENE642X, GENE650X, GENE651X, GENE653X, GENE669X
GENE685X, GENE694X, GENE834X, GENE844X, GENE849X

2.2 One class versus two others combined

In our exemplary classification task, one of the classes (DLCL) is larger than two others and there is set of 8 genes each of which perfectly separates this class from the others. This leads to a situation where, in the final set of genes, there appear too many genes that separate DLCL from the rest. Another problem is that there can be many genes of high mRI which are highly correlated (or, more generally, co-dependent) and, hence, whose classification ability is essentially the same. In the example studied, there are tens of genes capable of distinguishing between DLCL samples and the other samples next to perfectly. It follows that only a fraction of such genes is needed to be included in the set of genes on which to build a classifier.

In order to (possibly and at least partly) alleviate these problems, two different classes have been merged three times, thus producing three different two-class problems. (One should keep here in mind that it is a matter of the "geometry" of the classes whether such a move is a proper means – while the three classes can be relatively easily separable, combining two of them into one class can make the newly formed class hardly distinguishable from the third.) For each such problem, 60,000 random draws of 100 out of 4026 genes were made. For each draw, 30 trees were trained on 40 (randomly selected) samples and tested on the remaining 22 samples. From each 2-class classification problem, we selected 15 genes with the highest mRI, what gave us 45 genes in the final set of most informative genes:

GENE1192X, GENE1553X, GENE1583X, GENE1602X, GENE1606X
GENE1610X, GENE1613X, GENE1622X, GENE1625X, GENE1647X
GENE1661X, GENE1662X, GENE1672X, GENE1673X, GENE1676X
GENE2309X, GENE2340X, GENE2368X, GENE2402X, GENE2416X
GENE2426X, GENE2553X, GENE2668X, GENE30X, GENE3283X
GENE3285X, GENE3286X, GENE3704X, GENE3705X, GENE3969X
GENE454X, GENE524X, GENE530X, GENE537X, GENE542X
GENE598X, GENE622X, GENE639X, GENE642X, GENE651X
GENE653X, GENE669X, GENE685X, GENE694X, GENE844X

2.3 Two out of the three classes

In this version, we again aimed at avoiding the effects of having relatively many DLCL samples which can trivially be separated by the group of 8 genes. Again three runs have been processed but for each run only two classes have been selected. For example, the first run was processed on the following two classes: DLCL vs. FL. All the parameters, i.e. the number of random draws of 100 genes and the number of trees for each draw, were the same as in previous experiment. Thus, we run three times 1,800,000 trees. For the final set of genes, we selected 15 genes with the highest mRI from each run, what again gave us 45 in final set.

GENE1602X, GENE1605X, GENE1606X, GENE1610X, GENE1611X
GENE1613X, GENE1619X, GENE1622X, GENE1672X, GENE1673X
GENE2244X, GENE2346X, GENE2356X, GENE2364X, GENE2368X
GENE2378X, GENE2391X, GENE2402X, GENE2404X, GENE2426X
GENE2547X, GENE2553X, GENE2554X, GENE3497X, GENE3792X
GENE459X, GENE528X, GENE530X, GENE537X, GENE563X
GENE584X, GENE586X, GENE626X, GENE639X, GENE640X
GENE655X, GENE669X, GENE694X, GENE717X, GENE733X
GENE760X, GENE816X, GENE832X, GENE844X, GENE849X

3 Comparing the final sets of informative genes

In order to confirm that the genes found are truly informative in terms of their prediction capacity, we have performed the following experiment (for each version, we stuck to the 45 most informative genes obtained). The number of genes in the final set has been set arbitrarily. For each final set, we randomly split 62 samples into two sets 1 million times, one subset to serve as a training set and another as a test set. Training set always contained 40 samples. In this way, for each final set, 1,000,000 trees were obtained and their weighted accuracies stored.

In the fourth part of this same experiment, first, 100,000 sets of samples were randomly obtained, each with 45 genes (features) drawn at random. Each set of the samples was later randomly split into two subsets 10 times. As

Table 1. The table shows the numbers of common genes for pairs of the final sets; 1: 3-class classification; 2: one class versus two; 3: two out of three classes

method	1	2	3
1	x	30	26
2	30	x	17
3	26	17	x

previously, training set contained 40 samples. This time, therefore, 1,000,000 classification problems were obtained.

Results for *wAcc* for all 4 versions of the experiment are summarized by boxplots in Fig. 3. No doubt, the result obtained strongly supports the claim suggested at the beginning of the section.

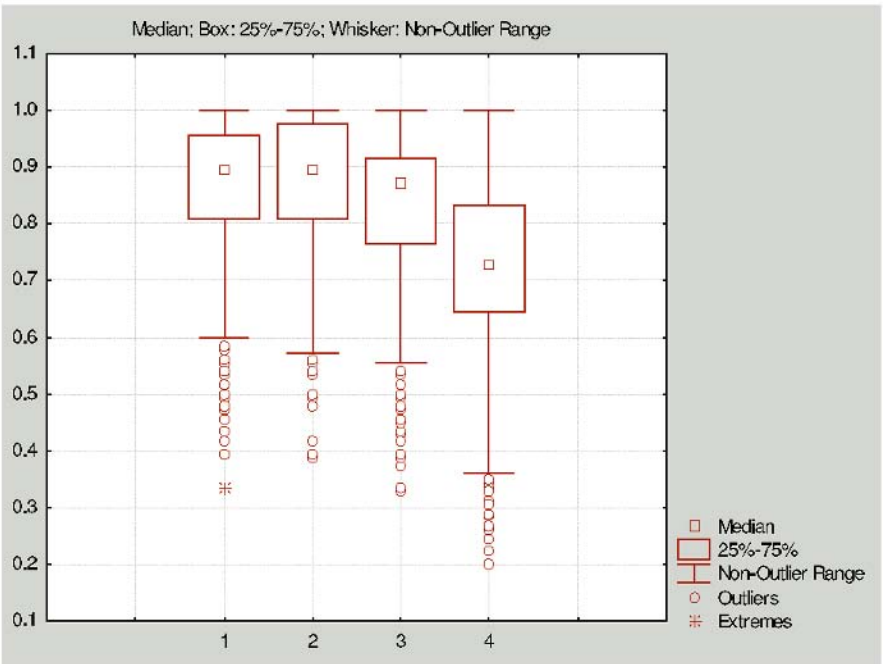


Fig. 1. *wAcc* for 45-feature samples; 1: 3-class classification; 2: one class versus two; 3: two out of three classes; 4: random selection of genes

It appears that the first (3-class classification) and the second (one class versus two) approaches have the highest median of *wAcc*. Interestingly, the second approach is slightly better than the first. It follows that, for the given

input data, the second approach has led to obtaining a slightly better set of attributes. It seems that (some of) the genes that better separate small classes (FL with 9 samples and CLL with 11 samples) have been included into the final set, instead of those that separate DLCL class from the rest. For the first approach, genes that separate DLCL have strong mRI because they occur more frequently in the trees (there are 8 gens that perfectly separate DLCL). When relying on the second approach, a much smaller number of outlying *wAcc*'s has been obtained. That is, for randomly chosen training sets, the results of classification prove more stable when the second approach is used. However, in the first approach only one run of the procedure is needed, while in each of the next two we needed three runs. All in all, the second approach, where we considered one class versus the rest, is slower than the first but has provided (slightly) better results.

4 Concluding remarks

It is clear that more experimental studies are needed to reliably evaluate practical merits of the approach proposed. Yet, it can be claimed already now that, with this approach, we are able to find truly important genes for classification, regardless of a classifier to be later used.

A type of objectivity of the gene ranking produced is a consequence of having used a sufficiently flexible classifier, namely a tree classifier, as well as of resting on Monte Carlo approach with all evaluations of the classifiers based on test sets (which are not used to build a classifier). Since we distinguish between gene selection and classification, when selecting informative genes we can use a test set of relatively big size without thus deteriorating classifier's performance.

Acknowledgements: Michał Dramiński and Jacek Koronacki were partially supported by the Swedish project, Human Research Potential and the Socio-economic Knowledge Base: Access to Research Infrastructures, HPRI-CT-2001-00153. Jan Komorowski was supported by the Wallenberg Foundation and The Swedish Foundation for Strategic Research.

References

1. Alizadeh, A.A. et al.: Distinct Types of Diffuse Large B-cell Lymphoma Identified by Expression Profiling. Nature **403**, (2000), 503-511.
2. Breiman, L. et al.: *Classification and Regression Trees*. Wadsworth International Group, 1984.
3. Dramiński M., Koronacki J., Ćwik J., Komorowski J. Monte Carlo gene screening for supervised classification Proceedings of the EUROFUSE 2004, B. De Baets, R. De Caluwe, G. De Tr, J. Fodor, J. Kacprzyk, S. Zadrozny (Eds.), pp. , Exit (2004).

4. Simon, R. et al.: Pitfalls in the Use of DNA Microarray Data for Diagnostic and Prognostic Classification. J. National Cancer Inst. **95** (2003), 14–18
5. Speed, T. [ed.]: *Statistical Analysis of Gene Expression Microarray Data*. Chapman & Hall/CRC, 2003.
6. Tibshirani, R. et al.: Class Prediction by Nearest Shrunken Centroids, with Applications to DNA Microarrays. Statistical Science **18**, 104-117.
7. Tibshirani, R. et al.: Class Prediction by Nearest Shrunken Centroids, with Applications to DNA Microarrays. Statistical Science **18**, 104-117.
8. Wittenn Ian H., Eibe Frank: Data Mining: Practical machine learning tools with Java implementations. Morgan Kaufmann, San Francisco, 2000.

Spatial Telemetric Data Warehouse Balancing Algorithm in Oracle9i/Java Environment

Marcin Gorawski and Robert Chechelski

Silesian University of Technology, Institute of Computer Science

Abstract. Balancing of parallel systems workload is very essential to ensure minimal response time of tasks submitted to process. Complexity of data warehouse systems is very high with respect to system structure, data model and many mechanisms used, which have a strong influence on overall performance. In this paper we present a dynamic algorithm of spatial telemetric data warehouse workload balancing. We implement HCAM data partitioning scheme which use Hilbert curves to space ordering. The scheme was modified in a way that makes possible setting of dataset size stored in each system node. Presented algorithm iteratively calculates optimal size of partitions, which are loaded into each node, by executing series of aggregation on a test data set. We investigate both situation in which data are and are not fragmented. Moreover we test a set of fragment sizes. Performed system tests conformed possibility of spatial telemetric data warehouse balancing algorithm realization by selection of dataset size stored in each node. Project was implemented in Java programming language with using of a set of available technology.

1 Introduction

A spatial telemetric data warehouse system (STDW) gathers telemetric information about media utilities consumption (gas, energy, water, heat) [5].

The solution is two-layered telemetric infrastructure which consists of:

1. a telemetric system of integrated meters reading,
2. a spatial warehouse system of telemetric data.

The telemetric system of integrated meters reading is based on AMR technology (Automated Meter Reading) and GSM/GPRS. The system enables to transfer data from media consumption meters, localized on a huge geographical area, to the telemetric server using a GSM cellular phone network in a GPRS technology. The spatial telemetric data warehouse system (STDW) is a support system for making a tactical decisions about media production amount based on a short-term prognoses of its using. The forecasts are made basing on analysis of the data stored in the data warehouse, supplied with data from a telemetric server. The telemetric server is a source of measured data, that are loaded into the STDW database during an extraction process.

1.1 Architecture of Spatial Telemetric Data Warehouse

The STDW system is based on a cascaded star data model indexed by an aR-tree. The system is implemented on Oracle9i/Java platform. The architecture

of STDW is a distributed structure with one client and a few servers. The system makes possible increasing of performance by parallel tasks execution. In the distributed data warehouse STDW the data are stored across many nodes, each of them is managed by independent Oracle9i RDBMS.

STDW has a distributed architecture of shared nothing type [9]. Every node in that architecture has its own processor, a memory area and disc. The communication between nodes is made only through network connections (message passing) in a RMI technology (Remote Method Invocation). The RMI technology allows communication between objects located on different nodes. Every object situated on a server implements methods, which can be invoked by applications on a client site. The RMI sends objects across network (along with methods and fields values) through the agency of TCP/IP protocol.

The source data for STDW system are partitioned among N nodes so, that every node possess the same data schema. A fact table, which usually takes more than 90% of the space occupied by all the tables, is splited into N fragments with fixed sizes. Dimension's tables are replicated into all nodes in the same form [1]. Distributed processing phase is executed in each node only on a data fragment stored in it.

1.2 The Need of STDW System Balancing

The main reason for the STDW distribution is parallel execution of following operations: data loading, indices creation, query processing, resumption of ETL process. The data gathered in a distributed data warehouse are stored among many nodes, each one is maintained by an independent RDBMS.

There is a few methods of increasing parallel systems performance: tree-based data structures, grid files, chunking. The tree-based data structures are used for declustering of a data set making a data cube and to splitting it into a number of partitions. Partitions are in turn allocated to distributed system nodes. Grid files or cartesian product files are very strong and important techniques of improving distributed systems performance. A degree of parallel processing is determined by a data distribution amongst nodes, called Data Allocation Strategy.

Our previous solutions of the STDW system balancing concerned on the case in which the system consisted of nodes having the same parameters or the same performance characteristics. The essence of the problem considered in this work is STDW system balancing (later called B-STDW) which is based on computers with different performance characteristics. The differences concern: computational capacity, size of operational memory and I/O subsystem throughput.

The rest of the paper is organized as follows. Section 2 describes the B-STDW system and figures out all the features significant to a balancing point of view. Section 3 provides a description of our algorithm of balancing, and

section 4 presents and discuses results of performed tests. Conclusions and discussion on future work is contained in section 5.

2 Fundamentals of the B-STDW Balancing

The average response time of tasks submitted to execute is minimized in balancing process in order to increase an overall efficiency of B-STDW system. Average response time is defined as a average interval between a moment when job was submitted to a system and a moment when it leaves after processing [10].

In the B-STDW system each node performs the same query only upon a piece of data stored in it. Size of the piece must be properly selected to ensure the same work time of a particular system nodes. In the case of STDW systems based on computers with the same parameters balancing is treated as ensuring the same load of every disk. This is achieved by using a proper data allocation scheme. The main task of the scheme is to guarantee that every node has the right contribution in evaluating different type of queries.

2.1 Data Allocation in the STDW

To deal with summarized data, which are the most important in the B-STDW system, an aR-tree with a virtual memory mechanism was added to the system. It causes that aggregates are computed as they come during query execution (unless they were computed in a response to previous queries), and that the query determines the data that will be drawn to an aggregation. The next advantage of this index structure is that the time of drawing the data already aggregated is very small in comparison to the time of aggregation, when huge amounts of detailed data are pulled out from data warehouse [5].

Analysis, currently supported by the B-STDW, concern on dimension containing measures data. Facts stored in it have a spatial dimension and a time dimension. The spatial dimension consist of three attributes of space location (X, Y, Z) describing localization of meters. Time dimension contains dates of measurements. The B-STDW support few types of meters. Each one has a different granularity in the time dimension (from a few minutes to a few hours). This causes that the cardinality of counter readings may vary [5,6].

Keeping in mind the features of the STDW system and its balancing presented above we decided to use data allocation schema giving good parallelism of responses on range queries (currently supported) and to set dataset size stored in all system nodes. The setting is performed in a way giving the same average work time of each node. In addition the fact table is partitioned in a way giving the same reading set size for each meter.

We chose HCAM as an allocation method. The choice was motivated by its adequateness for range queries. The method is based on the idea of a space filling curves. The curve visits every point of the space exactly once and never

cross itself [4]. It could be used for linear ordering of grid blocks made from a dimensions subset. In this schema Hilbert curve is used to select next grid block that is allocated next to the node in round-robin fashion. The idea is based on the fact that two blocks close in linear space should be also close in k-dimensional space, thus they should be allocated on a different nodes in order to ensure better parallelism of query execution [2,7].

3 The B-STDW System's Balancing Algorithm

We determine efficiency of particular nodes to match size of data set loaded in it. It is done by computation of the same aggregation on a test set by all nodes. The measure of efficiency is obtained aggregation time. The test set (used in balancing) is a subset of fact table.

For description purposes we introduce a concept of a fact table division factor — p_i. It represents a value being a ratio of fact table size stored into an i node to total size of fact table. Using aggregation times of test set, obtained by every node, the algorithm iteratively computes values of p_i factors in order to obtain a work time of a node similar to average work time.

Algorithm 1:

1. Load dimension tables to all nodes in the same form,
2. Set all fact table division factors as $p_i = 1/N$,
3. Load test subset of fact table, partitioned according to division factors, to all nodes,
4. Aggregate all test subset data,
5. Maximum imbalance is less than assumed ? Yes — go to 7, No — go to 6,
6. Correct division factors p_i using aggregation times, go to 3,
7. Load the whole fact table, partitioned according to last computed division factors, into all nodes.

A draft of balancing algorithm is presented above as Algorithm 1. First step of the algorithm is loading of dimension tables to all nodes — the same copy of dimension tables is loaded into each node. Then factors p_i for all nodes are set to value $1/N$ where N indicates the total number of nodes. Next step of algorithm is a fact table partition plan calculation (first part of step 3 in Algorithm 1). This is done in a way presented as Algorithm 2.

Algorithm 2:

1. Calculate h-value for a localization of each meter.
2. Sort all meters by h-value ascendant,
3. Split meter measurements into chunks with the size set by user and order into sequence: <chunk 1 of meter 1 measurements>, <chunk 2 of meter 1 measurements>, <chunk 1 of meter 2 measurements>, (meters order according to point 2),

4. Allocate chunks to nodes using round-robin method with skipping — to preserve the value of a divide factor for i node.

We use a code from [8] to compute h-values (transformation of k-dimensional space into linear with using Hilbert curve).

The algorithm next sends a partition table to all nodes using previously prepared partition plan. Described step executes splitting of fact table into partitions loaded to particular nodes using HCAM method. Moreover it respects precalculated dataset sizes for each node and uses chunking.

The next step is an aggregation that is performed using range query supplied by user. Every iteration uses the same query. Algorithm measures time of aR-trees building (trees are removed from servers memory before each execution of query).

On the basis of obtained aggregation times, imbalances of all system nodes with relation to the fastest node are calculated according to the formula (1)

$$imbalance_i = \frac{T_i - T_{fast}}{T_{fast}} \tag{1}$$

where: T_i and T_{fast} are aggregation times of i^{th} and the fastest node.

If maximum imbalance (amongst the all calculated) is greater than assumed then correction of fact table division factors (p_i) is performed (Algorithm 3)

Beside the aggregation times, the correction algorithm takes two additional factors: corrP, used when division factor is increased, and corrN, used when the factor is decreased. The correction is performed according to schema presented below as Algorithm 3.

Algorithm 3:

1. Calculate an average aggregation time of all nodes — avg
2. For each node calculate its imbalance of aggregation time with relation to avg. The imbalance is used next to correction of p_i factors. Imbalance is calculated according to formula (2).

$$imbalance_i = \frac{T_i - avg}{avg} \tag{2}$$

   (a) if imb > 0 then make correction using formula (3)

$$p_i = p_i \cdot (1 - corrN \cdot imb) \tag{3}$$

   (b) if imb < 0 then make correction using formula (4)

$$p_i = p_i \cdot (1 - corrP \cdot imb) \tag{4}$$

   (c) if imb $= 0$ then do not make correction
3. Correct all factors to 100%

When maximum imbalance is below assumed, the whole fact table is partitioned and loaded into nodes taking into consideration division factors calculated during the last iteration of balancing algorithm (by using the Algorithm 2).

4 Tests

Test platform was composed of 7 nodes (6 servers and a maintainer). Source data was stored on the maintainer node, that was responsible for test program realization, especially balancing (implementing algorithm described as Algorithm 1). Servers were performing operations ordered by the maintainer on their copy of data. Server nodes was configured as follows: 2 nodes — 2.8 GHz Intel Pentium 4, 512 MB RAM, 120GB 7200 RPM IDE HDD, 2 nodes — 1.8 GHz Intel Pentium 4, 256 MB RAM, 80GB 7200 RPM IDE HDD, 2 nodes — 1.7 GHz Intel Pentium 4, 256 MB RAM, 80GB 7200 RPM IDE HDD. The maintainer node had a configuration identical to the first two server nodes. Each node worked under Windows XP, had installed Oracle9i server and Java Virtual Machine v. 1.4.2_04. All the nodes was connected by 100Mbit Ethernet network.

The first test, the results of which are presented in fig. 1, checked the influence of chunk size changes on the balancing process. Graphs show, that bigger fragment sizes result in smoother graph of maximum imbalance (imbalance of the slowest node with respect to the fastest one) — imbalance function has fewer points of growing. Analyzing the influence of chunk size on balancing time we see that smaller fragments lengthen balancing process. In a case when chunk size was set to 500 tuples, finishing the balancing process was impossible. The reason was the decreasing of the data set size of the weak nodes to under 1% while the imbalance was still growing.

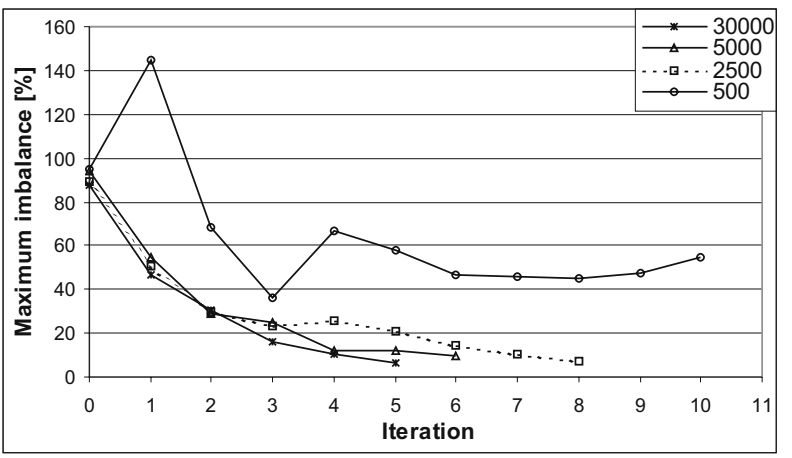


Fig. 1. Influence of chunk size on time of balancing process

The results are due to the construction of actual aggregation query. The query draws all measures for each meter and then performs sorting operation on it. When chunk size is decreasing the number of fragments increases. In

case when chunk size is set to 500 tuples, the situation that every node sorts measures of each meter is highly possible (measures are splited into 10 fragments at least). This is the reason why dataset sizes allocated on the weak nodes constantly decreased.

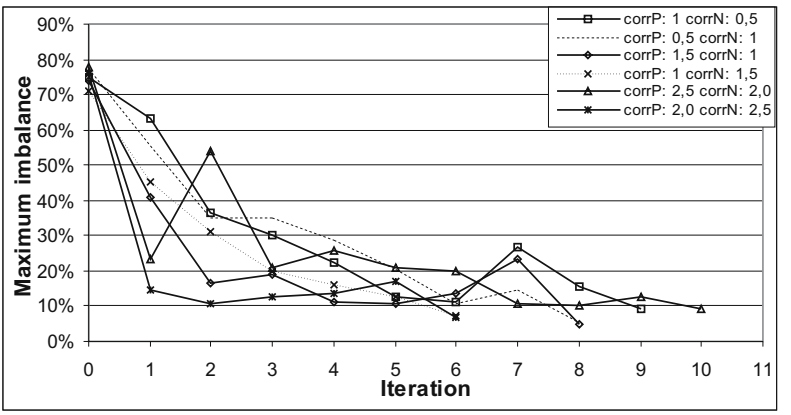


Fig. 2. The balancing process for different pairs of correction factors

In the second test we checked the influence of corrP and corrN changes on balancing process, graphs are presented on fig. 2. We have observed that bigger values of factors result in quick decreasing of imbalance, but the oscillating increasing of imbalance was often spotted. It is highly visible in the case of 2.5/2 and 2/2.5 pairs.

The final test checked how the same pairs of correction factors worked in different configurations of B-STDW (results presented on fig. 3). In 4-nodes configuration the strongest nodes was removed. The obtained results show that the values of correction factors should be set according to actual system imbalance. For configuration consisting of 4 nodes we have observed that the better was the pair with the factor of increasing having greater value, and in 6 nodes configuration the contrary pair returned better results.

We wish next to summarize the conclusions drawn from the performed tests:

1. Using of small size fragments, for aggregation query is actually realized in the B-STDW system, could lead to problems in system balancing and response time for user queries may grown,
2. A higher values of correction factors result in a fastest imbalance decreasing, but the total imbalance time may be greater,
3. The values of correction factors should be tightly calculated according to actual system imbalance.

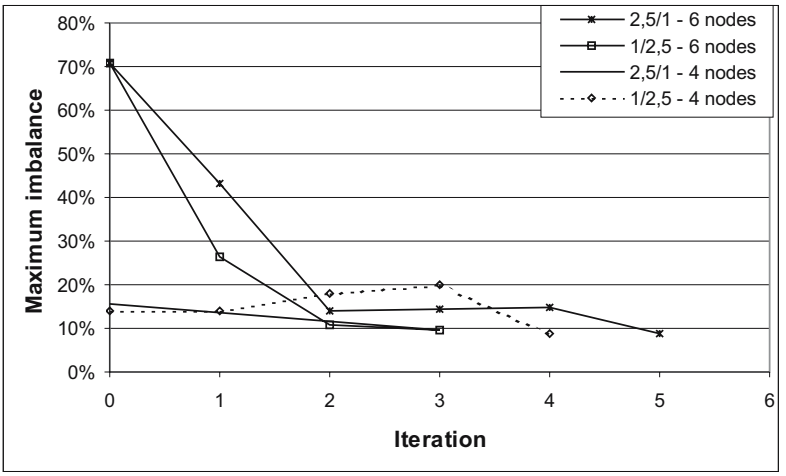


Fig. 3. Balancing process for the same correction factors pairs in different B-STDW environments

5 Conclusions

In this paper we presented a spatial telemetric data warehouse balancing algorithm. The results of our work is the designing of a preliminary stage of a balancing algorithm which gives promising results and the creating of an environment enabling the testing of the algorithm as well as the testing of particular system components. Java programming language and a large set of available technology was used to implement the system.

Preliminary results conformed the possibility of spatial distributed data warehouse balancing algorithm realization using the method of selecting dataset size stored in each system node.

Presented system could be developed in the following directions:

- improvement of balancing process in terms of: shorten of process duration time and achievement of smallest imbalances by a dynamic selection of correction factors,
- balancing of the data warehouse during data actualizations,
- look up over a different data allocation strategies with full support to grid files technique,
- using multi-processors and multi-disks machines in the B-STDW system.

References

1. Bernardino J., Madeira H.: Data Warehousing and OLAP: Improving Query Performance Using Distributed Computing. CAiSE*00 Conference on Advanced Information Systems Engineering Stockholm, 2000

2. Dehne F., Eavis T., Rau-Chaplin A.: Parallel Multi-Dimensional ROLAP Indexing. 3rd International Symposium on Cluster Computing and the Grid, Tokyo, Japan 2003.
3. C. Faloutsos and P. Bhagwat: Declustering using fractals in Proc. of the Int'l Conf. on Parallel and Distributed Information Systems, San Diego, California, January 1993, pp. 18-25.
4. Faloutsos C., Roseman S.: Fractals for Secondary Key Retrieval. Technical Report UMIACS-TR-89-47, CS-TR-2242, University of Maryland, Colledge Park, Maryland, May 1989
5. Gorawski M., Malczok R.: Distributed Spatial Data Warehouse Indexed with Virtual Memory Aggregation Tree, 5th Workshop on Spatial-Temporal DataBase Management (STDBM_VLDB'04), Toronto, Canada 2004.
6. Han, J., Stefanovic, N., Koperski, K. Selective Materialization: An Efficient Method for Spatial Data Cube Construction. In Research and Development in Knowledge Discovery and Data Mining, Second Pacific-Asia Conference, PAKDD'98, Melbourne, Australia, 1998.
7. Hua K., Lo Y., Young H.: GeMDA: A Multidimensional Data Partitioning Technique for Multiprocessor Database Systems. Distributed and Parallel Databases, 9, 211-236, University of Florida, 2001
8. Moore D. Fast hilbert curve generation, sorting, and range queries. http://www.caam.rice.edu/~dougm/twiddle/Hilbert
9. Papadias D., Kalnis P., Zhang J., Tao Y.: Efficient OLAP Operations in Spatial Data Warehouses. Spinger Verlag, LNCS 2001.
10. Zeng Z., Bharadwaj V.: Design and analysis of a non-preemptive decentralized load balancing algorithm for multi-class jobs in distributed networks. Computer Communications, 27, pp. 679-694, 2004

Comparison of Efficiency of Some Updating Schemes on Bayesian Networks

Tomasz Łukaszewski

Poznan University of Technology, Institute of Computing Science, Poznań, Poland

Abstract. The problem of efficiency of general reasoning with knowledge updating on Bayesian networks is considered. The optimization should take into account not only the reasoning efficiency but also the prospective updating issues. An improved updating process based on an idea of data removing is proposed. Further possible improvement of the reasoning architecture is presented. Comparison of existing and proposed approaches are made on a basis of a computational experiment. Results of this experiment are presented.

1 Introduction

Most Artificial Intelligence approaches to reasoning under uncertainty are based on Bayesian schemes. *Bayesian networks* (directed acyclic graphs) are successfully applied in a variety of problems, including machine diagnosis, user interfaces, natural language interpretation, planning, vision, robotics, data mining and many others. The most common task performed on Bayesian networks is a *general evidence propagation* - the calculation of the posterior marginal distributions of all non-evidence variables for a given set of evidence variables. This kind of reasoning is performed mainly by *message-passing* algorithms in a *join tree*. A Bayesian network is preprocessed into this single connected structure to avoid update anomalies and incorrect results. The best known message-passing algorithms are Lauritzen-Spiegelhalter [4], Hugin [1], Shafer-Shenoy [6]. These classical approaches have been the subject of many improvements, that in general explored the *static* nature of the reasoning process. However, in real intelligent systems, data (values of variables and distributions of probabilities in Bayesian networks) are subject to changes. The main reasons of these changes are the following:

- not precise data evaluation
- changes of the surrounding world
- performing the reasoning process for some test data

Such a change of data takes place more than once, for example in adaptive systems that natural feature are continuous changes of some data. The main motivation of this paper is the fact, that usually the response time in adaptive systems is a critical factor and an improvement of efficiency of the reasoning with data changes reduce costs (process controlling applications) or even save life (medical and military applications).

This paper is a part of our work on reasoning with data changes, called also the *reasoning with knowledge updating*. We show that the efficiency of the general evidence propagation with knowledge updating on Bayesian networks can be optimised on the following levels:

- (*Higher*) optimisation of updating processes in a given join tree
- (*Lower*) optimisation of the join tree structure

Let us notice, that in practise for a given updating process we optimise a join tree structure or optimise both the updating process and a join tree structure for this process. Let us call these approaches *L* and *HL* respectively.

The paper is organized as follows: in Section 2 we describe a simple updating process and a new improved approach. Section 3 describes the problem of the join tree structure optimisation. Section 4 presents results of the comparison of existing and proposed approaches to the reasoning with knowledge updating on Bayesian networks.

2 Optimisation of the updating process

Let us remind, that the reasoning process is performed by message-passing algorithms in a join tree build for a given Bayesian network. In a join tree vertices are sets of variables of the original Bayesian network. A join tree is usually enhanced by *separators* associated to each edge. A separator contains the intersection of the variables assigned to the connected vertices. Moreover, to each vertex V_i and separator S_i we associate potentials ϕ_{V_i} and ϕ_{S_i} respectively that are functions having the variables of V_i and S_i as domains. Initialy potentials ϕ_{V_i} are combinations of conditional probability distributions of the original Bayesian network and $\phi_{S_i} = 1$. The reasoning process is carried out by passing messages (potentials) between vertices in two phases: from leaves towards an arbitrary chosen root and in the opposite direction.

We define the knowledge update as changes of variables (observations, evidence) or changes of conditional probability distributions of the original Bayesian network. These changes update a correspondent potential function (a correspondend vertex of the join tree). We assume, that the knowledge updates do not change the structure of a Bayesian network. Each knowledge update forces the repetition of this reasoning process. However, some calculations need not be repeated. Let us call them *initial calculations*. They may be used as a base for *updating calculations*. The goal is to design an *updating process* so that the effort of updating calculations is minimised.

In this section we present a simple updating process, and an improved approach. Then we define the complexity of both approaches in terms of the number of elementary operations (additions). These approaches are compared in the section 4.

2.1 Simple updating process

A simple idea of the updating process, avoiding unnecessary computations is presented in [6]. This updating process was proposed to the Shafer-Shenoy algorithm and is based on the following observation: an updated vertex sends updating messages to all the other vertices. We identify updating messages in the following way:

1. Associate a mark "0" to all the messages of a join tree.
2. For each updated vertex: set a mark to "1" for all messages on all paths from the updated vertex to leaves of the join tree.

Messages with the mark "1" have to be updated. The number of updating messages is less or equal to the number of all messages. This updating process works efficiently, because there are two separators between any pair of neighbour vertices. These separators store messages passed in both directions. Updating messages have to be recalculated, not changed messages are retrieved from separators.

This updating process can not be applied without any changes to the Lauritzen-Spiegelhalter and Hugin algorithms. In these two algorithms we do not store messages in separators but potentials in vertices. This implicates, that we have to recalculate not only updating messages but also not changed ones. However, the efficiency of the updating process in these algorithms can be improved adding separators between any pair of neighbour vertices. In this way, not changed messages can be retrieved from these separators.

This extension of these two algorithms is motivated by the results of the comparison of all the three algorithms carried out for general reasoning without updating [7]. In the light of this comparison we can not make any general statement regarding relative computational efficiency of Hugin and Shafer-Shenoy algorithms their relative efficiency depends on the structure of a Bayesian network (Lauritzen-Spiegelhalter algorithm is always less efficient than Hugin algorithm). It seems that a comparison of these algorithms carried out for general evidence propagation with knowledge updating could lead to the same results. Moreover, we are interested in the Hugin approach for another reason - in the next section we present an idea of improved updating process dedicated to this message-passing architecture.

2.2 Improvement of the updating process

The improvement of the updating process in the Hugin architecture is based on the idea of removing an *old* message from a potential associated to a vertex and absorbing by this vertex an updating message. In this way we send *only* updating messages. The removing phenomenon is a part of the Hugin algorithm and takes place in the outward phase when we remove from a message data passed in the opposite, inward phase (we store this data in a separator). This removing is defined as a division of two potentials. Let us

notice that in the improved updating process *each* updating message contains data passed in the opposite direction and we have to refine *each* updating message from this data. We will store this data in two separators between any pair of vertices – one for each direction.

Summarizing, we send updating messages from leaves towards a root of the join tree (inward phase), and then backwards (outward phase). Passing of an updating message from a source to a target vertex consists of the following steps:

1. Recalculate the updating message from the potential of the source.
2. Remove data passed in opposite direction from the updating message.
3. Remove the old message from the potential of the target.
4. Store refined updating message in the separator and send it to the target.

Let us notice, that we can improve this process changing the order of the two last operations: we can divide the refined updating message by the old refined message and the result of this operation is sent to a target vertex division is carried out for smaller (or the same size) potentials. In the same way Hugin approach improved the Lauritzen-Spiegelhalter one. The correctness of the improved updating process comes from the analysis of the Hugin algorithm.

Let us compare the simple and the improved approach. In the simple one we recalculate updated messages, not changed messages are retrieved from separators. In the improved approach we only recalculate updated messages. However, we have to take into account the fact that in the improved approach message-passing is more complicated. Moreover, initialization of potentials differs in both approaches. In the simple updating process initial potential of each updated vertex is multiplied by all updated vectors of variable values. In the improved updating process we update current potentials of each updated vertex – for each updated variable we divide this potential by old vector of variable values and multiply by new one. We do not make any general statement regarding relative computational efficiency of these approaches, however we expect, that the improved updating process performs better than the simple one in cases where the number of updating messages is fewer than the number of all messages. Below we define the exact complexity of both approaches.

2.3 Complexity of updating processes

Let us start with the definition of two basic operations performed on potentials and their complexity. Sets of variables are denoted by boldface uppercase letters ($\mathbf{X}, \mathbf{Y}, \mathbf{Z}$) and their instantiations by boldface lowercase letters ($\mathbf{x}, \mathbf{y}, \mathbf{z}$). We define two basic operations on potentials: *marginalization* and *multiplication*. Suppose we have a set of variables $\mathbf{Y}$, its potential $\phi_{\mathbf{Y}}$, and a set of variables $\mathbf{X}$ where $\mathbf{X} \subseteq \mathbf{Y}$. The marginalization of $\phi_{\mathbf{Y}}$ into $\mathbf{X}$ is a potential $\phi_{\mathbf{X}}$ where each $\phi_{\mathbf{X}}(\mathbf{x})$ is computed as follows:

1. Identify the instantiations $\mathbf{y}_1, \mathbf{y}_1, ...$ that are consistent with $\mathbf{x}$.
2. Assign to $\phi_{\mathbf{X}}(\mathbf{x})$ the sum $\phi_{\mathbf{Y}}(\mathbf{y}_1) + \phi_{\mathbf{Y}}(\mathbf{y}_2) + ...$

Given two sets of variables $\mathbf{X}$ and $\mathbf{Y}$ and their potentials $\phi_{\mathbf{X}}$ and $\phi_{\mathbf{Y}}$, the multiplication of $\phi_{\mathbf{X}}$ and $\phi_{\mathbf{Y}}$ is a potential $\phi_{\mathbf{Z}}$, where $\mathbf{Z} = \mathbf{X} \cap \mathbf{Y}$ and each $\phi_{\mathbf{Z}}(\mathbf{z})$ is computed as follows:

1. Identify the instantiations $\mathbf{x}, \mathbf{y}$ that are consistent with $\mathbf{z}$.
2. Assign to $\phi_{\mathbf{Z}}(\mathbf{z})$ the product $\phi_{\mathbf{X}}(\mathbf{x})\phi_{\mathbf{Y}}(\mathbf{y})$.

Let us define the complexity of these two operations in terms of the number of elementary operations (additions). We assume, that each variable has only two states (true and false). For the marginalization $\phi_{\mathbf{Y}}$ into $\mathbf{X}$ we have summation of $2^{|\mathbf{Y}|-|\mathbf{X}|}$ elements repeated $2^{|\mathbf{X}|}$ times, what gives the total number of elementary operations:

$$2^{|\mathbf{X}|}(2^{|\mathbf{Y}|-|\mathbf{X}|} - 1) = 2^{|\mathbf{Y}|} - 2^{|\mathbf{X}|} \tag{1}$$

For multiplication of $\phi_{\mathbf{X}}$ and $\phi_{\mathbf{Y}}$ the total number of elementary operations is the following (α is a coefficient that represents the relative complexity of multiplication versus addition):

$$\alpha \, max(2^{|\mathbf{X}|}, 2^{|\mathbf{Y}|}) \tag{2}$$

Now, let us define the complexity of the simple and the improved updating processes. We assume that each variable has only two states (true and false). We define the number of elementary operations (additions) carried out by message passed between any pair of neighbour vertices $\{V_i, V_j\}$ during the inward phase (from V_i to V_j) and the outward phase (from V_j to V_i). We use the additional notation:

- $|V_i|$ is the number of variables associated to the vertex V_i
- u_i is the number of updated variables associated to the vertex V_i
- $\mathbf{U}$ is the set of updated vertices
- $\mathbf{E}$ is the set of edges of a join tree
- β represents the relative complexity of division versus addition
- w_{ij} is equal to 1 if the message from V_i to V_j has to be recalculated (updating message); 0 otherwise

Complexity of the simple updating process

- Initialization of potentials in updated vertices:

$$\sum_{V_i \in \mathbf{U}} u_i \, \alpha \, 2^{|V_i|} \tag{3}$$

- Inward phase – marginalization and multiplication:

$$\sum_{\{V_i,V_j\}\in\mathbf{E}} w_{ij}(2^{|V_i|} - 2^{|V_i\cap V_j|}) + (\alpha\, 2^{|V_j|}) \tag{4}$$

- Outward phase – marginalization, division and multiplication:

$$\sum_{\{V_i,V_j\}\in\mathbf{E}} w_{ji}(2^{|V_j|} - 2^{|V_i\cap V_j|}) + w_{ji}(\beta\, 2^{|V_i\cap V_j|}) + (\alpha\, 2^{|V_i|}) \tag{5}$$

Complexity of the improved updating process

- Initialization of potentials in updated vertices:

$$\sum_{V_i\in\mathbf{U}} u_i(\beta\, 2^{|V_i|} + \alpha\, 2^{|V_i|}) \tag{6}$$

- Inward phase – marginalization, double division and multiplication:

$$\sum_{\{V_i,V_j\}\in\mathbf{E}} w_{ij}(2^{|V_i|} - 2^{|V_i\cap V_j|}) + w_{ij}(2\beta\, 2^{|V_i\cap V_j|}) + w_{ij}(\alpha\, 2^{|V_j|}) \tag{7}$$

- Outward phase – marginalization, double division and multiplication:

$$\sum_{\{V_i,V_j\}\in\mathbf{E}} w_{ji}(2^{|V_j|} - 2^{|V_i\cap V_j|}) + w_{ji}(2\beta\, 2^{|V_i\cap V_j|}) + w_{ji}(\alpha\, 2^{|V_i|}) \tag{8}$$

3 Optimisation of a join tree structure

A join tree can be created by *triangulation* of the (moralised) graph for a given Bayesian network. For a given Bayesian network many different triangulated graphs can be built. Finding an optimal join tree for a given Bayesian network is NP-hard. In this chapter we present the classical approach to the join tree structure optimisation, that can be applied to the reasoning with knowledge updating. Then we present the idea of fitting of a join tree structure to the updating process, that should be more efficient than the classical approach. Both approaches are compared in the section 4.

3.1 Classical approach

In the classical, static approach to the general evidence propagation, the typical objective is to obtain Bayesian network triangulation with small probability tables. Therefore, the objective is to obtain triangulations of minimum weight [3]. Let $|V_i| = n_i$. Let $C_i = V_1, ..., V_k$ represent a clique of a triangulated graph G_T. The weight of G_T is defined as follows:

$$w(G_T) = log_2 \sum_{C_i} \prod_{V_i \in C_i} n_i \tag{9}$$

For a given triangulation the optimal join tree can be build in a very efficient way [2]. The basic technique applied to triangulate a graph G (that represents a given Bayesian network) is to add the extra edges T produced by eliminating the vertices of G one by one. This technique is not guaranteed to produce optimal triangulations (in terms of the weight of the triangulated graph) when the vertices are selected at random. Several heuristic algorithms were suggested and compared for establishing elimination orderings [3]. A simulated annealing metaheuristic and evolutionary algorithm were also applied. In [5] we proposed a local search heuristic based on the idea of evolutionary algorithms and compared with other heuristics (random generated graphs) and metaheuristics (benchmark graphs). Results obtained by the evolutionary algorithm were much better than results obtained by the heuristics. For the benchmark graphs the evolutionary algorithm was able to achieve the best known results, moreover average values were better than obtained by these two other metaheuristics.

3.2 Fitting a join tree to the updating problem

Let us notice, that the classical approach optimise the sum of sizes of cliques, not the computational effort of the updating process. Therefore the objective of the join tree optimisation should be the total cost of the updating process. That should improve the effectiveness of the updating process.

We assume, that evidence and hypotheses are known *a priori*, so we are able to build a join tree minimising the number of elementary operations of the updating process. We implemented the aforementioned evolutionary algorithm in order to obtain elimination sequences of vertices. For a given elimination sequence a join tree is build by the algorithm described in [2]. Evidence or hypothesis is associated to a clique of the minimal size that contains this variable. For such a structure the number of elementary operations of the updating process is calculated. We expected, that fitting a join tree to the updating process should be more effective than the classical approach.

4 Comparison of efficciency of updating schemes

Let us define an *updating scheme* as a connection of an updating process and a join tree optimisation. We compared the classical approach (C) and two updating schemes aforementioned in the introduction of the paper:

- (C) simple updating, a join tree optimised by the classical approach
- (L) simple updating, a join tree optimised to the simple updating
- (HL) improved updating, a join tree optimised to the improved updating

Join trees were optimised by the evolutionary algorithm: tournament selection method (constant tournament size equal to 3). Crossover OX2 with the rate 40%; mutation ISM with the rate 100%. The stop criteria is the following: 400 iterations without improvement of the fitness function, maximum 7000 iterations.

The experiments were carried out for randomly generated Bayesian networks with 50 variables (n), two states of each variable and different densities (d). The number of evidence variables (e) was equal to 1, 5 and 10. The number of hypotheses (h) was fixed and equal to 2. For each configuration of graph parameters (d, e) the following steps were carried out:

- Evidence and hypotheses were randomly chosen.
- For each approach (C, L, HL) we calculated the number of elementary operations (N_C, N_L, N_{HL} respectively) carried out by its updating process as follows: the evolutionary algorithm generating join trees was run twice and for the best generated join tree the number of elementary operations was calculated.
- The value N_C was treated as a reference point. Then, we computed the relative deviations RD_L, RD_{HL} between this reference point and results obtained by two other approaches as follows:

$$RD_L = \frac{N_C - N_L}{N_C} 100\,[\%] \tag{10a}$$

$$RD_{HL} = \frac{N_C - N_{HL}}{N_C} 100\,[\%] \tag{10b}$$

For each configuration of graph parameters 50 Bayesian networks were randomly generated in order to obtain the average values of RD_L and RD_{HL}. Values α and β were set respectively to 1.17 and 3.27 by a simple experiment. Below we present the comparison of average, minimal and maximal values of RD_L and RD_{HL} and standard deviations.

Table 1. Comparison of efficiency of updating schemes ($e = 1$)

n	d	ave RD_L	min RD_L	max RD_L	σ_L	ave RD_{HL}	min RD_{HL}	max RD_{HL}	σ_{HL}
50	10	13.04	-17.99	39.75	10.50	38.93	-3.75	69.14	11.23
	30	17.20	-27.58	52.43	14.15	41.10	12.27	65.89	10.02
	50	15.89	-16.95	49.88	12.53	40.12	14.69	70.88	9.86
	70	14.76	-10.67	42.91	11.64	43.62	-6.87	70.37	12.24
	90	9.35	0.00	45.50	11.56	22.74	-150.69	71.44	40.59

Table 2. Comparison of efficiency of updating schemes ($e = 5$)

n	d	$ave\ RD_L$	$min\ RD_L$	$max\ RD_L$	σ_L	$ave\ RD_{HL}$	$min\ RD_{HL}$	$max\ RD_{HL}$	σ_{HL}
50	10	7.69	-18.20	41.84	9.98	14.35	-22.75	62.57	17.12
	30	13.30	-70.00	54.90	18.15	21.79	-49.51	67.05	25.10
	50	18.70	0.00	47.85	12.24	37.07	-11.36	67.11	15.54
	70	18.22	0.00	51.85	12.96	36.54	-42.27	71.25	23.65
	90	7.86	0.00	34.76	10.01	-34.59	-238.69	59.15	60.75

Table 3. Comparison of efficiency of updating schemes ($e = 10$)

n	d	$ave\ RD_L$	$min\ RD_L$	$max\ RD_L$	σ_L	$ave\ RD_{HL}$	$min\ RD_{HL}$	$max\ RD_{HL}$	σ_{HL}
50	10	1.33	-87.47	64.74	27.22	-0.07	54.86	59.99	17.20
	30	10.60	-31.72	52.72	14.93	16.63	-14.88	56.72	16.68
	50	14.23	-97.93	54.26	20.20	23.78	-39.68	61.85	22.46
	70	17.78	0.00	66.86	16.98	22.10	-111.38	83.14	38.02
	90	7.71	0.00	47.91	12.71	-87.71	-185.04	43.85	68.16

Results obtained by the first proposed updating scheme L show that the optimisation of the join tree structure on average performs better than the classical approach – as we expected. The efficiency depends on the structure (density) of the network – proposed updating scheme tends to be clearly faster for densities between 30 and 70 %. However, the proposed updating scheme does not outperform the classical approach in all cases, what is indicated by negative values of $minRD_L$.

Further improvement of the efficiency is achieved by the second updating scheme *HL*. Let us notice, that the improvement strongly depends on the number of updates. This is expected as the number of updates has the important influence on the complexity of the improved updating process. For some cases, this updating scheme is on average even worse than the classical approach what is indicated by negative values of $aveRD_{HL}$.

5 Conclusions

This paper addresses the issue of efficiency of the general reasoning on Bayesian networks with data changes. Classical approach to the problem consists in the performing only the necessary calculations in a join tree. Two possible

improvements are described and discussed: optimisation of the updating process in a given join tree and optimisation of a join tree structure for a given updating problem. Formulas of computational complexity are given for improved and non-improved updating processes. Based on these improvements two updating schemes were proposed.

The main contribution of this paper is the comparison of efficiency of updating schemes in terms of the number of elementary operations. The results obtained in the experiment are encouraging and show that improved updating schemes outperform the classical approach for some cases. However, further investigation is necessary. Future research trends also include a comparison of updating schemes on Bayesian networks in terms of the reasoning time.

References

1. Andersen S. K., Jensen F. V., Olesen K. G. (1990) An algebra of Bayesian Belief Universes for Knowledge-Based Systems. Networks **20**, 637–659
2. Jensen F. V., Jensen F. (1994) Optimal Junction Trees. Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence. Morgan Kaufmann Publishers, San Francisco, 360–366
3. Kjarulff U. (1990) Triangulation of Graphs – Algorithms Giving Small Total State Space. Research Report R90-09, Department of Computer Science, Aalborg University, Denmark
4. Lauritzen S. L., Spiegelhalter D. J. (1988) Local computations with probabilities on graphical structure and their application to expert systems (with discussion). Journal of Royal Statistical Society, Series B, **50**, 157–224
5. Łukaszewski T. (2003) An evolutionary algorithm for Bayesian network triangulation. Selected Papers of the International Conference on Operations Research (SOR 2002), Springer-Verlag, Berlin, 365–370
6. Shafer G., Shenoy P. P. (1990) Axioms for probability and belief-function propagation. in Shachter R.D., Levitt T.S., Lemmer J.F., Kanal L.N. (Eds), Uncertainty in Artificial Intelligence, **4**, 169–198
7. Shenoy P., Lepar V. (1998) A Comparison of Lauritzen-Spiegelhalter, Hugin, and Shenoy-Shafer Architectures for Computing Marginals of Probability Distributions. in G. Cooper and S. Moral (eds.), Uncertainty in Artificial Intelligence, Morgan Kaufmann, San Francisco, CA, **14**, 328–337

Analysis of the Statistical Characteristics in Mining of Frequent Sequences

Romanas Tumasonis and Gintautas Dzemyda

Institute of Mathematics and Informatics, Akademijos str. 4, 08663 Vilnius, Lithuania

Abstract. The paper deals with the search and analysis of the subsequences in large volume sequences (texts, DNA sequences, etc.). A new algorithm ProMFS for mining frequent sequences is proposed and investigated. It is based on the estimated probabilistic-statistical characteristics of the appearance of elements of the sequence and their order. The algorithm builds a new much shorter sequence and makes decisions on the main sequence in accordance with the results of analysis of the shorter one.

1 Introduction

Sequential pattern mining [3-6], which finds the set of frequent subsequences in sequence databases, is an important datamining task and has broad applications, such as business analysis, web mining, security, and bio-sequences analysis. We will examine just plain text information. Text mining is a variation on a field called data mining, that tries to find interesting patterns from large databases. The difference between regular data mining and text mining is that in text mining the patterns are extracted from natural language text rather than from structured databases of facts. Databases are designed for programs to process automatically; text is written for people to read. Text mining methods can be used in bioinformatics for analysis of DNA sequences. A DNA sequence (sometimes genetic sequence) is a succession of letters representing the primary structure of a real or hypothetical DNA molecule or strand, The possible of sequence letters are A, C, G, and T, representing the four nucleotide subunits of a DNA strand (adenine, cytosine, guanine, thymine), and typically these are printed abutting one another without gaps, as in the sequence AAAGTCTGAC (see [2] for details).

The problem of mining sequential patterns is formulated, e.g., in [3,4]. Assume we have a set $L=\{i_1, i_2, \ldots, i_m\}$ consisting of m distinct elements, also called *items*. An *itemset* is a nonempty unordered collection of items. A *sequence* is an ordered list of itemsets. A *sequence* α is denoted as $(\alpha_1 \rightarrow \alpha_2 \rightarrow \ldots \rightarrow \alpha_q)$, where the sequence *element* α_j is an itemset. An item can occur only once in an item set, but it can occur multiple times in different item sets of a sequence [4]. We solve a partial problem, where itemset consists of one item only. A sequence $\alpha=(\alpha_1 \rightarrow \alpha_2 \rightarrow \ldots \rightarrow \alpha_n)$ is a *subsequence* of another sequence $\beta=(\beta_1 \rightarrow \beta_2 \rightarrow \ldots \rightarrow \beta_m)$, if there exist such numbers

$t_1, t_2, ..., t_n$, where $t_{j+1} = t_j + 1,\ j = 1, ..., n$ and $\alpha_j = \beta_{t_j}$ for all a_j. Here β_{t_j} are elements of the set L. We analyze a sequence (the main sequence) S that is formed from single elements of L (not their sets, as is used in the classical formulation [3, 4] of the problem). In general, the number of elements in S is much larger than that in L. We have to find the most frequent subsequences in S. The problem is to find subsequences whose appearance frequency is more than some threshold called minimum support, i.e. the subsequence is frequent iff it occurs in the main sequence not less than the minimum support times.

The most popular algorithm for mining frequent sequences is the GSP (Generated Sequence Pattern) algorithm. It has been examined in a lot of publication (see, e. g., [1, 3]). While searching frequent sequences in a long text, a multiple reviewing is required. The GSP algorithm minimizes the number of reviewings, but the searching time is not satisfactory for large sequence volumes. Other popular algorithms are e.g. SPADE [3], PrefixSpan [7], FreeSpan [8], and SPAM [9].

In this paper, a new algorithm for mining frequent sequences (ProMFS) is proposed. It is based on estimated statistical characteristics of the appearance of elements of the main sequence and their order. It is an approximate method. The other method of this class is ApproxMAP [10]. The general idea of ApproxMAP is is that, instead of finding exact patterns, it identifies patterns approximately shared by many sequences. The difference of our method is that we estimate the probabilistic-statistical characteristics of elements of the sequence database, generate a new much shorter sequence and make decisions on the main sequence in accordance with the results of analysis of the shorter one.

2 GSP (Generated Sequence Pattern) algorithm

Let us note that if the sequence is frequent, each its possible subsequence is also frequent. For example, if the sequence $AABA$ is frequent, all its subsequences A, B, AA, AB, BA, AAB, and ABA are frequent, too. Using this fact, we can draw a conclusion: if a sequence has at least one infrequent subsequence, the sequence is infrequent. Obviously, if a sequence is infrequent, all newly generated (on the second level) sequences will be infrequent, too.

At first we check the first level sequences. We have m sequences. After defining their frequencies, we start considering the second level sequences. There will be m^2 of such sequences ($i_1i_1, i_1i_2, \ldots, i_1i_m, i_2i_1, \ldots, i_2i_m, \ldots, i_mi_1, \ldots, i_mi_m$). However, we will not check whole the sequences set. According to the previous level, we will only define which sequences should be checked and which not. If the second level sequence includes an infrequent sequence of the previous level (first level), then it is infrequent and we can eliminate it even without checking it up. Let us analyze an example. Suppose

that some sequence is given:

$$S = ABCCCBBCABCABCABCBABCCCABCAABABCABC \quad (1)$$

Table 1. The first level

Level	Sequence	Frequency	Is it frequent?
1	A	10	+
1	B	13	+
1	C	13	+

Table 2. The second level

Level	Sequence	Shall we check it?	Frequency	Is it frequent?
2	AA	+	1	-
2	AB	+	9	+
2	AC	+	0	-
2	BA	+	2	-
2	BB	+	1	-
2	BC	+	9	+
2	CA	+	6	+
2	CB	+	2	-
2	CC	+	4	+

Table 3. The third level

Level	Sequence	Shall we check it?	Frequency	Is it frequent?
3	ABA	-	-	-
3	ABC	+	8	+
3	ABB	-	-	-
3	BCA	+	5	+
3	BCB	-	-	-
3	BCC	+	2	-
3	CAB	+	5	+
3	CAA	-	-	-
3	CAC	-	-	-
3	CCA	+	1	-
3	CCB	-	-	-
3	CCC	+	2	-

We will say that the sequence is frequent iff it occurs in the text not less than 4 times, i.e. the minimum support is equal to 4. We can see that all the sequences in the first level (Table 1) are frequent. Now we will generate the next level according to these sequences. We have checked all the sequences in the second level (Table 2), because all the previous the level sequences were frequent. Now we will generate next level. We will not check six newly got sequences: *ABA*, *ABB*, *BCB*, *CAA*, *CCB* of the third level (Table 3) since they include infrequent subsequences of the previous level. The reduced amount of checking increases the algorithm efficiency and reduces time input. As the third level does not contain frequent sequences, we will not form the forth level and say that the performance of the algorithm is completed. Frequent sequences will be *A*, *B*, *C*, *AB*, *BC*, *CA*, *CC*, *ABC*, *BCA*, *CAB*.

3 The probabilistic algorithm for mining frequent sequences (ProMFS)

The new algorithm for mining frequent sequences is based on the estimation of the statistical characteristics of the main sequence:

- the probability of element in the sequence,
- the probability for one element to appear after another one,
- the average distance between different elements of the sequence.

The main idea of the algorithm is following:

1. some characteristics of the position and interposition of elements are determined in the main sequence;
2. the new much shorter model sequence $\tilde{C}$ is generated according to these characteristics;
3. the new sequence is analyzed with the GSP algorithm (or any similar one);
4. the subsequences frequency in the main sequence is estimated by the results of the GSP algorithm applied on the new sequence.

Let:

1) $P(i_j) = V(i_j)/(VS)$ be a probability of occurrence of element i_j in the main sequence, where $i_j \in L, j = 1, ..., m$. Here L=$\{i_1, i_2, \ ... \ , i_m\}$ is the set consisting of m distinct elements. $V(i_j)$ is the number of elements i_j in the main sequence S; VS is the length of the sequence.

2) $P(i_j|i_v)$ be the probability of appearance of element i_v after element i_j, where $i_j, i_v \in L, j, v = 1, ..., m$.

3) $D(i_j|i_v)$ be the distance between elements i_j and i_v, where $i_j, i_v \in L, j, v = 1, ..., m$. In other words, the distance $D(i_j|i_v)$ is the number of elements that are between i_j and the first found i_v seeking from i_j to the end

of the main sequence, where $D(i_j|i_v)$ includes i_v. The distance between two neighboring elements of the sequence is equal to one.

4) $\hat{A}$ be the matrix of average distances. Elements of the matrix are as follows: $a_{jv} = Average(D(i_j|i_v), i_j, i_v \in L), j, v = 1, ..., m$. All these characteristics can be obtained during one search through the main sequence. According to these characteristics a much shorter model sequence $\tilde{C}$ is generated. The length of this sequence is l. Denote its elements by c_r, $r = 1, ..., l$. The model sequence $\tilde{C}$ will contain elements from L: $i_j \in L, j = 1, ..., l$. For the elements c_r, a numeric characteristic $Q(i_j, c_r)$, $r = 1, ..., l$, $j = 1, ..., m$, is defined. Initially, $Q(i_j, c_r)$ is the matrix with zero values that are specified after the statistical analysis of the main sequence. A complementary function $q(c_r, a_{rj})$ is introduced. This function increases the value of characteristics $Q(i_j, c_r)$ by one. The first element c_1 of the model sequence $\tilde{C}$ is that from L, that corresponds to $max(P(i_j))$, $i_j \in L$. According to c_1, it is activated the function $q(c_1, a_{1j}) \Longrightarrow Q(i_j, 1 + a_{1j}) = Q(i_j, 1 + a_{1j}) + 1, j = 1, ..., m$. Remaining elements c_r, $r = 2, ..., l$, are chosen in the way below. Consider the r-th element c_r of the model sequence $\tilde{C}$. The decision, which symbol from L should be chosen as c_r, will be made after calculating $max(Q(i_j, c_r))$, $i_j \in L$. If for some p and t we obtain that $Q(i_p, c_r) = Q(i_t, c_r)$, then element c_r is chosen by maximal value of conditional probabilities, i.e. by $max(P(c_{(r-1)}|i_p), P(c_{(r-1)}|i_t))$: $c_r = i_p$ if $P(c_{(r-1)}|i_p) > P(c_{(r-1)}|i_t)$, and $c_r = i_t$ if $P(c_{(r-1)}|i_p) < P(c_{(r-1)}|i_t)$. If these values are equal, i.e. $P(c_{(r-1)}|i_p) = P(c_{(r-1)}|i_t)$, then c_r is chosen in dependence on $max(P(i_p), P(i_t))$. After choosing the value of c_r, the function $q(c_r, a_{rj}) \Longrightarrow Q(i_j, r + a_{rj}) = Q(i_j, r + a_{rj}) + 1$ is activated. All these actions are performed consecutively for every $r = 2, ..., l$. In such way we get the model sequence $\tilde{C}$ which is much shorter than the main one and which may be analyzed by the GSP algorithm with much less computational efforts.

Consider the previous example with the main sequence (1) given in Section 2. $L = \{A, B, C\}$, i.e. m=3, $i_1 = A, i_2 = B, i_3 = C$. The sequence has VS=35 elements.

After one checking of this sequence such probabilistic characteristic are calculated:

$P(A) = 10/35 \approx 0.2857$, $P(B) = 12/35 \approx 0.3429$, $P(C) = 13/35 \approx 0.3714$, $P(A|A) = 0.1$, $P(A|B) = 0.9$, $P(A|C) = 0$, $P(B|A) \approx 0.1667$, $P(B|B) \approx 0.0833$, $P(B|C) \approx 0.7500$, $P(C|A) \approx 0.4615$, $P(C|B) \approx 0.1538$, $P(C|C) \approx 0.3077$.

Table 4. The matrix $\hat{A}$ of average distances

	A	B	C
A	3.58	1.10	2.50
B	2.64	2.91	1.42
C	2.33	2.25	2.67

Let us compose a model sequence $\tilde{C}$, whose length is l=8. At the beginning, the sequence $\tilde{C}$ is empty, and $Q(i_j, c_r) = 0$, $r = 1, ..., l$, $j = 1, ..., m$:

r	1	2	3	4	5	6	7	8
A	0	0	0	0	0	0	0	0
B	0	0	0	0	0	0	0	0
C	0	0	0	0	0	0	0	0
Model sequence $\tilde{C}$	-	-	-	-	-	-	-	-

The first element of $\tilde{C}$ is determined according to the largest probability $P(i_j)$. In our example, it is C, i.e. $c_1 = C$. Recalculate $Q(i_j, c_1)$, $j = 1, 2, 3$, according to the average distances. The situation becomes as follows:

r	1	2	3	4	5	6	7	8
A	0	0	**1**	0	0	0	0	0
B	0	0	**1**	0	0	0	0	0
C	0	0	0	**1**	0	0	0	0
Model sequence $\tilde{C}$	***C***	-	-	-	-	-	-	-

Let us choose c_2. All three values $Q(i_j, c_1)$, $j = 1, 2, 3$, are equal. Moreover, they are equal to zero. Therefore, c_2 will be determined by maximal value of conditional probabilities. $Max(P(C|A),\ P(C|B),\ P(C|C)) = P(C|A) = 0.4615$. Therefore, $c_2 = A$. Recalculate $Q(i_j, c_2)$, $j = 1, 2, 3$, according to the average distances. The situation becomes as follows:

r	1	2	3	4	5	6	7	8
A	0	0	1	0	0	**1**	0	0
B	0	0	**2**	0	0	0	0	0
C	0	0	0	1	**1**	0	0	0
Model sequence $\tilde{C}$	*C*	***A***	-	-	-	-	-	-

Next three steps of forming the model sequence are given below:

r	1	2	3	4	5	6	7	8
A	0	0	1	0	0	**2**	0	0
B	0	0	2	0	0	**1**	0	0
C	0	0	0	1	**2**	0	0	0
Model sequence $\tilde{C}$	*C*	*A*	***B***	-	-	-	-	-

r	1	2	3	4	5	6	7	8
A	0	0	1	0	0	**3**	0	0
B	0	0	2	0	0	**2**	0	0
C	0	0	0	1	2	0	**1**	0
Model sequence $\tilde{C}$	*C*	*A*	*B*	***C***	-	-	-	-

r	1	2	3	4	5	6	7	8
A	0	0	1	0	0	3	**1**	0
B	0	0	2	0	0	2	**1**	0
C	0	0	0	1	2	0	1	**1**
Model sequence $\tilde{C}$	C	A	B	C	$\boldsymbol{C}$	-	-	-

The resulting model sequence is $\tilde{C}$= $CABCCABC$. The GSP algorithm determined that the longest frequent subsequence of the model sequence is ABC when the minimum support is set to two. Moreover, the GSP algorithm has determined the second frequent subsequence CAB of the model sequence for the same minimum support. In the main sequence (1), the frequency of ABC is 8 and that of CAB is 5. However, the subsequence BCA, whose frequency is 5, has not been determined by the analysis of the model sequence. One of the reasons may be that the model sequence is too short.

4 Experimental results

The probabilistic mining of frequent sequences was compared with the GSP algorithm. We have generated the text file of 100000 letters (1000 lines and 100 symbols in one line). $L = \{A, B, C\}$, i.e. m=3, $i_1 = A, i_2 = B, i_3 = C$. In this text we have included one very frequent sequence $ABBC$. This sequence is repeated 20 times in one line. The remaining 20 symbols of the line are selected at random. First of all, the main sequence (100000 symbols) was investigated with the GSP algorithm. The results are presented in Figures 1 and 2. They will be discussed more in detail together with the results of ProMFS. ProMFS generated the following model sequence $\tilde{C}$ of length l=40:

$\tilde{C}$ = $BBCABBCABBCABBCABBCABBCABBCABBCABBCABBCA$

This model sequence was examined with the GSP algorithm using the following minimum support: 8, 9, 10, 11, 12, 13, and 14. The results are presented in Figures 1 and 2. Fig. 1 shows the number of frequent sequences found both by GSP and ProMFS. Fig. 2 illustrates the consumption of computing time used both by GSP and ProMFS to obtain the results of Fig. 1 (the minimum support in ProMFS is Ms=8; the results are similar for larger Ms). The results in Fig. 1 indicate that, if the minimum support in GSP analyzing the main sequence is comparatively small (less than 1500 with the examined data set), GSP finds much more frequent sequences than ProMFS. When the minimum support in GSP grows from 2500 till 6000, the number of frequent sequences by GSP decreases and by ProMFS increases. In the range of [2500, 6000], the number of frequent sequences found both by GSP and ProMFS is rather similar. When the minimum support in GSP continues growing, the number of frequent sequences found by both algorithms becomes identical. When comparing the computing time of both algorithms (see Fig. 2), we can conclude, that the ProMFS operates much faster. In the range of the minimum support in GSP [2500, 6000], ProMFS needs approximately 20

times less of computing time as compared with GSP to obtain the similar result.

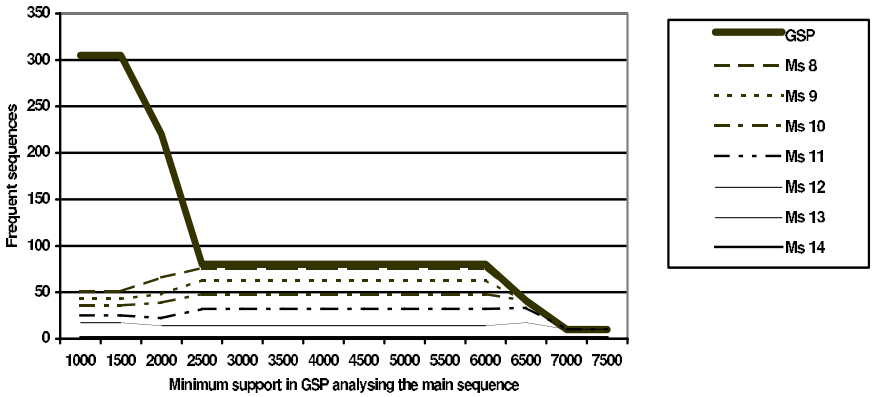


Fig. 1. Number of frequent sequences found both by GSP and ProMFS (minimum support in ProMFS is Ms=8, ..., 14)

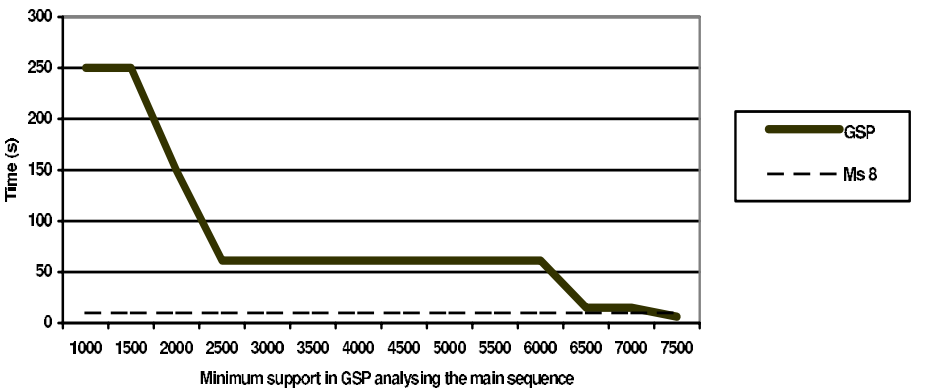


Fig. 2. Computing time used both by GSP and ProMFS (minimum support in ProMFS is Ms=8)

5 Conclusion

The new algorithm ProMFS for mining frequent sequences is proposed. It is based on the estimated probabilistic-statistical characteristics of the appearance of elements of the sequence and their order: the probability of element

in the sequence, the probability for one element to appear after another one, and the average distance between different elements of the sequence. The algorithm builds a new much shorter model sequence and makes decision on the main sequence in accordance on the results of analysis of the shorter one. The model sequence may be analyzed by the GSP or other algorithm for mining frequent sequences: the subsequences frequency in the main sequence is estimated by the results of the model sequence analysis. The experimental investigation indicates that the new algorithm allows to save the computing time in a large extent. It is very important when analyzing very large data sequences. Moreover, the model sequence, that is much shorter than the main one, may be easier understandable and perceived: in the experimental investigation, the sequence of 100000 elements has been modeled by a sequence of 40 elements. However, the sufficient relation between the length of the model sequence and the main sequence needs for a more deep investigation – both theoretical and experimental. In the paper, we present the experimental analysis of the proposed algorithm on the artificial data only. Further research should prove the efficiency on the real data. The research should disclose the optimal values of algorithm parameters (e.g. the length $l$of the model sequence $\tilde{C}$, the minimum support for analysis of the model sequence). Another perspective research direction is the development of additional probabilistic-statistical characteristics of large sequences. This may produce the model sequence that is more adequate to the main sequence.

References

1. Agrawal, R.C., Agrawal, C.C., Prasad, V.V. (2000) Depth first generation of long patterns. In Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Boston, Massachusetts 108-118
2. http://en.wikipedia.org/wiki/DNA_sequence
3. Zaki, M.J. (2001) SPADE: An efficient algorithm for mining frequent sequences. Machine Learning Journal. (Fisher, D. (ed.): Special issue on Unsupervised Learning). **42** (1/2) 31-60
4. Zaki, M.J. (2000) Parallel sequence mining on shared-memory machines. In: Zaki, M.J., Ching-Tien Ho (eds): Large-scale Parallel Data Mining. Lecture Notes in Artificial Intelligence, Vol. **1759**. Springer-Verlag, Berlin Heidelberg, New York 161-189
5. Pei, P.J., Han, J., Wang, W. (2002) Mining Sequential Patterns with Constraints in Large Databases. In Proceedings of the 11th ACM International Conference on Information and Knowledge Management (CIKM'02). McLean, VA 18-25
6. Pinto, P., Han, J., Pei, J., Wang, K., Chen, Q., Dayal, U. (2001) Multi-Dimensional Sequential Pattern Mining. In Proceedings of the 10th ACM International Conference on Information and Knowledge Management (CIKM'01). Atlanta, Georgia, 81-88
7. Pei, J., Han, J., Mortazavi-Asl, B., Pinto, H., Chen, Q., Dayal, U., Hsu, M.-C. (2001) PrefixSpan: Mining sequential patterns efficiently by prefix-projected

pattern growth. In Proc. 17th International Conference on Data Engineering ICDE2001. Heidelberg, 215-224
8. Han, J., Pei, J. (2000)FreeSpan: Frequent pattern-projected sequential pattern mining. In Proc. Knowledge Discovery and Data Mining. 355-359
9. Ayres, J., Flannick, J., Gehrke, J., Yiu, T. (2002) Sequential pattern mining using a bitmap representation. In Proc. Knowledge Discovery and Data Mining. 429-435
10. Kum, H.C., Pei, J., Wang, W. (2003) ApproxMAP: Approximate Mining of Consensus Sequential Patterns. In Proceedings of the 2003 SIAM International Conference on Data Mining (SIAM DM '03). San Francisco, CA, 311-315

PCA and ICA Methods for Prediction Results Enhancement

Ryszard Szupiluk[1,2], Piotr Wojewnik[1,2], and Tomasz Ząbkowski[1]

[1] Polska Telefonia Cyfrowa Sp. z o.o.
Al. Jerozolimskie 181, 02-222 Warsaw, Poland
[2] Warsaw School of Economics
Al. Niepodległości 162, 02-554 Warsaw, Poland

Abstract. In this paper we show that applying of multidimensional decompositions can improve the modelling results. The predictions usually consist of twofold elements, wanted and destructive ones. Rejecting of the destructive components should improve the model. The statistical methods like PCA and ICA with new modifications are employed. The example from the telecom market proofs correctness of the approach.

1 Introduction

Data mining is an automatic process of finding trends and patterns in data to provide us with the previously unknown knowledge usable for business purposes [7,11]. Usually, the techniques are used in problems of fraud detection, client's market segmentation, risk analysis etc. The typical DM process includes testing many models, their comparison, choosing of the best and leaving the rest. In this paper we propose not to waste this ȘrestŤ, but to use it improving all the results. It can be treated as problem of models aggregation. Usually, it is solved by mixing of the parameters or averaging of the models' results [8,16,21]. Our approach concentrates on the decomposition of models results into interesting basis components. In this way we try to find the signals with positive and negative influence on final results. After the destructive signals are rejected and an operation inverse to the previous decomposition is applied [17,18] then the modelling improvement should be achieved. The full methodology can be treated as a postprocessing stage in data mining, involving: decomposition, identification, elimination and composition (DIEC) steps.

Because we are looking for latent components the methodology of blind signal separation (BSS) is suitable here and transformations like independent component analysis (ICA) and principal component analysis (PCA) can be used [9,10]. The DIEC stage can be performed after many models are trained and tested so it is convenient to implement it as a batch type algorithm. Therefore we propose modification of standard ICA Natural Gradient, from online to batch form.

The presented methodology will be applied to the problem of estimating a payment risk of the customer. The high risk clients are usually a small group

in the population what means sparseness of input variables. We propose a new type of nonlinear function in ICA algorithm for such variables.

2 Model Results' Integration

The models try to represent the dependency between input data and target, but their precision is limited [6]. We assume that each model results include two types of components: constructive (good) associated with target and destructive (bad) associated with inaccurate learning data, individual properties of models etc. Many of good and bad components are common to all the models due to the same target, learning data set, similar model structures or optimization methods. Our aim is to explore information given simultaneously by many models to identify and eliminate components with destructive influence on models results.

To analyse the data structure we assume that result of model x_i, $i = 1, \ldots, m$, with N observations, is linear combination of constructive components $t_1, t_2, \ldots, t_p$, and destructive components $v_1, v_2, \ldots, v_q$, what gives

$$x_i = \alpha_{i1} t_1 + \ldots + \alpha_{ip} t_p + \beta_{i1} v_1 + \ldots + \beta_{iq} v_q \quad . \tag{1}$$

In close matrix form we have

$$x_i = \mathbf{a}_i \mathbf{t} + \mathbf{b}_i \mathbf{v} \quad , \tag{2}$$

where $\mathbf{t}=[t_1, \ldots, t_p]^{\mathrm{T}}$ is a $p \times N$ matrix of target components, $\mathbf{v}=[v_1, \ldots, v_q]^{\mathrm{T}}$ is a $q \times N$ matrix of residuals, $\mathbf{a}_i = [\alpha_{i1}, \ldots, \alpha_{ip}]$, $\mathbf{b}_i = [\beta_{i1}, \ldots, \beta_{iq}]$ are vectors of coefficients. In case of many models we have close matrix form

$$\mathbf{x} = \mathbf{A}\mathbf{s} \quad , \tag{3}$$

where $\mathbf{x} = [x_1, x_2, \ldots, x_m]^{\mathrm{T}}$ is a $m \times N$ matrix of model results, $\mathbf{s} = [\mathbf{t}\ \mathbf{v}]^{\mathrm{T}}$ is a $n \times N$ matrix of basis components ($n = p + q$), and $\mathbf{A} = [\mathbf{a}_1 \mathbf{b}_1, \ldots, \mathbf{a}_m \mathbf{b}_m]^{\mathrm{T}}$ is $m \times n$ matrix of mixing coefficients.

Our concept is to identify the source signals and the mixing matrix from the observed models' results $\mathbf{x}$, and reject the common destructive signals, what means replacing $\mathbf{v}$ in $\mathbf{s}$ with zero. After proper identification and rejection we have $\hat{\mathbf{s}} = [\mathbf{t}\ \mathbf{0}]^{\mathrm{T}}$ what allows us to obtain purified target estimation

$$\hat{\mathbf{x}} = \mathbf{A}\hat{\mathbf{s}} = \mathbf{A}[\mathbf{t}\ \mathbf{0}]^{\mathrm{T}} \quad . \tag{4}$$

The main problem is to distinguish $\mathbf{t}$ from $\mathbf{v}$. This task requires some decision system. If we don't have any sophisticated method we can simply check the impact of all $\mathbf{s}$ components on final results. It means the rejecting one by one, sole source signals s_i and the mixing the rest in transformation inverse to decomposition system.

Although we can deal with binary response variables, the standard models like logistic regression, MLP, and RBF, give continuous predictions. In such situation we perform DIEC methodology before the cut off value transforms continues outcome into a binary.

3 Decomposition Methods

From many transformations which can be applied for our task we focus on linear multivariate ones like ICA or PCA. The methods use different features and properties of data but both of them can be considered as looking for a data representation of the form (3). To find the latent variables $\mathbf{A}$ and $\mathbf{s}$ we often can use an transformation defined by matrix $\mathbf{W} \in R^{n \times m}$, such that

$$\mathbf{y} = \mathbf{W}\mathbf{x} \quad , \tag{5}$$

where $\mathbf{y}$ is related to $\mathbf{s}$ and it satisfies specific criteria as e.g. decorrelation or independence. Particular methods of $\mathbf{W}$ estimation are given as follows. For simplicity we assume that $n = m$ and $E\{\mathbf{x}\} = \mathbf{0}$.

3.1 Principal Component Analysis (PCA) and Decorrelation Methods

PCA is a second order statistics method associated with model (1) where the main idea is to obtain orthogonal variables $y_1, \ldots, y_m$ ordered by decreasing variance [10]. To find the transformation matrix $\mathbf{W}$ the eigenvalue decomposition (EVD) of the correlation matrix $\mathbf{R}_{xx} = E\{\mathbf{x}\mathbf{x}^{\mathrm{T}}\}$ can be performed by:

$$\mathbf{R}_{xx} = \mathbf{U}\mathbf{\Sigma}\mathbf{U}^{\mathrm{T}} \quad , \tag{6}$$

where $\mathbf{\Sigma} = \mathrm{diag}[\sigma_1, \ldots, \sigma_m]$ is the diagonal matrix of eigenvalues ordered by decreasing value, $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_m]$ is orthogonal matrix of $\mathbf{R}_{xx}$ eigenvectors related to specific eigenvalues. The transformation matrix can be obtained as

$$\mathbf{W} = \mathbf{U}^{\mathrm{T}} \quad . \tag{7}$$

Table 1. Decorrelation methods

Decorrelation method	Factorization matrix	Transformation matrix
Correlation matrix	$\mathbf{R}_{xx} = \mathbf{R}_{xx}^{\frac{1}{2}}\mathbf{R}_{xx}^{\frac{1}{2}}$	$\mathbf{W} = \mathbf{R}_{xx}^{-\frac{1}{2}}$
LU decomposition	$\mathbf{R}_{xx} = \mathbf{L}\mathbf{U}$	$\mathbf{W} = \mathbf{L}^{-1}$
Cholesky factorization	$\mathbf{R}_{xx} = \mathbf{G}^{\mathrm{T}}\mathbf{G}$	$\mathbf{W} = \mathbf{G}^{-\mathrm{T}}$

Next to PCA there exist many other decorrelation methods giving different set of decorrelated variables [20]. In Table 1 we briefly present selected decorrelation method based on linear transformation (5).

3.2 Independent Component Analysis (ICA)

Independent Component Analysis is a statistical tool, which allows us to decompose observed variable into independent components [3,9]. After ICA decomposition we have got signals (variables) without any linear and nonlinear statistical dependencies. This is the main difference from the standard correlation methods (PCA), which allow us to analyze only the linear dependencies. Form many existing ICA algorithms we focus on Natural Gradient method [1] which online form is as follows:

$$\mathbf{W}(k+1) = \mathbf{W}(k) + \mu(k)[\mathbf{I} - \mathbf{f}(\mathbf{y}(k))\mathbf{g}(\mathbf{y})^{\mathrm{T}}(k)]\mathbf{W}(k) \quad , \tag{8}$$

where $\mathbf{W}(k)$ denotes the transformation matrix in k iteration step, $\mathbf{f}(\mathbf{y}) = [f_1(y_1), \ldots, f_n(y_n)]^{\mathrm{T}}$ and $\mathbf{g}(\mathbf{y}) = [g_1(y_1), \ldots, g_n(y_n)]^{\mathrm{T}}$ are vectors of nonlinearities which can be adaptively chosen according to normalized kurtosis of estimated signals $\kappa_4(y_i) = E\{y_i^4/\}/E^2\{y_i^2/\} - 3$, see Table 2 [3–5].

Table 2. Nonlinearities for Natural Gradient

	$f_i(y_i)$	$g_i(y_i)$
$\kappa_4(y_i) > 0$	$\tanh(\beta_i y_i)$	$\mathrm{sign}(y_i)\lvert y_i\rvert^{r_i}$
$\kappa_4(y_i) < 0$	$\mathrm{sign}(y_i)\lvert y_i\rvert^{r_i}$	$\tanh(\beta_i y_i)$
$\kappa_4(y_i) = 0$	y_i	y_i

where $r_i > 0$, β_i are constant

The main problem with the choice of nonlinearities based on kurtosis is fact that the statistics not always exist. Many signals and variables can be impulsive or sparse and their distributions have not higher order moments as e.g. the family of α–stable distributions, which do not posses statistics of order higher than second [14,15]. We propose nonlinear function which allows us to analyse such distributions [17].

One of the main problems in ICA connected with the α-stable distributions is the lack of the close form for their probability density function (pdf) with only small number of exceptions like Cauchy, Pareto or Gaussian [12,13]. In general, the distribution is given by characteristic function of the form [14]:

$$E(e^{itZ}) = \begin{cases} \exp\{-\sigma^\alpha |t|^\alpha (1 - i\beta \cdot \mathrm{sign}(t) \cdot \tan\frac{\pi\alpha}{2}) + i\mu t\} & \text{if } \alpha \neq 1 \\ \exp\{-\sigma|t|(1 + \frac{2i\beta}{\pi} \cdot \mathrm{sign}(t) \cdot \ln|t|) + i\mu t\} & \text{if } \alpha = 1 \end{cases} , \tag{9}$$

where $0 < \alpha \leq 2$ describes ṢthicknessT̆ of the tails, $\sigma > 0$ dispersion, $-1 \leq \beta \leq 1$ symmetry and μ location. The α parameter is crucial for description of the α–stable distributions, because for given distribution there are no moments higher than α. Using expansion of the pdf's into series we can estimate the shape of the nonlinearities for functions with various α. Based on numerical simulation we propose a general form of nonlinearities for α–stable distributions as:

$$f_i(y_i) = \frac{-2 \cdot y_i}{\theta \cdot \sigma^2 + y_i^2 \cdot (2 - \alpha)} \quad , \tag{10}$$

where σ is parameter of dispersion and typically $\theta = \alpha$. The crucial thing in (10) is proper alpha parameter estimation. For Symmetric Alpha Stable (SαS) distributions it can be computed from

$$\sigma_z^2 = \frac{\pi^2}{6}(\alpha - x^{-2} - 0.5) \quad , \tag{11}$$

where σ_z^2 is second moment of $z = \ln(|x|)$. It should be noted that there exist the family of α–stable distributions, called sub-gaussian α–stable distributions that are always mutually dependent [14], so (10) is not addressed to such task.

Batch type ICA algorithm

The standard Natural Gradient algorithm is strongly recommended for non–stationary environments due to its online form. But there are several disadvantages associated with such algorithm type as high sensitivity to learning rate choice or online distributions estimation. To avoid the above difficulties we propose the more convenient form of the algorithm, which is the batch type, so we modify (8). We take expected value of (8) and use the fact that after adequate learning we have $\mathbf{W}(j+1) = \mathbf{W}(j)$. Therefore assuming $\mu(k) = 1$ we have

$$\mathbf{R}_{fg} = E\{\mathbf{f}(\mathbf{y})\mathbf{g}(\mathbf{y})^{\mathrm{T}}\} = E\{\mathbf{f}(\mathbf{W}\mathbf{x})\mathbf{W}\mathbf{x}^{\mathrm{T}}\} = \mathbf{I} \quad , \tag{12}$$

what is a condition of proper $\mathbf{W}$ estimation. The full batch algorithm is as follows:

1. Initialize $\mathbf{W} = \mathbf{W}_0$ and $\mathbf{y} = \mathbf{W}\mathbf{x}$.
2. Perform $\mathbf{R}_{fg} = E\{\mathbf{f}(\mathbf{y})\mathbf{g}(\mathbf{y})^{\mathrm{T}}\}$ and than obtain symmetric matrix by

$$\bar{\mathbf{R}}_{fg} = \frac{1}{2}[\bar{\mathbf{R}}_{fg} + \bar{\mathbf{R}}_{fg}^{\mathrm{T}}] \quad .$$

3. Use the eigenvalue decomposition $\bar{\mathbf{R}}_{fg} = \mathbf{U}\mathbf{\Sigma}\mathbf{U}^{\mathrm{T}}$.
4. Compute $\mathbf{Q} = \mathbf{\Sigma}^{-1/2}\mathbf{U}^{\mathrm{T}}$ and next $\mathbf{z} = \mathbf{Q}\mathbf{y}$.
5. Substitute $\mathbf{Q} \leftarrow \mathbf{Q}\mathbf{W}$, $\mathbf{y} \leftarrow \mathbf{z}$ and repeat steps 2–4 .

To improve efficiency of ICA algorithms the data preprocessing such as decorrelation can be performed [3]. A typical number of iterations for the batch algorithm is about 50–100.

4 Practical Experiment

The methodology was applied to estimate the payment risk of the telecom customers. The problem is to predict whether the client will pay the invoice. The models use six variables: trade code of the customer, activation date, total payment, total payments within last 12 months, the number of open invoices and the open amount from the last invoices. Five neural network models with MLP6-2-1, MLP6-2-2-1, MLP6-4-1, MLP6-18-1, RBF6-12-1 structures where chosen for testing the approach described in this paper. The learning set included 120 000 observations, the validating one 90 000, and the testing one 90 000. In DIEC stage the PCA and ICA decompositions were performed.

In this specific application we are mainly interested in two measures which are the Area under a Receiver Operating Characteristic (ROC) Curve (AUC) and the number of risky clients (*Bad clients*) in the first two thousand of the worst clients. ROC is constructed by plotting a series of pairs of true positive rate and false positive rate calculated from varying cuts of positivity escalated by given increments in predicted probability [2]. The number of risky clients (*Bad clients*) in the first two thousand of the worst cases is taken due to business objectives. By inspecting AUC and *Bad clients* before and after decompositions we can discover that, rejecting of the specific component, improved the prediction quality of all the models.

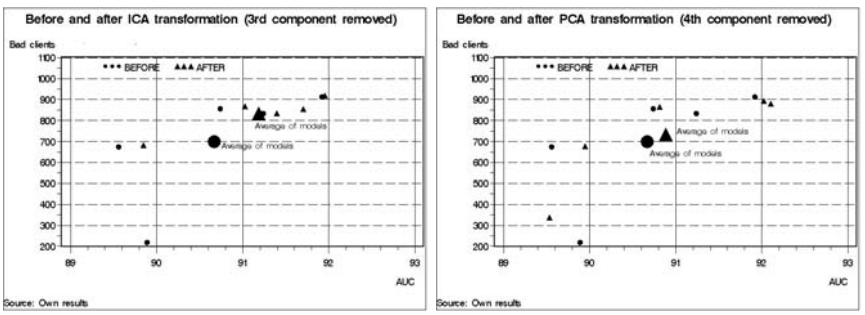


Fig. 1. Results after ICA and PCA transformations

The graphs in Fig. 1 present the improved results after applying specific decomposition and removing destructive component. The dots and triangles denote the models. The biggest figures are the models' results averaging.

Average accuracy of models for risky and good client in predicted and real terms is shown in Fig. 2. There were about 35% of bad, risky clients

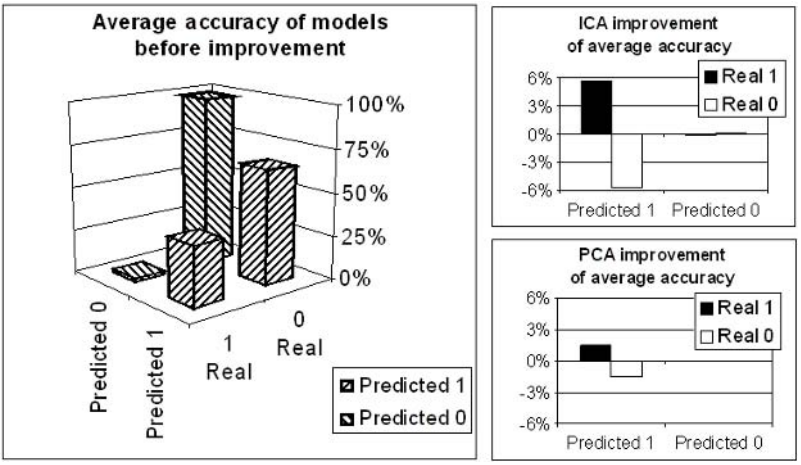


Fig. 2. Average models' accuracy and its improvement

among those predicted as the bad ones. On the right we can observe the prediction improvements. ICA algorithm in this case improves the accuracy of bad clients prediction by 6% and by the same percent reduces the error of predicting good clients as the bad ones. PCA lets us to improve the models by 1,5% on average.

5 Conclusions

In this article we present a new concept of many models results utilizing. The methodology called DIEC allows us to eliminate common destructive components from model results. It helps to improve the modelling accuracy and to enhance generalization abilities due to combining the constructive components common to all the models.

The investigation of common negative components can give us information about using improper factors in data mining process. The destructive components can be present due to inadequate optimizations methods, bad learning data, improper variable selection etc. Analysis of those questions gives us many issues for future research. The other question is what happens, if it is not possible to check the assumption of the linear dependency between models' results and source signals. We can still use the presented approach, but the interpretation of the estimated signals is not straightforward.

References

1. Amari S., Cichocki A., and Yang H.H. (1996) A new learning algorithm for blind signal separation. Advances in Neural Information Processing Systems, NIPS–1995 **8**, MIT Press, Cambridge, MA, 757–763

2. Bradley A.P. (1997) The use of the area under the ROC curve in the evaluation of machine learning algorithms. Pattern Recognition **30**(7), 1145–1159
3. Cichocki A., Amari S. (2002) Adaptive Blind Signal and Image Processing. John Wiley, Chichester
4. Cichocki A., Sabala I., Choi S., Orsier B., Szupiluk R. (1997) Self adaptive independent component analysis for sub–Gaussian and super–Gaussian mixtures with unknown number of sources and additive noise. NOLTA-97 **2**, Hawaii, USA, 731–734
5. Cruces S., Cichocki A., Castedo L. (1999) An iterative inversion method for blind source separation. Proc. of the 1st Int. Workshop on Independent Component Analysis and Blind Signal Separation (ICA'99), Assois, France, 307–312
6. Greene W.H. (2000) Econometric analysis, NJ Prentice Hall
7. Groth R., Data Mining. Building Competitive Advantage, Prentice Hall Inc., Upper Saddle River, New Jersey
8. Hoeting J., Madigan D., Raftery A., and Volinsky C. (1999) Bayesian model averaging: a tutorial. Statistical Science **14**, 382–417
9. Hyvärinen A., Karhunen J., Oja E. (2001) Independent Component Analysis. John Wiley
10. Jolliffe I.T. (2002) Principal Component Analysis. Springer Verlag
11. Kennedy R.L. (ed.), Lee Y., van Roy B., Reed C., and Lippman R.P. (1997) Solving Data Mining Problems with Pattern Recognition. Prentice Hall
12. Kidmose P. (2000) Alpha–Stable Distributions in Signal Processing of Audio Signals. 41st Conf. on Simulation and Modeling, Scandinavian Simulation Society SIMS, 87–94
13. Kidmose P. (2001) Independent Component Analysis Using the Spectral Measure for Alpha-Stable Distributions. Proc. of IEEE–EURASIP Workshop on Nonlinear Signal and Image Processing NSIP
14. Nikias C.L., Shao M. (1995) Signal Processing with Alpha–Stable Distributions and Applications. John Wiley and Sons
15. Samorodnitskij G., Taqqu M.S. (1994) Stable non–Gaussian random processes: stochastic models with infinitive variance. Chapman and Hall, New York
16. Schwarz G. (1978) Estimating the Dimension of a Model. The Annals of Statistics **6**, 461–471
17. Szupiluk R., Wojewnik P., Zźbkowski T. (2004) Independent Component Analysis for Filtration in Data Mining. Proc. of IIPWM'04 Conf., Zakopane, Poland. Advances in Soft Computing, Springer Verlag, Berlin, 117–128
18. Szupiluk R., Wojewnik P., Zźbkowski T. (2004) Model Improvement by the Statistical Decomposition. Artificial Intelligence and Soft Computing–ICAISC 2004. Lecture Notes in Computer Science, Springer-Verlag, Heidelberg, 1199–1204
19. Szupiluk R., Wojewnik P., Zźbkowski T., Siwek K. (2004) Independent Component Analysis with α–Stable Distributions. VI International Workshop Computational Problems of Electrical Engineering, Zakopane, Poland
20. Therrien C.W. (1992) Discrete Random Signals and Statistical Signal Processing. Prentice Hall, New Jersey
21. Yang Y. (2001) Adaptive regression by mixing. Journal of American Statistical Association **96**, 574–588

Creating Reliable Database for Experiments on Extracting Emotions from Music

Alicja Wieczorkowska[1], Piotr Synak[1], Rory Lewis[2], and Zbigniew Ras[2]

[1] Polish-Japanese Institute of Information Technology,
Koszykowa 86, 02-008 Warsaw, Poland
[2] University of North Carolina, Charlotte, Computer Science Dept.,
9201 University City Blvd., Charlotte, NC 28223, USA

Abstract. Emotions can be expressed in various ways. Music is one of possible media to express emotions. However, perception of music depends on many aspects and is very subjective. This paper focuses on collecting and labelling data for further experiments on discovering emotions in music audio files. The database of more than 300 songs was created and the data were labelled with adjectives. The whole collection represents 13 more detailed or 6 more general classes, covering diverse moods, feelings and emotions expressed in the gathered music pieces.

1 Introduction

It is known that listeners respond emotionally to music [12], and that music may intensify and change emotional states [9]. One can discuss if feelings experienced in relation to music are actual emotional states, since in general psychology, emotions are currently described as specific process-oriented response behaviours, i.e. directed at something (circumstance, person, etc.). Thus, musical emotions are difficult to define, and the term "emotion" in the context of music listening is actually still undefined. Moreover, the intensity of such emotion is difficult to evaluate, especially that musical context frequently misses influence of real life, inducing emotions. However, music can be experienced as frightening or threatening, even if the user has control over it and can, for instance, turn the music off.

Emotions can be characterized in appraisal and arousal components, as shown in Figure 1 [11]. Intense emotions are accompanied by increased levels of physiological arousal [8]. Music induced emotions are sometimes described as mood states, or feelings. Some elements of music, such as change of melodic line or rhythm, create tensions to a certain climax, and expectations about the future development of the music. Interruptions of expectations induce arousal. If the expectations are fulfilled, then the emotional release and relaxation upon resolution is proportional to the build-up of suspense of tension, especially for non-musician listener. Trained listener usually prefer more complex music.

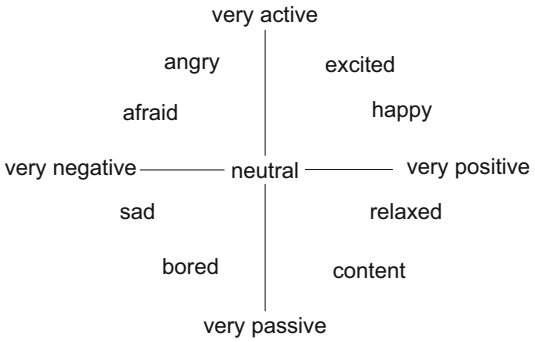


Fig. 1. Examples of emotions in arousal vs. appraisal plane. Arousal values range from very passive to very active, appraisal values range from very negative to very positive

2 Data Labelling

Data labelling with information on emotional contents of music files can be performed in various ways. One of the possibilities is to use adjectives, and if the data are grouped into classes, a set of adjectives can be used to label a single class. For instance, Hevner in [2] proposed a circle of adjective, containing 8 groups of adjectives. Her proposition was later redefined by various researchers, see for instance [5]. 8 categories of emotions may describe: fear, anger, joy, sadness, surprise, acceptance, disgust, and expectancy.

Other way of labelling data with emotions is represent emotions in 2 or 3-dimensional space. 2-dimensional space may describe amount of activation and quality, or arousal and valence (pleasure) [6],[11], as mentioned in Section 1. 3-dimensional space considers 3 categories, for instance: pleasure (evaluation), arousal, and domination (power). Arousal describes the intensity of emotion, ranging from passive to active. Pleasure describes how pleasant is the perceived feeling and it ranges from negative to positive values. Power relates to the sense of control over the emotion. Examples of emotions in 3-dimensional space can be observed in Figure 2.

In our research, we decided to use adjective-based labelling. The following basic labelling was chosen, yielding 13 classes, after Li and Ogihara [5]:

- cheerful, gay, happy,
- fanciful, light,
- delicate, graceful,
- dreamy, leisurely,
- longing, pathetic,
- dark, depressing,
- sacred, spiritual,
- dramatic, emphatic,

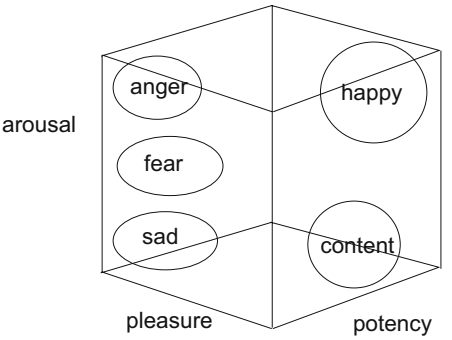


Fig. 2. Emotions represented in 3-dimensional space

- agitated, exciting,
- frustrated,
- mysterious, spooky,
- passionate,
- bluesy.

Since some of these emotions are very close, we also used more general labelling, which gathers emotions into 6 classes [5], as presented in Table 1.

Table 1. Emotions gathered into 6 classes

Class Number	*Class Name*	*Number of Objects*
1	*happy*, *fanciful*	57
2	*graceful*, *dreamy*	34
3	*pathetic*, *passionate*	49
4	*dramatic*, *agitated*, *frustrated*	117
5	*sacred*, *spooky*	23
6	*dark*, *bluesy*	23

3 Collection of Music Data

Gathering the data for such experiment is a time-consuming task, since it requires evaluating a huge amount of songs/music pieces. While collecting data, attention was paid to features of music, which are specific for a given class. Altogether, 303 songs were collected and digitally recorded - initially in

MP3 format, then converted into .snd format for parameterization purposes. For the database the entire songs were recorded.

The main problem in collecting the data was deciding how to classify the songs. The classification was made taking into account various musical features. Harmony is one of such factors. From personal experience (R. Lewis), it is known that a 9th going back to a major compared to a 5th going back to a major chord will make the difference between pathetic and dark. Pathetic is, in one view, the sense one gets when the cowboy loses his dog, wife and home when she leaves him for another (all on 7ths and 9ths for dissonance, and then we go back to a major right at the chorus and there is a sense of relief, almost light hearted relief from this gloomy picture the music has painted). In other view, pathetic is when our army starts an important battle, flags are slating and bravery is in the air (like in Wagner's Walkirie). Dark - is Mozart's Requiem - continuous, drawn out, almost never ending diminishing or going away from the major and almost never going back to the major let alone going even "more major" by augmenting.

Certain scales are known for having a more reliable guarantee to invoke emotions in human being. The most guaranteed scale to evoke emotion is the Blues scale which is a Pentatonic with accentuated flattened III and VIIth. Next comes the minor Pentatonic which is known for being "darker". The Major Pentatonic has "lighter" more "bright" sound and is typically utilized in lighter country, rock or jazz. Other scales such as Mixolydian an Ionian and so forth would diverge into other groups but are not as definitive in extracting a precise emotion from a human being.

There are Minor Pentatonics used primarily in rock and then Major Pentatonic used primarily in Country - which is sweeter. "Penta" means five, but the reason it has six notes is because we also add the lowered 5th of the scale as a passing tone making this a six note scale.

The "Blues scale" is really a slang name the Pentatonic Minor scale that accentuates its flattened III and VIIth's. For instance, when a common musician plays with a trained pianist who is not familiar with common folk slang, if the musician wanted the trained pianist to play with the band while it played a bluesy emotional song, one could simply tell the to play in C6th (notes C E G A C E) over Blues chord progression in the key of A" [root / b3 / 4th / b5th / 5th / b7 back to root) The aforementioned will make any audience from Vermont to Miami, South Africa to Hawaii feel bluesy. But why? What is in this mathematical correlation between the root wave and the flatted fifths and sevenths that guarantees this emotion of sadness?

But getting back to the Pentatonic. It is the staple jazz, Blues, country and bluegrass music. The two different Pentatonic scales are major Pentatonic R - 2 - 3 - 5 - 6 which goes great over major chords. The minor Pentatonic is R - b3 - 4 - 5 - b7 and works well for chord progressions based on minor chords. Now the b3 is where we can bend the emotions of the crowd, it separates country and "nice" music to Blues, Metal and so forth because it sounds horribly out

of place over a major chord. So, we avoid this by playing the b3 with a bend or slide into the 3rd before going to the root - that is Blues. But the twangy country sound uses the major Pentatonic and it keeps returning to the tonic note. The sound that makes the twang sound is produced by bending the second interval. When a person like Stevie Ray Vaughn, or B.B. King leans back so overcome with emotion he is really simply playing these five notes, with that b3 and sometimes the b7 and creating pleasing improvisations over this "Blues scale". Almost all the notes will sound good with almost any basic Blues tune, in a major or minor key so long as the scale is used with the same root name as the key you are playing in.

Another issue to consider when collecting the song was copyright. The copyright issue is a two part test:

1. Did the original means of obtaining the media comply with copyright law? and
2. Is it being used for personal use, or conversely for financial means and/or pier to pier use, i.e. giving away or selling the media without the owner's consent?

In our case, the original music was bought by one of the authors, R. Lewis, through CD's in the store or/and from iTunes. The authors are not selling neither giving the music to others. The authors went to great lengths with UNCC security and legal to make sure that it was password protected. Regarding length, in the past it used to be seven consecutive notes. Recently, a Federal Court in the US issued a ruling that stated that if a jury believes it was stolen off then regardless of the length.

Therefore, our data collection was prepared respecting the copyright law.

4 Features for Audio Data Description

Since the audio data itself are not useful for direct training of a classifier, parameterization of audio data is needed, possibly yielding reasonably limited feature vector. Since the research on automatic recognition of emotions in music signal has started quite recently [5], [13], there is no well established set of features for such a purpose. We decided to use features describing timbre of sound and the structure of harmony. To start with, we apply such parameterization to a signal frame of 32768 samples, for 44100 Hz sampling frequency. The frame is taken after 30 second from the beginning of the recording. The recordings are stored in MP3 format, but for parameterization purposes they are converted to .snd format. The feature vector, calculated for every song in the database, consists of the following 29 parameters [14], [15]:

- *Freq*: dominating pitch in the audio frame
- *Level*: maximal level of sound in the frame

- $Trist1, 2, 3$: Tristimulus parameters for $Freq$, calculated as [7]:

$$Trist1 = \frac{A_1^2}{\sum_{n=1}^{N} A_n^2} \quad Trist2 = \frac{\sum_{n=2,3,4} A_n^2}{\sum_{n=1}^{N} A_n^2} \quad Trist3 = \frac{\sum_{n=5}^{N} A_n^2}{\sum_{n=1}^{N} A_n^2} \quad (1)$$

  where A_n - amplitude of n^{th} harmonic, N - number of harmonics available in spectrum, $M = \lfloor N/2 \rfloor$ and $L = \lfloor N/2 + 1 \rfloor$
- $EvenH$ and $OddH$: Contents of even and odd harmonics in the spectrum, defined as

$$EvenH = \frac{\sqrt{\sum_{k=1}^{M} A_{2k}^2}}{\sqrt{\sum_{n=1}^{N} A_n^2}} \qquad OddH = \frac{\sqrt{\sum_{k=2}^{L} A_{2k-1}^2}}{\sqrt{\sum_{n=1}^{N} A_n^2}} \quad (2)$$

- $Bright$: brightness of sound,. i.e. gravity center of the spectrum, calculated as follows:

$$Bright = \frac{\sum_{n=1}^{N} n\, A_n}{\sum_{n=1}^{N} A_n} \quad (3)$$

- $Irreg$: irregularity of spectrum, defined as [4], [1]

$$Irreg = \log\left(20 \sum_{k=2}^{N-1} \left| \log \frac{A_k}{\sqrt[3]{A_{k-1} A_k A_{k+1}}} \right|\right) \quad (4)$$

- $Freq1, Ratio1, ..., 9$: for these parameters, 10 most prominent peaks in the spectrum are found. The lowest frequency within this set is chosen as $Freq1$, and proportions of other frequencies to the lowest one are denoted as
  $Ratio1, ..., 9$
- $Ampl1, Ratio1, ..., 9$: the amplitude of $Freq1$ in decibel scale, and differences in decibels between peaks corresponding to $Ratio1, ..., 9$ and $Ampl1$.

5 Usefulness of the Data Set: Classification Experiments

We decided to check usefulness of the obtained data set in experiments with automatic classification of emotions in music. K-NN (k nearest neighbors) algorithm was chosen for these tests. In k-NN the class for a tested sample is assigned on the basis of the distances between the vector of parameters for this sample and the majority of k nearest vectors representing known samples. CV-5 standard cross-validation was applied in tests, i.e. 20% of data were removed from the set for training and afterwards used for testing; such an experiment was repeated 5 times. In order to compare results with Li and Ogihara [5], experiments were performed for each class separately, i.e. in each classification experiment, a single class was detected versus all

other classes. The correctness ranged from 62.67% for class no. 4 (dramatic, agitated and frustrated) to 92.33% for classes 5 (sacred, spooky) and 6 (dark, bluesy). Therefore, although our database still needs enlargement, it initially proved its usefulness in these simple experiments.

6 Summary

Although emotions induced by music may depend on cultural background and other contexts, there are still feelings commonly shared by all listeners. Thus, it is possible to label music data with adjectives corresponding to various emotions. The main topic of this paper was preparing a labelled database of music pieces for research on automatic extraction emotions from music.

Gathering of data is not only a time-consuming task. It also requires finding the reasonable number of music pieces representing all classes chosen, i.e. emotions. Labelling is more challenging, and in our research it was performed by a musician (R. Lewis). However, one can always discuss whether other subjects would perceive the same emotions for these same music examples. We plan to continue our experiments, expanding the data set and labelling it by more subjects.

The data we collected contain a few dozens of examples for each of 13 classes, labelled with adjectives. One, two, or three adjectives were used for each class, since such labelling may be more informative for some subjects. The final collection consists of more than 300 pieces (whole songs or other pieces). These audio data were parameterized, and feature vectors calculated for each piece constitute a database, used next in experiments on automatic classification of emotions using k-NN algorithm. Therefore, our work yielded a measurable outcomes thus proving its usefulness.

References

1. Fujinaga, I., McMillan, K. (2000) Realtime recognition of orchestral instruments. Proceedings of the International Computer Music Conference, 141–143
2. Hevner, K. (1936) Experimental studies of the elements of expression in music. American Journal of Psychology **48**, 246–268
3. Jackson, W. H. (1998) Cross-Cultural Perception and Structure of Music. On-line, available at http://internet.cybermesa.com/ bjackson/Papers/xc-music.htm
4. Kostek, B, Wieczorkowska, A. (1997) Parametric Representation Of Musical Sounds. Archives of Acoustics **22, 1**, 3–26
5. Li, T., Ogihara, M. (2003) Detecting emotion in music, in *4th International Conference on Music Information Retrieval ISMIR 2003*, Washington, D.C., and Baltimore, Maryland. Available at http://ismir2003.ismir.net/papers/Li.PDF

6. Marasek, K. (2004) Private communication
7. Pollard, H. F., Jansson, E. V. (1982) A Tristimulus Method for the Specification of Musical Timbre. Acustica **51**, 162–171
8. Rickard, N. S. (2004) Intense emotional responses to music: a test of the physiological arousal hypothesis. Psychology of Music **32, (4)**, 371–388. Available at http://pom.sagepub.com/cgi/reprint/32/4/371
9. Sloboda, J. (1996) Music and the Emotions. British Association Festival of Science, The Psychology of Music
10. Smith, H., Ike, S. (2004) Are Emotions Cros-Culturally Intersubjective? A Japanese Test. 21 Centure COE "Cultural and Ecological Foundations of the Mind", Hokkaido University. The Internet: http://lynx.let.hokudai.ac.jp/COE21/
11. Tato, R., Santos, R., Kompe, R., and Pardo, J. M. (2002) Emotional Space Improves Emotion Recognition. *7th International Conference on Spoken Language Processing ICSLP 2002*, Denver, Colorado, available at http://lorien.die.upm.es/partners/sony/ICSLP2002.PDF
12. Vink, A. (2001) Music and Emotion. Living apart together: a relationship between music psychology and music therapy. Nordic Journal of Music Therapy, **10(2)**, 144–158
13. Wieczorkowska, A. (2004) Towards Extracting Emotions from Music. International Workshop on Intelligent Media Technology for Communicative Intelligence, Warsaw, Poland, PJIIT - Publishing House, 181–183
14. Wieczorkowska, A., Wroblewski, J., Slezak, D., and Synak, P. (2003) Application of temporal descriptors to musical instrument sound recognition. Journal of Intelligent Information Systems **21(1)**, Kluwer, 71–93
15. Wieczorkowska, A., Synak, P., Lewis, R., and Ras, Z. (2005) Extracting Emotions from Music Data. 15th International Symposium on Methodologies for Intelligent Systems ISMIS 2005, Saratoga Sprins, NY, USA

Part VI

Poster Session

You Must Be Cautious While Looking For Patterns With Multiresponse Questionnaire Data

Guillermo Bali Ch.[1], Dariusz Czerski[2], Mieczysław Kłopotek[2], Andrzej Matuszewski[2]

[1] Monterrey Institute of Technology, Mexico City Campus, gbali@itesm.mx
[2] Institute of Computer Science, Polish Academy of Sciences,
ul. Ordona 21, 01-237 Warszawa, Poland {amat,klopotek}@ipipan.waw.pl

Abstract. The problem of testing statistical hypotheses of independence of two multiresponse variables is considered. This is a specific inferential environment to analyze certain patterns particularly for the questionnaire data. Data analyst normally looks for certain combination of responses being more frequently chosen than the other ones. As a result of experimental study we formulate some practical advices and suggest areas of further research.
Keywords: "pick any" questions, statistical dependency of database fields, statistical bootstrap, questionnaire surveys, χ^2 distribution, contingency table

1 Introduction

Modern society is requested more and more frequently to express its opinions in terms of questionnaires accompanying various opinion polls. Multiresponse variables are present or even dominating in those surveys. For an analyst, not only the responses to particular questions, but relationships among them are usually of highest interest. The presence of absence of dependence in responses to various questions may be crucial for different decision making processes on the side of the surveying institution. Issues like "is the sample really representative?", "Are the responders really honest?" or "Can I influence the population's behavior on one issue acting on another?" require finding out if such dependencies exist between carefully designed questions.

In this (and the parallel [9]) paper we want to draw attention to the fact that analysis of dependence/independence between multiresponse questions is not as simple as in case of single-response questions. In fact, depending on the assumed model of independence the questions of presence or absence of correlations may have different answers for a given survey and the analyst must be particularly careful when designing a questionnaire with multiresponse questions.

Assume a questionnaire was developed to survey certain population from a point of view that is relevant for a practical purpose. A specific sample of the appropriate population was surveyed with this questionnaire. We will call this sample the "real" one assuming that it is "representative" for the population.

Representativity of the sample has many practical aspects from one hand but it can be partially formalized (it follows from our considerations) within the statistical bootstrap methodology from the other. This general methodology will be developed for our specific problem and can be viewed as a mechanism the "equivalent" samples are "generated" from population.

We consider a specific: "correlation" vs. "independence" problem connected with the data analysis of questionnaire answers. This problem has many aspects that may or have to be considered. We propose a set of those aspects that seems to finally enable a specific, meaningful inference.

"Correlation" applies to the pair of questions: **A** and **B** that propose A and B options respectively. Each responder in the sample (equivalent to given record in the dataset) could pick any subset of options (including empty subset) for both questions. This kind of questions we propose to call the simple multi-response.

2 Examples of datasets

A questionnaire survey was performed in 2004 with the aim to recognize the patterns of energy consumption by households at the countryside. A sample of 200 households is representative (from several points of view) for a certain region of Poland.

We will consider here two multiresponse questions that allow 4 options each i.e. $A = B = 4$.

- **A** – What material is used for heating?
  - a1 – coal
  - a2 – oil
  - a3 – gas LPG
  - a4 – timber and/or similar
- **B** – Type of income of members of household
  - b1 – agriculture
  - b2 – employee
  - b3 – employer
  - b4 – pension

Two subsets of database will be taken for computations, that are available on the Web page `http://www.ipipan.waw.pl/~amat/multiresponse/pl/`.

1. Records corresponding to households that consists of 2 persons.
2. Records corresponding to households that consists of 5 persons.

3 Models, statistics and direct p-values

In this section we list the notions that seems to be relevant for the dependence vs. independence measurement procedure. It should be pointed out, however, that the list is connected with a methodology that covers only a certain aspect of a more general problem. It is not clear enough how to formulate this more general problem since it is not sufficiently researched. Temporary names that are still not adequate are as follows.

- Pattern search corresponding to **A** and **B** (multiresponse) options
- Mining significant combinations of **A** and **B** answers.

Let's restrict ourselves to a specific - dependence-independence hypothesis testing between two multiresponse - framework. Theoretical considerations that make a basis for the following list are presented in [9]. Particularly a definition of bootstrap P-value – called direct P-value – is formulated in that paper.

We use a generalized statistical bootstrap approach. Adjective "generalized" is used here to stress the fact that models we use can be applied without performing the standard resampling procedure, too.

First well-founded statistic that is oriented for a pair of multiresponse (earlier one component of the pair was assumed to a be single-response question) was proposed by Thomas and Decady [11]. They used an important approach formulated for more general purposes (see e.g.: [10,6]). The statistic will be called: Thomas-Decady.

Another approach that appeared in paper of Agresti and Liu [1] and then advocated in papers [4,5] led to two statistics (among others), which will be called:

- mainstream – a "natural" chi-square type sum,
- symmetric max – maximum of components of the mainstream sum.

3 other statistics that will be called:

- traditional chi sq,
- democratic chi sq,
- non-symmetric max,

follow from other line of research. The approach, which had its first mathematical formulation in paper [8] is oriented for further generalizations. The point is that simple multiresponse must be treated as a particular case of hierarchical multiresponse and mathematical formalism should take it into account. The question **A** above asks for heating material. In certain household assume they use coal (a1) and wood (a4) e.g.. With the simple multiresponse question you obtain this information but not the fact that in this specific household they actually obtain 90% of the useful energy from coal and only 10% from wood. With question that allows hierarchy of options, information obtained would be deeper.

More rationale for the above 3 statistics and the following bootstrap sampling schemes:

Super Marginal (SM), Exhaustive Marginal (EM), Restricted Marginal (RM) are given in: [3] and [9].

A version of our experimental program is available at: `http://www.ipipan.waw.pl/~amat/multiresponse/pl/`.

Two basic options of the program allow computations similar to those presented below and in [9] respectively. Besides the classical statistical bootstrap calculation program offers exact calculation of direct P-values (for more information on this aspect see [2]. The algorithm for exact calculation is very complex computationally, even for very moderate sample sizes, however.

In the Internet location mentioned one can find also all our significant publications.

4 Results of computation

3 models that are used for the experiment presented here, as a consequence of their "philosophy", differ in number of parameters. Generally RM is the "richest" in this sense since it considers subtables that are taken only partially into account by EM. SM is still more simple. Clearly RM loses its parametric advantage if subtables in the sample (i.e. in the dataset) are empty. E.g. if there no household with 3 kinds of heating materials and 3 sources of income then subtable (3,3) does not exist and disappear parameters of this table.

We used classical statistical bootstrap calculations with number of resamplings equal to 40 000. This gives the standard error, of the entries in tables presented below, at level less than 0,003.

The characteristics of the first dataset are visible in table 1. Among 32 probabilities (16 for each of both questions) that are defined by model EM, only 11 are greater than zero. There are 5 subtables (of RM) with non-zero frequencies but 3 of them have only 1 household.

Table 1. 2-person household – direct P-values for different statistics and models

	Super Marginal	Exhaustive Marginal	Restricted Marginal
traditional chi sq	0,009	0,067	0,201
democratic chi sq	0,005	0,081	0,137
Thomas-Decady	0,062	0,078	0,135
symmetric max	0,014	0,038	0,169
non-symmetric max	0,082	0,213	0,311
mainstream	0,015	0,037	0,175

Among 32 probabilities that define EM model for the second dataset (see table 2), 17 are greater than zero.

There are 4 subtables in RM with non-zero frequencies. For combination (2,2) e.g. we have the following marginal probabilities (p's correspond to **A** while q's to **B**): p1100 = 0 p1010 = 0 p0110 = 0,14 p1001 = 0,43 p0101 = 0 p0011 = 0,43 q1100 = 0,43 q1010 = 0,14 q0110 = 0,14 q1001 = 0,14 q0101 = 0,14 q0011 = 0

These probabilities are among the RM parameters.

Table 2. 5-person household – direct P-values for different statistics and models

	Super Marginal	Exhaustive Marginal	Restricted Marginal
traditional chi sq	0,069	0,233	0,293
democratic chi sq	0,253	0,643	0,699
Thomas-Decady	0,156	0,180	0,351
symmetric max	0,307	0,313	0,521
non-symmetric max	0,044	0,148	0,211
mainstream	0,307	0,321	0,516

5 Conclusions

Generally, majority of statistics indicate that 5-persons households present smaller dependence, between heating materials they use and a type of income they have, than 2-persons households. There exists an exception for non-symmetric maximum statistic. Thus the higher "correlation" property is not absolute.

Note that SM has the smallest number of parameters so, as expected, the SM indicates the highest significance of relation between the questions.

The RM model gives smaller "correlations" (i.e. higher P-values) then the EM one. It seems that for 4x4 tables RM takes into account richer variety of aspects of distribution that characterizes behavior of **A** and **B** questions than EM model. Therefore a possible difference between EM and RM for a given statistic type can be meaningful. One may suspect that the significance of statistics for EM in main stream and symmetric statistics (2-person household) is an artifact and vanishes if we split the table into subtables. So it may be caused by a tendency of people to choose systematically two or one answer and has nothing to do with the content of a question.

Out of the list of statistics studied here, only SM model considers directly the parameters that appear in most appreciated in literature null hypothesis that formalizes "independence" between **A** and **B**. But, as we see here, apparently SM model is exaggeratedly "optimistic" in establishing significant correlation. This diagnosis needs of course a careful verification in further investigations, also with respect to implication for independence results of people not giving any answer a question or choosing all the answers.

References

1. Agresti A., Liu L-M, "Modelling a categorical variable allowing arbitrary many category choices", Biometrics 55, 936-943, 1999.
2. Bali G., Czerski D., Klopotek M. A., Matuszewski A., „Details that count for a data mining problem connected with the survey sample multi-response questions", Artificial Intelligence Studies, 23, 5-12, 2004.
3. Bali G., Matuszewski A., Klopotek M. A., "Dependence of two multiresponse variables – importance of the counting method" in: M. A. Klopotek, M. Michalewicz, S. T. Wierzchon (ed.), "Intelligent Information Systems 2003", Physica-Verlag (Springer), 2003.
4. Bilder C R., Loughin T. M., "On the first-order Rao-Scott correction of the Umesh-Loughin-Sherer statistic", Biometrics 57, 1253-1255, 2001.
5. Bilder C R., Loughin T. M., "Testing for marginal independence between two categorical variables with multiple responses", Biometrics 60, 241-248, 2004.
6. Decady Y. J., Thomas D. R., "A simple test of association for contingency tables with multiple column responses", Biometrics 56, 893-896, 2000.
7. Loughin T., Scherer P. N., "Testing association in contingency with multiple column responses", Biometrics 54, 630-637, 1998.
8. Matuszewski A., Trojanowski K., "Models of multiple response independence", in: M. A. Klopotek, M. Michalewicz, S. T. Wierzchon (ed.), "Intelligent Information Systems 2001", Physica-Verlag (Springer), pp.209-219, 2001.
9. Matuszewski A., Czerski D., Bali G., "Null hypotheses and statistics for testing independence between two multiresponse", ICS PAS Report, 2005.
10. Rao J.N.K., Scott A. J., "On chi-squared tests for multi-way tables with cell proportions estimated from survey data", Annals of Statist., 12, 46-60, 1984.
11. Thomas D. R., Decady Y. J., "Analyzing categorical data with multiple responses per subject", SSC Annual Meeting, Proceedings of the Survey Methods Section, 121-130, 2000.

Immunological Selection Mechanism in Agent-Based Evolutionary Computation

Aleksander Byrski and Marek Kisiel-Dorohinicki

Institute of Computer Science,
AGH University of Science and Technology,
Mickiewicz Avn. 30, 30-059 Cracow, Poland.

Abstract. Artificial immune systems turned out to be interesting technique introduced into the area of *soft-computing*. In the paper the idea of an immunological selection mechanism in the agent-based evolutionary computation is presented. General considerations are illustrated by the particular system dedicated to function optimization. Selected experimental results conclude the work.

1 Introduction

Looking for a computational model, that can be used to solve different difficult problems, researchers very often turned to the processes observed in nature. Artificial neural networks, which principles of operation originate from phenomena occuring in the brain, evolutionary algorithms based on the rules of organic evolution, multi-agent systems originating from observation of social processes are the examples of such approach. Also artificial immune systems, inspired by the human immunity, recently began to be the subject of increased researchers' interest. Such techniques are often combined together to create hybrid systems, that by the effect of synergy exhibit some kind of intelligent behaviour, which is sometimes called *computational intelligence* as opposed to rather symbolic artificial intelligence [2].

Following this idea a hybrid optimization technique of evolutionary multi-agent systems have been applied to a wide range of different problems [1]. In this paper immunological mechanisms are proposed as a more effective alternative to classical energetic ones. Particularly the introduction of negative selection may lead to early removing of improper solutions from the evolving population, and thus skip the process of energetic evaluation [5] and speed up the computation. Below, after a short presentation of the basics of human immunity and artificial immune systems, the details of the proposed approach are given. Some preliminary results allow for the first conclusions to be drawn.

2 Human immune system

Human immune system plays a key role in maintaining the stable functioning of the body. It allows for detection and elimination of disfunctional endogenous cells, termed infectious cells and exogenous microorganisms, infectious

non-self cells such as bacteria and viruses, which enter the body through various routes including the respiratory, digestive systems and damaged dermal tissues [4].

A key role in humoral immunity (the cellular immune layer) play lymphocytes, that can be divided into two major subtypes [7]:

B-lymphocytes mature in bone marrow, they are responsible for production of immunoglobulins (antibodies) – proteins that neutralize pathogens.

T-lymphocytes mature in thymus, they assist B-lymphocytes in eliminating of pathogens and are responsible for removing of disfunctional cells of the body. T-cells are subjected to a process called negative selection in thymus, where they are exposed to a wide variety self proteins, and destroyed if they recognize them.

3 Artificial immune systems

Different immune-inspired approaches were constructed and applied to enhance algorithms solving many problems, such as classification or optimization [8]. Basic principles of immune-based algorithms may be clearly illustrated on a classical problem of machine learning – concept learning – as described below.

Concept learning can be framed as the problem of acquiring the definition of a general category given a sample of positive and negative training examples of these categories' elements [6]. Thus the solutions can be divided into two groups: self (positive) and non-self (negative). Artificial immune system which consists of a large number of T-cells can be trained to recognize these solutions in the process called negative selection. Negative selection algorithm corresponds to its origin and consists of the following steps:

1. Lymphocytes are created, as yet they are considered immature.
2. The binding of these cells (affinity) to present self-cells (e.g. good solutions of some problem) is evaluated.
3. Lymphocytes that bind themselves to "good" cells are eliminated.
4. Lymphocytes that survive are considered mature.

Mature lymphocytes are presented with the cells that have unknown origin (they may be self, or non-self cells), and they are believed to have possibility of classifying them [8].

4 Immune-based selection in evolutionary multi-agent systems

The key idea of *EMAS* is the incorporation of evolutionary processes into a multi-agent system at a population level. This means that besides interaction

mechanisms typical for agent-based systems (such as communication) agents are able to *reproduce* (generate new agents) and may *die* (be eliminated from the system). For more details on this approach see [5].

In order to speed up the process of selection, based on the assumption that "bad" phenotypes come from the "bad" genotypes, a new group of agents (acting as lymphocyte T-cells) may be introduced. They are responsible for recognizing and removing agents with genotypes similar to the genotype pattern posessed by these lymphocytes. Of course there must exist some predefined affinity function, e.g. using real-value genotype encoding, it may be based on the percentage difference between corresponding genes. The lymphocyte-agent may be created in the system during the removal of the solution-containing agent, so it contains the genotype pattern of this agent, which was perceived as "bad". In both cases, the new lymphocytes must undergo the process of negative selection. In a specific period of time, the affinity of the immature lymphocytes patterns to the "good" agents (posessing relative high amount of energy) is tested. If it is high (lymphocytes recognize "good" agents as "non-self") they are removed from the system. If the affinity is low – probably they will be able to recognize "non-self" individuals ("bad" agents) leaving agents with high energy intact.

5 Immunological evolutionary multi-agent system for optimization

The immunologically-enhanced optimization algorithm may be described as follows:

1. Initialize the whole population of agents with random problem solutions and the initial energy level.
2. Let every agent and T-cell perform its tasks:
   (a) agent's specific tasks look as follows:
      - agent rendezvous – compare fitness with neighbours and exchange a portion of energy with them.
      - agent reproduction – if energy reaches certain level, look for the neighbour that is able to reproduce, and create new agent, then pass the fixed portion of the parents' energy to their child,
      - agent migration – if energy reaches certain level, leave the current island and move to another,
      - agent death – if energy falls below certain level – the agent is removed from the system and T-cell is created based on the genotype of the removed agent.
   (b) T-cell specific task: affinity testing – existing T-cells search the population for agents similar to the genotype pattern contained in a T-cell. If the T-cell survived the process of negative selection – it causes the removal of similar agent. Otherwise T-cell is removed from the system.

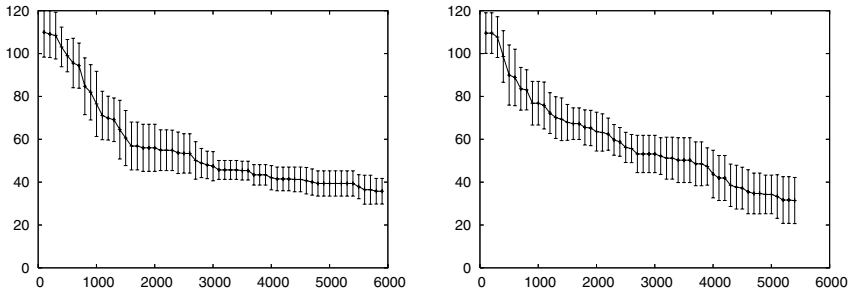


Fig. 1. Optimization results obtained for EMAS (a), and iEMAS (b)

3. Look for the best current solution presented by one of the agents in the population. Go to 2. until the ending condition is true.

In the proposed approach, real-value encoding of the solutions is used, and the affinity was determined using the relation of the corresponding genes, in the following way. Let $p = [p_1 \dots p_n]$ be a paratope (genotype pattern owned by the lymphocyte) and $e = [e_1 \dots e_n]$ be an epitope (genotype owned by the antigen – in the described system it is simply the genotype of the tested agent), and n is the length of the genotype. If d_{max} is the maximum deviation and c_{min} is the minimum count of corresponding genes, the construction of the set of corresponding genes may be considered:

$$G = \{p_i : \left|\frac{p_i}{e_i}\right| \leq d_{max}\}$$

The lymphocyte is considered as stimulated (its affinity reached minimal level) when $\overline{\overline{G}} \geq c_{min}$, and considered as non-stimulated otherwise.

6 Preliminary results

The experiments were performed in order to show whether the introduction of the immunological-based selection mechanism into EMAS will make the classical energetic selection more effective. Ten-dimensional Rastrigin function was used as a benchmark optimization problem.

Evolutionary multi-agent system consisted of one evolutionary island, with initial population of 100 agents. Before maturing, every lymphocyte undergone the negative selection process. During 10 system steps, if immature lymphocyte was stimulated by an agent with energy of 150 or higher, the cell was removed from the system.

In figure 1 the optimization results are presented in terms of fitness of the best solution in consecutive steps of the system activity. Since the number of agents in the system changes, these steps reflect the total number of

agents evaluated in the system. Each plot shows an average (and the value of standard deviation) fitness of the best agent obtained from the 10 runs of optimization with the same parameters. Comparing the plots, it seems that introduction of the immunological selection allows for slightly better approximation of the solution in the same time, yet the difference observed is not convincing at a time.

7 Conclusion

In the paper an immune-based selection mechanism for evolutionary multi-agent systems was presented. As the preliminary experimental results show, it allows for improper individuals to be removed from the system faster than using classical energetic selection, and thus allows for obtaining slightly better solutions in comparable time.

Up till now it is still too early to compare this method with various other optimization heuristics known from the literature. The work in progress should allow for the comparison with classical methods of immune-based and evolutionary optimization.

The approach seems to be especially adequate for solving problems in which fitness evaluation takes relatively long time, e.g. optimization of neural network architecture which requires training of the neural network each time it is evaluated [3]. Thus further research will surely concern applications based on evolutionary neural networks.

References

1. M. Kisiel-Dorohinicki, G. Dobrowolski, and E. Nawarecki. Agent populations as computational intelligence. In Leszek Rutkowski and Janusz Kacprzyk, editors, *Neural Networks and Soft Computing*, Advances in Soft Computing, pages 608–613. Physica-Verlag, 2003.
2. P. Bonissone. Soft computing: the convergence of emerging reasoning technologies. *Soft Computing*, 1(1):6–18, 1997.
3. A. Byrski, M. Kisiel-Dorohinicki, and E. Nawarecki. Agent-based evolution of neural network architecture. In M. Hamza, editor, *Proc. of the IASTED Int. Symp.: Applied Informatics*. IASTED/ACTA Press, 2002.
4. W.H. Johnson, L.E. DeLanney, and T.A. Cole. *Essentials of Biology*. New York, Holt, Rinehart and Winston, 1969.
5. M. Kisiel-Dorohinicki. Agent-oriented model of simulated evolution. In William I. Grosky and Frantisek Plasil, editors, *SofSem 2002: Theory and Practice of Informatics*, volume 2540 of *LNCS*. Springer-Verlag, 2002.
6. T. M. Mitchell. *Machine learning*. McGraw-Hill, 1997.
7. J. Twycross. *An Immune System Approach to Document Classification*, volume HPL-2002-288. HP Laboratories Bristol, 2002.
8. S. Wierzchoń. *Artificial Immune Systems [in polish]*. Akademicka oficyna wydawnicza EXIT, 2001.

Unfavorable Behavior Detection in Real World Systems Using the Multiagent System

Krzysztof Cetnarowicz[1], Renata Cięciwa[2], Edward Nawarecki[1], and Gabriel Rojek[3]

[1] Institute of Computer Science
AGH University of Science and Technology
Al. Mickiewicza 30, 30-059 Kraków, Poland
cetnar@agh.edu.pl
[2] Department of Computer Networks
Nowy Sacz School of Business — National-Louis University
ul. Zielona 27, 33-300 Nowy Sacz, Poland
rcieciwa@wsb-nlu.edu.pl
[3] Department of Computer Science in Industry
AGH University of Science and Technology
Al. Mickiewicza 30, 30-059 Kraków, Poland
rojek@agh.edu.pl

Abstract. Nowadays detecting destructive attacks and dangerous activities is crucial problem in many real world security systems. A security system should enable to distinguish some actors which effects of behavior are perhaps unfavorable for a considered area. The considered areas are real world systems e.g. airports, shops or city centers. Project of real world security system should assume the changing and unpredictable type of dangerous activities in real world systems. Security system has to detect and react to new kind of dangers that have never been encountered before. In this article there are presented methods derived from some ethically–social and immunological mechanisms that should enable automated intrusion detection.

1 Real world system and its computer model

A real world system (e.g. airport, shop or city center) can be considered as an environment with resources and actors. Resources (e.g. cars, rubbish bins) have characteristic properties and can be perceived, destroyed or modified by actors. Actions of every actor collected in a period of time form a sequence that define his behavior. The length of this sequence is meaningful and analyze of actor's behavior is more unambiguous if stored sequences of actions are longer.

On the base of the real world system it is possible to build computer model that simulates events which take place in this real system. It can be realized a real-time simulation of a real system. Events in simulated system are determined by real events taking place in the real world system. In presented work there is used a multiagent model of the real world system in which the real resources are represented by simulated resources and actors of the real system are represented by agents.

2 Estimation of behavior

Actions undertaken by an agent are considered as objects. These objects form a sequence which is registered by all agents existing in the system. Similarly to real world systems where all individuals in society observe and estimate other entities (whose actions could be noticed), in multiagent system all agents can observe and estimate behavior of all individuals in the monitored area.

Registered sequences of objects could be processed in order to decide whether the behavior of the acting agent is *good* — favorable or *bad* — unfavorable or with unfavorable consequences. It should be mentioned, that the quoted notions of *good* and *bad* do not have absolute meaning. *Good* behavior or *good* agent is a desirable individual for a system (monitored area) in which evaluation takes place. *Bad* agent is an undesirable agent in a considered system.

2.1 Behavior estimation inspired by immunological mechanisms

The immune system appears to be precisely tuned to the problem of detecting and eliminating pathogens. There are a lot of similarities between the problem of detecting pathogens (unfavorable chains of molecules) and detecting of *bad* behavior (unfavorable chains of actions). Particular mechanisms, inspired by immunological mechanisms (as found in [4–6]) are applied to estimate the behavior of an agent.

Immunological mechanisms which are used in unfavorable behavior detection operate on structures that consist of actions committed by agents. That structures have form of sequences (chains) of objects — actions performed by agents. The length of a chain is defined as l (i. e. every chain contains l objects). Every object represents one action undertaken by an agent.

The algorithm of behavior estimation inspired by immunological mechanisms is the following (from a point of view of an exemplary agent):

1. permanently observe and store actions (corresponding objects) undertaken by every agent visible in the environment (system);
2. once after a fixed number of observed actions (undertaken by every agent) generate corresponding detectors;
3. when detectors are generated — evaluate (estimate) behavior of every agent in the system on the base of observed actions which are permanently stored.

The generated detectors have a form of fragments of sequences (subsequences) composed of objects representing actions. A subsequence may be considered as a detector if it does not appear in a sequence of actions of an agent that owns this detector. The process of creation of detectors takes place after a fixed number of memorized actions because we need to gather knowledge about undertaken actions. Behavior estimation consists of verification

of sequences of all agents' actions in monitored area. Behavior of an agent is evaluated as *bad* if its sequence of actions is similar (within a fixed level) to detectors.

2.2 Behavior estimation in multiagent system

Section 2.1 presentes behavior estimation mechanisms from a point of view of an agent. That mechanisms built-in into an exemplary agent are named *division profile*. Precise description of division profile was presented in [1–3]. From a point of view of agents' society, all agents in the simulated system should be equipped with division profile so the security will be assured by all agents existing in the system. This is very similar to some ethically–social mechanisms that act in societies — security in a society is assured by individuals who function in this particular society.

If all agents undertake autonomous decisions about their aims which agent is *bad* and should be eliminated, there have to be applied mechanisms which collect that results of behavior estimations. Mentioned mechanisms (presented precisely in [1–3]) prevent deleting of agents which are chosen to be deleted by small amounts of agents.

3 Experiment

In order to confirm effectiveness and utility of proposed solutions multiagent system with asynchronously acting agents was implemented. In the environment of simulated system there are two types of resources: resources of type A and resources of type B. Resources are used by agents separately, but refilling of all resources is possible only when both resources reach an established low level. This simulation reflect a real world systems with some operations which should be executed in couples e.g. leaving baggage / taking baggage.

Two stated types of agents can exist in the environment of research system:

- *type g=0 agents* — agents which in every life cycle take one unit of randomly selected (A with probability 50%, B with probability 50%) resource; type g=0 agents need units of resources to refill energy; if energy level of an type g=0 agent goes down, this agent is eliminated;
- *type g=1 agents* — agents which take one unit of A resource in every life cycle; their existence does not depend on their energy level.

3.1 Results: unsecured multiagent system

Two cases were simulated in the first set of experiments, that were initially 50 *type g=0 agents* or 40 *type g=0 agents* and 10 *type g=1 agents* in the system, which agents did not have any security mechanisms. 10 simulations

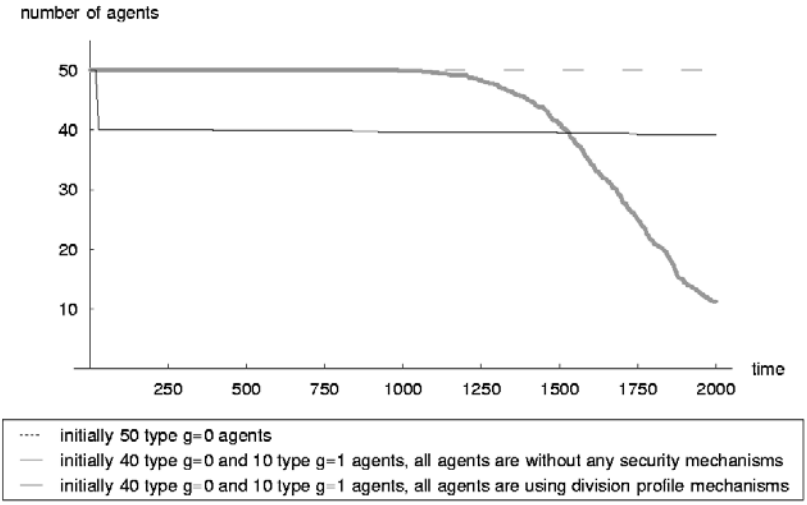


Fig. 1. The system without intruders, intruders inside the system, intruders inside the system with agents with built-in division profile

were performed and the diagram in Fig. 1 shows the average numbers of agents in separate time periods.

It is noticeable, that if there are not any *type g=1 agents* in the simulated system, all *type g=0 agents* could exist without any disturbance. The existence of *type g=1 agents* in the system caused problems with executing tasks of *type g=0 agents* which died after some time periods. Intruders still remained in the system.

3.2 Results: secured multiagent system

There was a case simulated, in which initially there were 40 *type g=0 agents* and 10 *type g=1 agents* and all agents in the system were equipped with the division profile mechanisms. 10 simulations were performed and diagram in Fig. 1 shows the average number of agents in separate time periods.

Last 18 actions undertaken by every agent were stored in the environment. During first 18 periods of time all agents acted synchronously. After 18 actions had been undertaken by every agent, detectors were constructed of length $l = 5$ and all agents' actions became asynchronous. Agents used their division profile mechanisms to calculate which neighboring agent they want to eliminate. Agent demanded to eliminate these neighbors which had the maximum of detector's matchings. The environment sumed up these coefficients and eliminated the agent a when final sum of coefficients (attributed to agent a) was larger than constant OU and more than 50 per cent of agents had a goal to eliminate agent a. The constant OU was set up to 300 (more

information about constant *OU* and other mechanisms of division profile can be found in [1–3]).

As results presented in Fig. 1 verify after detectors had been constructed, intruders were distinguished and deleted due to the division profile mechanisms what makes it possible for *type g=0 agents* to function freely.

4 Conclusion

Some conceptions presented in this paper show how to solve the problem of automated security assuring in a real world system such as airport or shopping center. Use of multiagent technology was proposed, particularly equipping all agents existing in a system with some security functions (called the division profile). Presented solutions are inspired by some ethically–social and immunological mechanisms.

In order to confirm the effectiveness of our conception a multiagent system was implemented to reflect a simple real world system with two types of actions which can be monitored. The results obtained in these simulations demonstrate that the proposed method successfully detects an abnormal behavior of actors in monitoring system. Main features of applying mechanisms (i.e. distribution, flexibility, adaptability) seem to guarantee a good detection of unfavorable behavior in security systems.

References

1. Cetnarowicz, K., Rojek, G. (2003) Unfavorable Beahvior Detection with the Immunological Approach. In Proceedings of the XXVth International Autumn Colloquium ASIS 2003, MARQ, Ostrava, 41–46
2. Cetnarowicz, K., Cięciwa, R., Rojek, G. (2004) Behavior Based Detection of Unfavorable Activities in Multi-Agent Systems. In MCPL 2004, Conference on Management and Control of Prodaction and Logistics, Santiago — Chile, 325–330
3. Cetnarowicz, K., Rojek, G. (2004) Behavior Based Detection of Unfavorable Resources. In Lecture Notes in Computer Science, Proceedings of Computational Science - ICCS 2004: 4th International Conference, Springer-Verlag, Heidelberg, 607–614
4. Forrest, S., Perelson, A. S., Allen L., Cherukuri R. (1994) Self-nonself Discrimination in a Computer. In Proc. of the 1994 IEEE Symposium on Research in Security and Privacy, IEEE Computer Society Press, Los Alamitos, 202–212
5. Hofmeyr, S. A., Forrest, S. (2002) Architecture for an Artificial Immune System. Evolutionary Computation, vol. 7, no. 1, 45–68
6. Wierzchoń, S. T. (2001) Sztuczne systemy immunologiczne: teoria i zastosowania. Akademicka Oficyna Wydawnicza Exit, Warszawa

Intelligent Navigation in Documents Sets Comparative Study

Maciej Czyżowicz

Warsaw University of Technology, Plac Politechniki 1, Warsaw, Poland

Abstract. Today's text searching mechanisms are based on keyword search. However a large number of results makes the searches less effective. This paper presents results when search of relevant documents is obtained based on currently browsed document instead of keywords. Five keywords based search methods were tested. Comparative study was done on following methods: LSA, PLSA, WebSOM, PHITS, Query Dependant PageRank.

1 Introduction

A constantly growing information set stored in the Internet requires better and faster models of searching. Role of a navigation system in documents set is to present documents relevant to current document.

This paper presents usability of five different methods in obtaining most relevant documents. Capabilities of those methods were studied in terms of keyword query. Keyword queries contain up to few terms. Achieved results present how those methods behave when a whole document was designated as a query. Therefore for calculation of similarity not only words from document could be used. Structure of document such as word frequency or position (both absolute and relative) could be extracted and used to find similar or related document.

First works were carried out in field of library science. Those works based on a content of documents. Words of document created whole of search space. After Internet boom a set of algorithms was devised to support exact and similarity searches in text documents, other were based on based on other characteristics other then only word content. Some of those algorithms were deterministic [1], other algorithms were probabilistic [2]. An aspect of cross-referencing was used mostly. Some researches are done in changing representation of search results from text based to graph based [3]. Current works investigate usage of probabilistic methods. A new method emerged in 2002 called "Latent Dirichlet Analysis" which is a second order method. Studies on that method are currently underway.

Next section describes how comparison was done, section three gives a short overview of methods compared, in section four an empirical results are presented, and section five gives conclusions.

2 Metrics

Five methods were compared in this article. Those methods can be separated into two distinct groups. LSA, PLSA, WebSOM can be defined as methods based on text (words) of the document. PHITS and PageRank are methods based on references between documents.

Data used for this study were obtained from *National Institute of Standards and Technology* (NIST). NIST created set of documents as a test data for *Document Understanding Conferences*[1]. (79 groups).

Different characteristics of compared methods required defining a parameters that independent of characteristic of given algorithm. Each of compared method was returning a set of documents. In this document following phrases will be used:

similar documents documents that were in the same group in original data set

answer set set of documents that were returned by method for given query

For purpose of this study two different parameters where defined:

recall This parameter is a proportion of similar documents that were returned by method to number of documents in original set.

precision This parameter defines proportion of similar documents that were returned by method to number of documents returned by method.

Similar documents were understood to be ones that belong to the same group in original grouping done by experts.

For text based searches each document was preprocessed to have its vector representation (so called bag-of-words representation). For each text based method conducted tests included following cases of document sets: all words in space, all words in space without stop-list, stem of words in space, stem of words in space without stop-list. Based on occurrence of words in whole space a stop-list was obtained. Stop list contains words that constitute top 10% of total number of used words.

For reference based methods a simulated internet environment was set up. For each group a whole graph was created. Each document in group referenced every other document in the same group. This ideal graph was randomly modified. Three variants of modifications were created: 2%, 5% and 10% of total number of edges was deleted and a same number of edges was randomly inserted.

Input data for all of the algorithms used in this study were stripped of any information about groups, that were assigned by experts. Each of the probabilistic methods were run three times so that bad runs could be dismissed.

[1] http://www-nlpir.nist.gov/projects/duc/

3 Compared Methods

In this section a short definition of methods, and specific modification for this study will be presented.

Latent Semantic Analysis [1] assumes that each document belongs to a group. Number of groups is known apriori. Factor which determines each document assignment to a specific group is unknown.

A query in this method would be passed as a vector. Each document from original space has assigned position in reduced space. Documents that have same position in reduced space are in the same answer set.

Idea of **Probabilistic Latent Semantic Analysis** is the same as LSA, however the method of achieving the assignment is probabilistic. PLSA approach is based on idea of learning from dyadic data without supervision. In case of intelligent navigation those independent sets are documents and words [2].

Each document has assigned probability of belonging to each aspect. Therefore each resulting document set was chosen on basis of an aspect. Document belongs to a given set, when the probability for an aspect is the highest.

WebSOM is an extension of Self-Organizing Map created by prof. T. Kohonen [3]. Since WebSOM works better with square maps (when small number of nodes is created), instead of simulating exact number of tested groups, a nearest (in number of nodes) square map was used.

For purpose of this study set of nodes was relatively small, and after achieving stability each document was assigned node. Node represented as an answer to a query for a document from that group.

Probabilistic Hypertext Induced Topic Selection [4] which comes from merging two ideas: PLSA and HITS. HITS algorithm is one of the first algorithms that uses references as a source of information about document. PHITS instead of words as a second independent variable it uses references. Therefore two independent variables come from the same space.

An answer set was created similarly to a PLSA method.

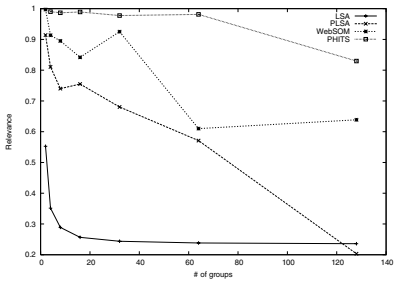
Query-Dependent PageRank(QDPR) is a modification of PageRank [5] which dismisses notion of uniform weights for links. Links between documents are weighted based on relevance to a given query.

Answer set was constructed on a basis of choosing top 50 documents that had the highest probability.

4 Results

All algorithms were run on the same set of data. Each algorithm divided documents into groups of 2, 4, 8, 16, 32, 64, 128[2] documents (Original number of 79 document groups was not chosen). Best result for each of two measures

[2] For WebSOM actual number of groups were 4, 9, 16, 36, 64, 144.



# of groups	LSA	PLSA	WebSOM	PHITS
2	0.552	0.914		0.997
4	0.351	0.810	0.913	0.990
8	0.288	0.739	0.895	0.986
16	0.256	0.754	0.841	0.989
32	0.243	0.680	0.925	0.977
64	0.238	0.570	0.610	0.981
128	0.235	0.203	0.638	0.829

Fig. 1. The best average recall.

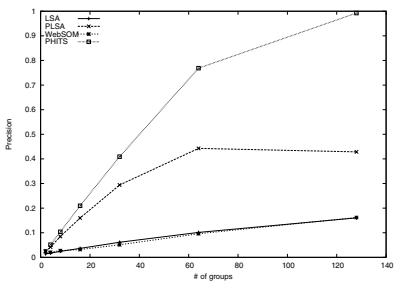
is represented in figures 1 and 2. Overall best performance was achieved by PHITS.

The second the best for recall was WebSOM. WebSOM creates pseudodocuments which contain keyword characteristic for a group of documents. PLSA method, which was the longest running method, shows dramatic decrease in ability to find similar documents when number of aspects is greater then number of groups set by experts.

In terms of having relevant documents in an answer set for given document PLSA had best results (except PHITS). Result around 40% is not satisfactory. It means that more then half of documents in returned data set is irrelevant. Almost all of the algorithms were achieving increase in precision of the returned answer set with increasing number of groups (except PLSA).

Results of PLSA suggest possibility of a good algorithm for finding empirically optimal number of groups (classes). In both measures PLSA shows decrease in performance when number of groups was exceeding original number of groups.

LSA had the worst performance in this study. Closer examination showed that LSA has tendency to cluster results in non-uniform fashion. Table 1 shows number of empty groups for case of 64 and 128 groups. Similar precision results for WebSOM suggest similar effect. Much better recall values show better theme grouping ability of WebSOM.



# of groups	LSA	PLSA	WebSOM	PHITS
2	0.014	0.024		0.026
4	0.017	0.041	0.020	0.051
8	0.024	0.084	0.027	0.103
16	0.036	0.159	0.031	0.209
32	0.061	0.293	0.051	0.409
64	0.101	0.442	0.095	0.768
128	0.160	0.428	0.162	0.992

Fig. 2. The best average precision.

	64	128
All words	5	45
All words (no stop list)	0	21
Stems	8	54
Stems (no stop list)	2	33

Table 1. Number of empty groups for LSA

All text based algorithms were achieving best results on input sets that had excluded words from stop-list. The interesting fact is that using stems instead of words was only marginally better.

Query-Dependent PageRank based on the best results of PLSA was unsatisfactory. That is the best average precision was at the level of 14.1% and the best average recall value was at the level of 18.7%.

5 Conclusions

Ability to find relevant documents based on text of document oscillates at level of 40-50% percent . Using references for finding related documents allows for a big increase in efficiency. Therefore it seems that intelligent navigation systems should be based on references while text analysis and text similarities should be used as support. For text analysis a WebSOM algorithm seems to be better then others, while LSA algorithm is the worst choice.

Some other research into text analysis should be done. Research into field of phrase and sentence analysis should be done. It is expected that running WebSOM or PLSA based only on nouns will increase its efficiency will increase efficiency while decreasing time and resource required to arrange document set. Other extension for navigation systems would be to offer relevant documents not only as a result of single query, but using history of usage (both as history of queries, or history of user movement in document set).

References

1. Deerwester Scott C., Dumais Susan T., et al. (1990) Indexing by Latent Semantic Analysis. Journal of the American Society of Information Science **41**, 391–407
2. Hofmann T. (1999) Probabilistic Latent Semantic Analysis. Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence, 289–296
3. Kohonen T. (1995) Self-Organizing Maps, Springer Series in Information Sciences
4. Cohn D., Chang H., (2000) Learning to Probabilistically Identify Authoritative Documents. Proc. 17th International Conf. on Machine Learning 167–174
5. Richardson M., Domingos P., (2002) The intelligent surfer: Probabilistic combination of link and content information in pagerank. In Advances in Neural Information Processing Systems 14.

Assessing Information Heard on the Radio

Antoni Diller

School of Computer Science, University of Birmingham, B15 2TT, UK

Abstract. Much work in AI is devoted to the problem of how computers can acquire beliefs about the world through perception, but little effort is devoted to the problem of how computers can acquire beliefs through testimony. This paper is part of a continuing project whose ultimate goal is that of constructing an implementable model of how agents acquire knowledge through testimony. In particular, it looks at how agents acquire information from the radio and many factors are identified that may cause an agent to override the defeasible rule to believe what he hears.

1 Introduction

This paper is written as part of the continuing work of the Electric Monk Research Project [3–10]. The Electric Monk is a fictional character, created by the science-fiction humorist Douglas Adams, who 'believed things for you, thus saving you what was becoming an increasingly onerous task, that of believing all the things the world expected you to believe' [1, p. 3]. The main goal of this Research Project is to model the human ability to acquire knowledge through testimony in sufficient detail to allow a computer program to be written to emulate this ability. The study of testimony is almost totally ignored in AI and yet AI's ultimate goal, namely that of constructing an artificial person [2, p. 7], cannot be achieved until we can give androids the ability to acquire beliefs through testimony. As well as having a large amount of in-built knowledge and the ability to acquire beliefs by means of perception, a general-purpose, intelligent and autonomous android would also have to have the ability to extend its knowledge by accepting other people's assertions. Such an ability is needed if the android is going to be able to interact with human beings in any meaningful way [7,9].

Although we get most of our information about the world from other people by believing what they assert, we do not believe every assertion. I have argued that our way of dealing with assertions is governed by the defeasible rule to believe them [7,9]. This sounds very simple, but it is extremely fruitful because it forces us to look at the reasons why we decide not to accept another person's word. In other papers I have looked at the factors that cause us to override the defeasible rule to believe other people when we are hearing what another person says to us in the flesh [5], when we are reading a book [6] and when we are reading a journal article [4]. In this short paper I briefly summarise the factors that may cause an agent not to accept the assertions that he hears on the radio. Examples of how these factors work in practice can be found in the long, but unpublished, version of this paper [8].

The ability to evaluate assertions is very different from that of understanding them. My concerns are epistemological and not semantic. Natural-language processing deals with the problem of how language is understood. I am interested in the totally different question of how we decide whether or not to accept an assertion once we have understood it.

Several years ago I introduced a two-stage model of belief-acquisition [3]. Since then the model has been considerably improved. Here I briefly summarise the latest version [7, pp. 4–7]. There are two main ways in which we acquire beliefs directly, namely through perception and through testimony. These constitute the first phase of belief-acquisition. How we acquire knowledge through perception is not my concern here. I have argued that our acceptance or rejection of other people's assertions is governed by the defeasible rule to believe them [9]. This rule, however, is not infallible. Sometimes we acquire beliefs that are in fact false and sometimes we reject assertions that are in fact true. Thus, there is a second phase of belief-acquisition in which we critically examine some of our beliefs in order to weed out the false ones and replace them with better ones. In the second phase we can also re-examine some of the assertions we have rejected in order to check whether or not they are true after all.

The way in which I unpack the defeasible rule to believe other people's assertions is to represent it as an ordered set of rules all of which except the last are conditional ones. The last rule is the non-defeasible rule to believe the assertion in question. I call such a set of rules an *assessment component*. There are many reasons why someone may decide not to accept an encountered assertion and each such reason becomes the antecedent of one of the conditional rules in the assessment component. For example, a radio play is a work of fiction and so we do not normally believe the various assertions that the actors performing it make. This can be captured by adding the following conditional rule to the assessment component, 'If the assertion X is uttered by an actor during the performance of a radio play, then reject X.' (This is a simplification of the actual way in which we treat assertions by actors, since sometimes they are made to say things that are true in the real world.)

So far I have been describing the first stage of belief-acquisition. The evaluations that we carry out concerning the assertions we encounter have to be done in real time. The decision has to be made virtually instantaneously whether or not to accept an assertion. This means that the assessment cannot be very sophisticated. So, people will come to have some false beliefs and they will also reject some assertions which, as a matter of fact, are true. In my model, therefore, there is a second stage of belief-acquisition in which a small number of assertions are subjected to a detailed and possibly time-consuming investigation. A person can either check something that he believes or investigate an assertion that he previously rejected. When someone checks one of his beliefs, he may either decide that he was correct in having that belief in the first place or he may conclude that on this occasion he made a

mistake and reject the belief. Similarly, when someone investigates the truth of an assertion that he rejected previously, he may change his mind about its truth-value and add it to his belief-system or he may decide that he was correct in rejecting it and not alter his assessment of it. When a person does change his mind about the status of an assertion, he will probably need to make further revisions to his belief-system in order to remove any obvious inconsistencies arising out of the change of mind. I do not deal with belief-revision in this paper.

Many different critical methodologies are used in the second stage of belief-acquisition to thoroughly test the accuracy of our beliefs. This is because there is not a single methodology that can be used to check the correctness of every kind of factual assertion. Different methods are used for different kinds of assertion. For example, the way in which someone checks a physical assertion, such as 'The speed of light in a vacuum is 299,792,458 metres per second', is very different from the historical methodology that has to be used to decide whether, for example, General Sikorski died in an accident or was murdered and, if he was assassinated, who was responsible. Furthermore, not everybody has the ability to check every kind of belief. A historian would know how to investigate the circumstances of General Sikorski's death and a physicist would know how to determine the value of the speed of light, but the historian is unlikely to have the knowledge to work out what the speed of light is and the physicist is unlikely to be able to conduct historical research into issues relating to what happened in the past. The topic of how and why people go about checking their beliefs is a vast one, but, unfortunately, I cannot pursue it further here.

2 Assessing Information Heard on the Radio

2.1 Introduction

A person listening to the radio does not accept every assertion he hears. There are many reasons for this. I group these into five categories: (1) We sometimes take *external* factors into account. These relate to such things as the country in which the radio station is based and the time at which we happen to be listening to the radio. (2) Sometimes information about the *assertor* may cause us to be wary of accepting an assertion he makes. He may, for example, be well known to be unreliable. (3) There may be something about the *manner of delivery* of an assertion that makes us hesitate to accept it. (4) The actual *content* of the assertion may make us think that it is not correct and incline us to reject it. (5) There may be something about the *listener* that makes him wary of accepting a particular assertion.

2.2 External Factors

There are features of the context in which an assertion is made, rather than of the assertion itself, that may incline an agent to reject that assertion.

These include the time when the assertion is broadcast, the owner of the radio station which broadcast it, the location of that radio station, the type of programme during which it was uttered and the political situation in the country at the time the assertion was made. Examples of how these factors work can be found in the full version of this paper [8, pp. 9–11].

2.3 Factors Relating to the Assertor

Sometimes there are things that we know about the assertor that influence our assessment of the correctness of what he says. These relate to the role that the assertor plays in the programme concerned, the assertor's ideology, the character of the assertor, any possible agenda that the assertor may have in appearing on the radio, any relevant expertise that the assertor may have (if talking about some specialist topic), the age of the assertor and the assertor's gender. Examples of how these factors work can be found in the full version of this paper [8, pp. 12–13].

2.4 Factors Relating to the Manner of Delivery

All of us learn our first language in a social context and as we learn how to speak we also acquire beliefs through accepting what we are told. In due course, we learn not to trust everything that we are told. We are taught what to look for in an assertion that may indicate that it is not worth accepting. For example, a person's body language may make us think that he is lying to us. Clearly, such features are not available to us when we are listening to the radio. There are, however, features of the way in which someone speaks that may make us think that he is not being completely truthful. Examples of why an agent may choose not to believe what he hears on the radio, because of the way in which the assertion is uttered, can be found in the full version of this paper [8, p. 14].

2.5 Factors Relating to the Content of the Message

There may be features of the content of an assertion that alert us to the possibility that it may be inaccurate. Such features include the consistency of the message, whether it induces any strong emotion in the listener, whether it is about something that is out of the ordinary and whether it is about the sorts of thing that people often lie about or make mistakes about. Examples of how these factors work can be found in the full version of this paper [8, pp. 14–15].

2.6 Factors Relating to the Listener

There are a number of properties of the person listening to the radio that may make him wary of accepting what he hears. These relate to the listener's pre-existing knowledge, to any possible consequences that result from accepting

the assertion being evaluated, to the relevance of the assertion to the listener, to the listener's character, to the maturity of the listener, to the listener's gender and to the situation in which the listener is in when he hears the assertion. Examples of how these factors work can be found in the full version of this paper [8, pp. 16–18].

3 Conclusion

Many problems have to be overcome before scientists succeed in building a humanoid robot with intellectual abilities analogous to those possessed by human beings. Before we can even begin to design such an android we have first to understand the abilities that humans have. In this paper I have been concerned with developing a model of the human ability to evaluate testimony. I have looked at how people acquire information from the radio and I have mentioned many factors that may cause them to override the defeasible rule to believe others. A great deal of work still needs to be done before we can implement a realistic model of belief-acquisition in a computer program. I have, however, done some of the ground-breaking work on this topic. I hope that some people reading this paper will be stimulated to join me in this exciting, but sadly neglected, field of AI research.

References

1. Adams, D. (1988) Dirk Gently's Holistic Detective Agency. Pan, London.
2. Charniak, E., McDermott, D. (1985) An Introduction to Artificial Intelligence. Addison-Wesley, Reading.
3. Diller, A. (1999) The belief-filter component. Cognitive Science Research Papers CSRP–99–9, School of Computer Science, University of Birmingham.
4. Diller, A. (2000) Evaluating information found in journal articles. In Nepomuceno, Á., Quesada, J.F., Salguero, F.J., editors, Logic, Language and Information: Proceedings of the First Workshop on Logic and Language, 71–78. Kronos, Sevilla.
5. Diller, A. (2000) Everyday belief-acquisition. In Henning, G.P., editor, Argentine Symposium on Artificial Intelligence (ASAI2000) Proceedings, 221–232. SADIO, Buenos Aires.
6. Diller, A. (2001) Acquiring information from books. In Bramer, M., Preece, A., Coenen, F., editors, Research and Development in Intelligent Systems XVII: Proceedings of ES2000, 337–348. Springer, London.
7. Diller, A. (2002) A model of assertion evaluation. Cognitive Science Research Papers CSRP–02–11, School of Computer Science, University of Birmingham.
8. Diller, A. (2002) Assessing Information Heard on the Radio. Cognitive Science Research Papers CSRP–02–12, School of Computer Science, University of Birmingham.
9. Diller, A. (2003) Designing androids. Philosophy Now, no. 42, 28–31.
10. Diller, A. (2003) Modelling assertion evaluation. AISB Quarterly, no. 114, 4.

Belief Rules vs. Decision Rules: A Preliminary Appraisal of the Problem

Jerzy W. Grzymała-Busse[1,2], Zdzisław S. Hippe[1], and Teresa Mroczek[1]

[1] Department of Expert Systems and Artificial Intelligence, University of Information Technology and Management, ul. Sucharskiego 2, 35-225 Rzeszów, Poland, e-mail: {zhippe,tmroczek}@wenus.wsiz.rzeszow.pl
[2] Department of Electrical Engineering and Computer Science, University of Kansas, Lawrence KS 66045-7523, USA, e-mail: jerzy@ku.edu

Abstract. An in-house developed computer program system ***Belief*SEEKER**, capable to generate belief networks and to convert them into respective sets of belief rules, was applied in mining the melanoma database. The obtained belief rules were compared with production rules generated by **LERS** system. It was found, that belief rules can be presumably treated as a generalization of standard **IF...THEN** rules.

1 Introduction

In this research two distinct learning models were generated (using an in-house developed computer program system ***Belief*SEEKER** [1]), and then applied for classification of unseen cases of melanoma skin lesions. The first model was based on belief networks, whereas the second one was grounded on typical decision rules. The quality of the developed learning models was additionally compared with results gained by the program **LERS** [2], a well-known standard of rules generating systems.

2 Research tools used

- ***Belief*SEEKER** — is a computer program system, capable to generate learning models (for any type of decision table prepared in the format set about by Pawlak [3]) in a form of belief nets, applying various algorithms [4]. The development of belief networks is steadily controlled by a specific parameter, informing about the maximum dependence between variables, known as marginal likelihood:

$$ML = \prod_{i=1}^{v} \prod_{j=1}^{q_i} \frac{\Gamma(\alpha_{ij})}{\Gamma(\alpha_{ij} + n_{ij})} \prod_{k=1}^{c_i} \frac{\Gamma(\alpha_{ijk} + n_{ijk})}{\Gamma(\alpha_{ijk})} \tag{1}$$

  Here:
  i=1,...,v — where v is the number of nodes in the network,

$j = 1, ..., q_i$- is the number of possible combinations of parents of the node X_i. (if a given attribute does not contain nodes of the type "parent", then q_i get the value of 1),
$k = 1, ..., c_i$ — where c_i is the number of classes within the attribute X_i,
n_{ijk} — is the number of rows in the database, for which parents of the attribute X_i have value j, and this attribute has the value of k, and
α_{ijk}, α_{ij} - are parameters of the initial Dirichlet's [5] distribution.

Characteristic features of the system, yet not described by us are: (**i**) capability to generate various exhaustive learning models (Bayesian networks) for different values of Dirichlet's parameter [5], (**ii**) capability to convert generated belief networks into relevant sets of belief rules of the type **IF...THEN**, (**iii**) built-in classification mechanism of unseen cases, and (**iv**) built-in mechanism for the assessment of obtained rules. It is worth to mention, that developed learning models (belief nets) can be converted into some sets of belief rules, characterizes by a specific parameter called by us *belief factor* **BF**, that reveals indirectly the influence of the most significant descriptive attributes on the dependent variable. Also, to facilitate the preliminary evaluation of generated rules, additional mechanism supports the calculation of their specifity, strength, generality, and accuracy [6].

- **LERS** [7] — data mining system (acronym from **L**earning from **E**xamples based on **R**ough **S**ets), developed at the University of Kansas, Lawrence, KS, USA, induces a set of rules from any kind of data (even from inconsistent data) and classifies new cases using the set of rules induced previously. These rules are more general than information contained in the original input database, since more new cases may be correctly classified by rules than may be matched with cases from the original data. More details about the system **LERS** can be find elsewhere [8].

3 Experiments. Chosen results and discussion

The well-known database [9] containing 548 cases of melanoma skin lesions was used in our research. It should be stressed that each case stored in the database was diagnosed, and confirmed histopathologically as belonging to one of the four concepts: *Benign nevus*, *Blue nevus*, *Suspicious nevus*, and *Melanoma malignant*. The database was randomly divided into two independent subsets. The first subset (**E384144.TAB**; **384** cases, **14** attributes, **4** concepts) was used for development of learning models, whereas the second (**E164144.TAB**; **164** cases, **14** attributes, **4** concepts) served for testing models. Information regarding the distribution of concepts in these sets is shown in Table 1.
In the first step of the research, computer program system ***Belief*SEEKER** was used to generate three different belief networks, having **Dirichlet's** para-

Table 1. Distribution of investigated cases in learning and training sets

Diagnosed concept	**E384144.tab** (learning set)	**E164144.tab** (testing set)
Benign nevus	160	88
Blue nevus	64	14
Suspicious nevus	80	34
Melanoma malignant	80	28

meter α=1, 30 and 60 respectively. The basic difference in the structure of these networks relied on the number of the most distinctive (important) attributes, displaying direct influence on the dependent variable (type of skin lesion, the decision). It was found, for example, that for the network with α=1 two descriptive attributes, namely $< TDS >$ and $<$ *color blue* $>$ seemed to be the most important. Then, for networks with α=30, additional attribute ($<$ *asymmetry* $>$), appeared in the network. Finally, in the case of the third network, besides attributes pointed out earlier, the attribute $<$ *dark brown* $>$ was observed.

The significance of these attributed was of great importance. Based on this information — in the next step of the research — for variable values of **BF** parameter — various sets of rules were generated for each of the networks (for ***Belief*SEEKER**), and one, unique set of rules was generated by **LERS** (see Table 2 and Table 3). It was found that the best results were obtained for **BF** = 0.7.

Table 2. The summary of results of classification of unseen cases by ***Belief*SEEKER** and **LERS**

	***Belief*SEEKER**			
	Dirichlet's parameter			**LERS**
	α=1	**α=30**	**α=60**	
Number of rules	20	31	42	49
Number of cases classified correctly [%]	95.12	95.12	87.80	87.8
Number of cases classified incorrectly [%]	**4.88**	**4.27**	**4.88**	**6.10**
Number of cases unclassified [%]	0	0.61	7.32	6.10

In the last step of the research, selected information (like specificity, strength, generality and accuracy of generated rules) allowed to fix the strongest belief rules and decision rules (Table 4).

It can be assumed that, the increase of Dirichlet's parameter causes extending the most crucial belief rules (Table 4) by merging another attributes. It was

Table 3. Occurrences of selected attributes in belief rules (***Belief*SEEKER**) and decision rules (**LERS**)

Descriptive attributes		***Belief*SEEKER**			**LERS**
		Dirichlet's parameter			
		α=1	**α=30**	**α=60**	
	asymmetry		31	42	16
	border				25
Color	*black*				11
	blue	20	31	42	18
	dark brown			42	5
	light brown				6
	red				16
	white				11
Diversity of structure	*pigment dots*				11
	pigment globules				6
	pigment network				15
	structureless areas				9
	branched streaks				9
	TDS	20	30	42	33

Table 4. Selected examples of the most important belief rules and decision rules

***Belief*SEEKER**			**LERS**
Dirichlet's parameter			
α=1	**α=30**	**α=60**	
RULE 5 IF TDS >= 4.080 AND TDS < 4.850 AND C_BLUE IS absent THEN DIAGNOSIS IS Benign_nev	**RULE 3** IF TDS >=4.080 AND TDS < 4.850 AND C_BLUE IS absent AND ASYMMETRY IS 1_axial_as THEN DIAGNOSIS IS Benign_nev	**RULE 6** IF TDS >= 4.080 AND TDS < 4.850 AND C_BLUE IS absent AND ASYMMETRY IS 1_axial_as AND C_d_BROWN IS present THEN DIAGNOSIS IS Benign_nev	**RULE 1** IF C_BLUE IS absent AND C_d_BROWN IS present AND C_RED IS absent AND D_PIGM_DOTS IS present AND ASYMMETRY IS sym_change THEN DIAGNOSIS IS Benign_nev
RULE 4 IF TDS >= 3.310 AND TDS < 4.080 AND C_BLUE IS absent THEN DIAGNOSIS IS Benign_nev	**RULE 7** IF TDS >= 3.310 AND TDS < 4.080 AND C_BLUE IS absent AND ASYMMETRY IS sym_change THEN DIAGNOSIS IS Benign_nev	**RULE 10** IF TDS >= 3.310 AND TDS < 4.080 AND C_BLUE IS absent AND ASYMMETRY IS sym_change AND C_d_BROWN IS present THEN DIAGNOSIS IS Benign_nev	**RULE 15** IF C_BLUE IS absent AND TDS >= 4.75 AND TDS < 5.35 AND C_WHITE IS present AND C_BLACK IS absent AND D_PIGM_NETW IS absent THEN DIAGNOSIS IS Benign_nev
RULE 19 IF TDS >= 4.850 AND TDS < 5.620 AND C_BLUE IS absent THEN DIAGNOSIS IS Suspicious	**RULE 28** IF TDS >= 4.850 AND TDS < 5.620 AND C_BLUE IS absent AND ASYMMETRY IS 1_axial_as THEN DIAGNOSIS IS Suspicious	**RULE 39** IF TDS >= 4.850 AND TDS < 5.620 AND C_BLUE IS absent AND ASYMMETRY IS 1_axial_as AND C_d_BROWN IS present THEN DIAGNOSIS IS Suspicious	**RULE 24** IF D_STREAKS IS absent AND D_PIGM_DOTS IS absent AND C_BLUE IS present THEN DIAGNOSIS IS Blue_nevus

also observed that strongest rules generated from the first network (α=1), became a base for rules generated from other networks (α=30 and 60).
The strongest decision rules generated by **LERS** system present more detailed study of cases, mainly because they contained greater number of descriptive attributes than respective belief rules, moreover, the attributes used were often completely different. Only the attribute <*color blue*> was common for both sets of rules. Considering the overall results of classification (Table 2) the best results were obtained for belief networks with α=1 or α=30. Expanding the set of rules (α=60) with another attribute — <*color dark brown*> — did not improve the classification process. It may be pointed out, that generalizing of the rule sets yielded positive effects in term of classification process. Results of the carried out research has also showed that belief rules (produced by ***Belief*SEEKER**) seemed to be some kind of generalization of rules developed by **LERS** system. In the future research, the detailed foundation of this interesting finding will be dealt with.

References

1. P. Błajdo, J.W. Grzymała-Busse, Z.S. Hippe, M. Knap, T. Marek, T. Mroczek, M. Wrzesień: *A suite of machine learning tools for knowledge extraction from data*, **In:** R. Tadeusiewicz, A. Ligęza, M. Szymkat (Eds.), Computer Methods and Systems in Scientific Research, Edition of "Oprogr. Naukowe", Cracow 2003, pp. 479–484 (in Polish).
2. M.R. Chmielewski, J.W. Grzymała-Busse, Neil W. Peterson, S. Than: *The Rule Induction System LERS — A Version for Personal Computers*, **Foundations of Computing and Dec. Sciences 1993** (18, No 3–4) pp. 181–211.
3. Z. Pawlak: *Knowledge and rough set*, **In**: W. Traczyk (Eds.), Problem of artificial intelligence, Wiedza i życie — Warsaw 1995, pp. 9–21 (in Polish).
4. T. Mroczek, J.W. Grzymała-Busse, Z.S. Hippe: *Rules from belief networks: A Rough Set Approach*, **In**: S. Tsumoto, R. Słowiński, J. Komorowski, J.W. Grzymała-Busse (Eds.), Rough Sets and Current Trends in Computing, Springer-Verlag, Uppsala, Sweden 2004, pp.483–487
5. D. Heckerman: *A Tutorial on Learning Bayesian Networks*, Technical report MSR-TR-95-06. heckerman@microsoft.com.
6. Z.S. Hippe: *Design and Application of New Knowledge Engineering Tool for Solving Real World Problems*, **Knowledge-Based Systems 9**(1996)509–515.
7. J.W. Grzymała-Busse: *A New Version of the Rule Induction System **LERS***, **Fundamenta Informaticae 31**(1997)27–39.
8. J.W. Grzymała-Busse, W. Ziarko: *Data Using Based on Rough Sets*, **In**: J.Wang (Eds.), Data Mining: Opportunities and Challenges, Idea Group Publishing, Hershey 2003, pp. 142–173.
9. A. Alvarez, S. Bajcar, F.M. Brown, J.W. Grzymała-Busse, Z.S. Hippe: *Optimization of the ABCD Formula Used for Melanoma Diagnosis*, **In**: M.A. Kłopotek, S.T. Wierzchoń, K. Trojanowski (Eds.), Advances in Soft Computing, Springer-Verlag, Heidelberg 2003, pp.233–240.

iMatch — A New Matchmaker For Large Scale Agent Applications

Ashwin Bhat Gurpur

Concordia University, Montreal, Canada

Abstract. This paper discusses a new design for the matchmaker used in DECAF [1], called iMatch. The paper discusses how we have successfully used the Cerebellar Model Arithmetic Computer - CMAC [4-6] algorithm in the matchmaking algorithm to create a robust high speed matchmaker, which can be used for developing any large scale agent application having rigid time constraints.

1 Introduction

DECAF provides us with a good framework for developing Agent systems. But the DECAF matchmaker[1] executes the matchmaking algorithm every time it receives a service request from an agent. This slows down applications when a large number of agents/agent services are used. We have suggested a way here by which the matchmaker learns to select an agent using the CMAC algorithm without the need to run the matchmaking algorithm each time such a service is requested.

2 Use of CMAC with the matchmaker

Inputs to the current DECAF matchmaker algorithm are the number of keywords in the query, threshold percentage, number of agents required and output is a list of tuples (agent name, percentage match). We saw that the complexity of this algorithm is $O(X[YZ + Z + log_2 X])$, where X is the total number of ads at the time of executing the algorithm, Y is the total number of keywords in the query and Z is the total number of keywords in all the ads. We see that this algorithm is very slow for a large X,Y or Z. Hence we have developed the iMatch matchmaker, which learns to choose an agent to provide a particular service without the need to run this algorithm everytime, thus saving a lot of processing time for each service query.

It may seem to the reader that this problem could be solved with a look up table. Look up tables work only when we have a definite input for which we have a definite output. But in this case, the queries are random combination of keywords. The number of keywords that will be used is not a finite value. Hence we need a technique by which the matchmaker learns to handle the random combination of known keywords as well as unknown keywords, present in a query. Also, for an extremely large number of possible queries, a

look up table is not practical. Hence, we use the CMAC machine-learning algorithm to enable the matchmaker to make reliable decisions about choosing the right agent.

3 Implementation of iMatch

We have used the CMAC algorithm implemented by the robotics and vibration control laboratory, University of New Hampshire for this project. We have converted this to Java code to suit easy usage in the DECAF matchmaker (DECAF has been written in Java). CMAC code was included in the mmMakeMatch.java file provided by DECAF. The CMAC implementation has the training code as well as the testing code to test if the error is below a particular threshold. The CMAC code iteratively trains and tests till the percentage error goes below a pre—specified threshold percentage. We had set the threshold percentage to 1%. So in our case, when CMAC trains to less than 1% error, it stops. Training was done as told in the next section. The following points were used during the implementation of iMatch:

The keywords of the query are converted to numbers as follows: each letter in the alphabet has a two digit number associated with it for example A is 01, B is 02 and so on till Z is 26. For example if query is oil, its numerical value is150912. All keywords in both the advertisements and queries are converted to such numbers for training and querying CMAC. The CMAC must be trained only after the matchmaker stabilizes after the agents have made most of the advertisements. The training manager included with the matchmaker decides this and initiates the training. The training manager keeps track of number of advertisements per minute and when it reduces to a minimum, it initiates the training We train the CMAC using the technique described in the next section. Once the CMAC is trained sufficiently, it selects the agent, which provides the required service in constant time, without the need to run the matchmaking algorithm If a few agents add their advertisements with the matchmaker after the CMAC is trained, then the matchmaker algorithm is run only on its keywords and compared with the CMAC result to make a decision instead of running the algorithm with all the keyword of all the agents In case an agent withdraws its advertisement and CMAC comes up with its name as the selected agent, then without the need to run the algorithm, the matchmaker decides that no agent matches the request. Without CMAC, it would have to compare all keywords of all agents to make this decision If a number of agents start advertising, the training manager initiates the training of CMAC again. The training manager has a threshold number of agents that can advertisement without the need to initiate another training During a training, if a query arrives, only for such queries, the normal matchmaker algorithm is run for all the advertisements.

4 Training iMatch

For testing purposes, we have limited the number of keywords in the query to be two. We must use each of the keywords in the advertisements and also combinations of each of these with a maximum of two in each combination as inputs and use the matchmaker algorithm to find the output to train iMatch. Consider three advertisements: *car repair*, *radio repair*, *watch repair*

For each keyword like *radio*, *watch*, *car* which appears in the ads, and also for their combinations like *radio repair* we must use the matchmaking algorithm to select the agent. For example, the agent for keyword *radio* is 2, which is got by running the matchmaking algorithm. This agent output and the input keywords must be used to train iMatch.

Another way to train iMatch is to use real queries and outputs to train iMatch rather than using all possible keyword combinations. This avoids the combinatorial problem for a large number of keywords K with M keyword subsets. After such training, even if a new query arrives, iMatch decides the right response though such a query is used for the first time. While it is being trained, iMatch performs like a normal matchmaker since training and query processing are done at the same time. But once it is trained, we see that it performs better than the normal matchmaker as explained in the next section.

5 Test results

Although iMatch is still under development, all tests based on our new technique have given good results. We have simulated a scenario where a large number of Agents participate in advertisements and a large number of Agents query the matchmaker and observed the results, which we discuss below:

All implementations are in Java and the simulations were done on Windows 2000 OS running on Intel Pentium-4 2.4 GHz processor with 520 MB RAM. No other applications were running during the simulation. All agents and matchmaker were run in the same computer: The agents were created using the DECAF framework. All agents were created using a simple DOS batch file, which iteratively made the required number of copies of the agents plan file [1]. Each agent has two files associated with it. For example, the first agent has 1.lsp and 1.plan, the next one has 2.lsp and 2.plan and so on. Once the batch file completed this, the ANS server and DECAF matchmaker was initiated. When they were up and running, another DOS batch file, continuously start these agents using the standard DECAF method of initiating an agent. Here, the agents only job is to advertise with a set of keywords. The next agent advertises only after the first one receives a confirmation that its advertisements were successful. Each agent was programmed to write a time stamp to a separate file when it started. Thus when all agents finish advertising, we can use the timestamps of the first and last agent to know how much

time the whole process took. A similar batch file created a thousand agents whose job was to produce a random query and get a response for it. Time for all of these to query and to get a result was noted down as earlier. The whole procedure of advertising and querying was repeated for iMatch and normal DECAF matchmaker separately to compare their performance.

The results are shown in Fig. 1. Currently iMatch returns the name of one agent for each query.

Trial 1: 54 agents were made to advertise to the DECAF matchmaker. After the advertisements were completed, the thousand agents were made to query it as explained earlier, each with one query. Time taken for these thousand agents to query and get a response was noted. We repeated this five times. The average time it took was 1032234 milliseconds. The same experiment with the same number of agents was conducted on iMatch. It took 1030302 ms.

Trial 2: this time, 9999 copies of the agent were made to advertise but the same thousand agents were used for querying the matchmaker. For DECAF matchmaker, time for 5 trials averaged to 1119549 ms. For iMatch, its 5 trial average was 1031243 ms, which is almost same as what it took when there were only 54 agents.

Trial 3: In this trial, 20413 copies of the agent were made to advertise but the same thousand agents were used for querying the matchmaker. For DECAF matchmaker, time for 5 trials averaged to 1150495 ms. The 5 trial average in case of iMatch was 1030275 ms, which is almost the same as earlier two trials.

6 Analysis of the Results and conclusions

Initially when there were only 54 agents, both iMatch and DECAF matchmaker had almost the same performance since the time taken to process 108 keywords from these agents was a trivial task for the DECAF matchmaker. But when the number of agents was increased to about ten thousand, the twenty thousand keywords took some time to process while CMAC maintained the same time to give the output. When the number of agents was increased to about twenty thousand with forty thousand keywords, the DECAF matchmaker showed a real increase in time taken to provide the output, while iMatch still took the same time it took for 54 agents. From these experiments we can conclude that what ever be the number of agents, once iMatch is trained, it uses constant time to provide us with the answers to queries whereas the time taken by DECAF matchmaker for the same number of queries increases as the number of agents increases.

We will see similar results if each of the agents advertise more than one advertisements. This would increase the DECAF matchmakers response time but iMatch would continue to provide constant response time. This means that in future if we are to develop large-scale agent applications, where thou-

sands of software agents work in different parts of the world contacting a single matchmaker, then using iMatch would be more suitable than the currently used Matchmakers. The training manager used in iMatch is being improved to reduce the training time.

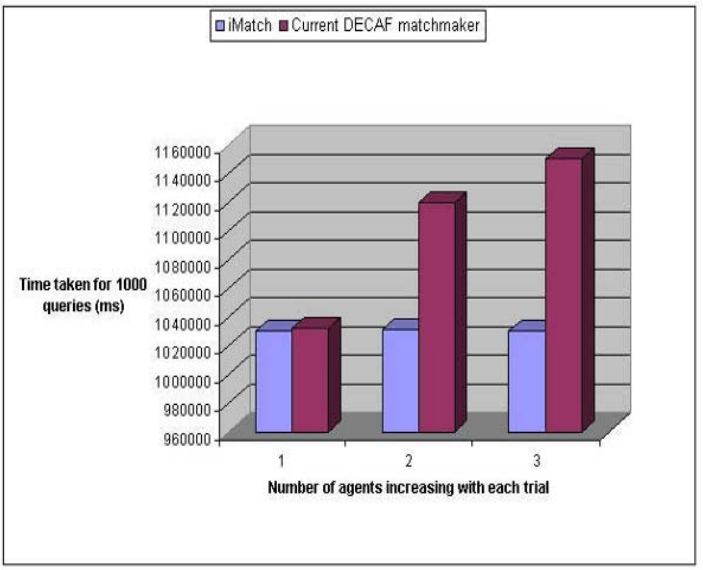


Fig. 1. Comparing the performance of iMatch with DECAF matchmaker

References

1. Graham J. R., Decker Keith. S. et al. (2003) DECAF - A Flexible Multi Agent System Architecture, Autonomous Agents and Multi-Agent Systems (AAMAS 2003) conference proceedings.
2. Wei Chen, Keith Decker S. (July 2002) Applying Coordination Mechanisms for Dependency Relationships among Multiple Agents. Autonomous Agents and Multi-Agent Systems (AAMAS 2003) conference proceedings.
3. John Graham (March 2000) Real-Time Scheduling for Distributed Agents. AAAI-Spring Symposium on Real-Time Autonomous Systems, Palo Alto, CA.
4. Miller et al. (Oct. 1990) CMAC: An Associative Neural Network Alternative to Backpropagation. Proceedings of the IEEE, Special Issue on Neural Networks **78**, 1561–1567
5. Albus James. S. (1975) A New Approach to Manipulator Control: The Cerebellar Model Articulation Controller (CMAC). Journal of Dynamic Systems, Measurement and Control, American Soc. of Mechanical Engineers.
6. Filson Glanz H. (Aug. 1991) An overview of the CMAC neural network. IEEE Conf. on Neural Networks for Ocean Engineering.

Machine Learning and Statistical MAP Methods

Mark Kon[1], Leszek Plaskota[2], and Andrzej Przybyszewski[3]

[1] Boston University, Boston, MA 02215, USA
[2] Warsaw University, 02-097 Warsaw, Poland
[3] McGill University, Montreal, Quebec H3A 1B1, Canada

Abstract. For machine learning of an input-output function f from examples, we show it is possible to define an a priori probability density function on the hypothesis space to represent knowledge of the probability distribution of f, even when the hypothesis space H is large (i.e., nonparametric). This allows extension of maximum a posteriori (MAP) estimation methods nonparametric function estimation. Among other things, the resulting MAPN (MAP for nonparametric machine learning) procedure easily reproduces spline and radial basis function solutions of learning problems.

1 Introduction

In machine learning there are a number of approaches to solving the so-called function approximation problem, i.e., learning an input-output function $f(\mathbf{x})$ from partial information (examples) $y_i = f(\mathbf{x}_i)$ (see [6,9]). This is also the regression problem in statistical learning [12,8]. The problem has evolved from a statistical one dealing with low dimensional parametric function estimation (e.g., polynomial regression) to one which tries to extrapolate from large bodies of data an unknown element f in a nonparametric (large or infinite dimensional) hypothesis space H of functions. Recent nonparametric approaches have been based on regularization methods [12], information-based algorithms [9,10], neural network-based solutions [6], Bayesian methods [13], data mining [2], optimal recovery [5], and tree-based methods [3].

We will include some definitions along with a basic example. Suppose we are developing a laboratory process which produces a pharmaceutical whose quality (as measured by the concentration y of the compound being produced) depends strongly on a number of input parameters, including ambient humidity x_1, temperature x_2, and proportions $x_3, \ldots, x_n$ of chemical input components. We wish to build a machine which takes the above input variables $\mathbf{x} = (x_1, \ldots, x_n)$ and whose output predicts the desired concentration y. The machine will use experimental data points $y = f(\mathbf{x})$ to learn from previous runs of the equipment. We may already have a prior model for f based on simple assumptions on the relationships of the variables.

With an unknown i-o function $f(x)$, and examples $Nf \equiv (f(\mathbf{x}_1), \ldots, f(\mathbf{x}_n)) = (y_1, \ldots, y_n) = \mathbf{y}$, we seek an algorithm ϕ which maps information Nf into

the best estimate $\phi(Nf)$ of f. The new algorithm presented here (MAP for nonparametric machine learning, or MAPN) is an extension of methods common in parametric (finite dimensional) learning. In the approach, an a priori distribution P (representing prior knowledge) on the hypothesis space H of functions is given, and the function is learned by combining data Nf with a priori information μ.

One possible a posteriori estimate based on Nf is the conditional expectation $E(\mu|Nf)$ [7,10,9], which can be done in high (nonparametric) and low (parametric) dimensional situations. In low dimensions an easier estimation procedure is often done using maximum a posteriori (MAP) methods, in which a density function $\rho(x)$ of the probability measure P is maximized. In data mining on the other hand, a full (nonparametric) f must be estimated, and its infinite dimensional hypothesis space H does not immediately admit MAP techniques. We show that in fact densities $\rho(f)$ exist and make sense even for nonparametric problems, and that they can be used in the same way as in parametric machine learning. Given information $\mathbf{y} = Nf$ about an unknown $f \in H$, the MAPN estimate is simply $\widehat{f} = \arg\max_{f \in N^{-1}\mathbf{y}} \rho(f)$. Density functions $\rho(f)$ have some important advantages, including ease of use, ease of maximization, and ease of conditioning when combined with examples $(y_1, \ldots, y_n) = Nf$ (see examples in Section 3). Since they are also likelihood functions (representing our intuition of how "likely" a given guess f_1 is as compared to another f_2), they can be modified on a very intuitive basis (see also, e.g., [1]). For example, if we feel that we want our a priori guess at the unknown f to be smoother, we can weight the density function $\rho(f)$ (for the measure μ) with an extra factor $e^{-||Af||^2}$, with A a differential operator, in order to give less weight to "nonsmooth" functions with high values of $||Af||$. By the Radon-Nikodym theorem we will be guaranteed that the new (intuitively motivated) density $\rho(f)e^{-||Af||^2}$ will be the density of a bona fide measure ν, with $d\nu = e^{-||Af||^2} d\mu$.

2 The maximization algorithm

Let P be a probability distribution representing prior knowledge about $f \in H$, with the hypothesis space H initially finite dimensional. Let λ be uniform (Lebesgue) measure on H, and define the probability density function (pdf) of P (assuming it exists) by

$$\rho(f) = \frac{dP}{d\lambda}. \tag{1}$$

It is possible to define ρ alternatively up to a multiplicative constant through

$$\frac{\rho(f)}{\rho(g)} = \lim_{\epsilon \to 0} \frac{P(B_\epsilon(f))}{P(B_\epsilon(g))}. \tag{2}$$

That is the ratio of densities of two measures at f equals the ratio of the measures of two small balls there. Here $B_\epsilon(f)$ is the set of $h \in H$ which are within distance ϵ from f. Though definition (1) fails to extend to (infinite dimensional) function spaces H, definition (2) does. Henceforth it will be understood that a density function $\rho(f)$ is defined only up to a multiplicative constant (note (2) only defines ρ up to a constant). The MAP algorithm ϕ maximizes $\rho(f)$ subject to the examples $\mathbf{y} = Nf$. Thus (2) extends the notion of a density function $\rho(f)$ to a nonparametric H. Therefore it defines a likelihood function to be maximized a posteriori subject to $\mathbf{y} = Nf$. It follows from the theorem below that this in fact can be done for a common family of a priori measures [10]. For brevity, the proof of the following theorem is omitted.

Theorem 1. *If μ is a Gaussian measure on the function space H with covariance C, then the density $\rho(f)$ as defined above exists and is unique (up to a multiplicative constant), and is given by $\rho(f) = e^{-\langle f, Af\rangle}$, where $A = C^{-1/2}$.*

Under the assumption of no or negligible error (we will later not restrict to this), the MAPN estimate of f given data $Nf = \mathbf{y}$ is $\phi(Nf) = \widehat{f} = \arg\max_{Nf=y} \rho(f)$. More generally, these ideas extend to non-Gaussian probability measures as well; the theorems are omitted for brevity.

3 Applications

We consider an example involving a financial application of the MAPN procedure for incorporating a priori information with data. We assume that a collection of 30 credit information parameters are collected from an individual borrower's credit report by a large bank. These include total debts, total credit, total mortgage balances, and other continuous information determined earlier to be relevant by a data mining program. We wish to map this information into a best estimated debt to equity ratio two years hence. A (limited) database of past information is available, containing recent information (as of the last year) on debt to equity ratios, together with data on the $d = 30$ parameters of interest We wish to combine this information with an earlier estimate (taken 4 years earlier), consisting of a function $f_0 : J^{30} \to I$ from the (normalized) credit parameters into a debt to equity ratio (also normalized), where $J = [-1, 1]$ and $I = [0, 1]$. In order to avoid boundary issues, we will extend f_0 smoothly to a periodic map $K^{30} \to I$, where $K = [-1.5, 1.5]$, with -1.5 identified with 1.5, so that smooth functions on K must match (as well as all their derivatives) at the endpoints ± 1.5. Similarly, a function on the torus K^{30} is smooth if it is periodic and smooth everywhere, including on the matching periodic boundaries. The purpose of this is to expand a differentiable function f on K^{30} in a Fourier series.

On the belief that the current form f: $K^{30} \to I$ of the desired function is different from the (a priori) form f_0 earlier estimated, we make the prior assumption that there is a probability distribution P for the sought (currently

true) f_1 centered at the earlier estimate f_0, having the form of a Gaussian on H, the set of square integrable functions from K^{30} to I. This a priori measure P favors deviations from f_0 which are sufficiently smooth to be well-defined pointwise (but not too smooth) and small, and so P is given the form of a Gaussian measure with a covariance C defined on the orthonormal basis (here a is a normalization constant) $\{b_{\mathbf{k}} = ae^{\frac{2}{3}\pi i\mathbf{x}\cdot\mathbf{k}}\}_{\mathbf{k}\in\mathbf{Z}^{30}}$ ($\mathbf{Z}$ is the integers) for $L^2(K^{30})$ by $C(e^{\frac{2}{3}\pi i\mathbf{x}\cdot\mathbf{k}}) = \frac{1}{(1+|\mathbf{k}|)^{31}}e^{\frac{2}{3}\pi i\mathbf{x}\cdot\mathbf{k}}$ with $\mathbf{k} = (k_1, \ldots, k_{30})$ a multiinteger, and $\mathbf{x} \in K^{30}$ (note that P forms a Gaussian measure essentially concentrated on functions $f \in L^2(K^{30})$ with 15.5 square integrable derivatives, which guarantees that such functions' pointwise values are well-defined, since $15.5 > \frac{d}{2}$). We uniquely define the operator A by $C = A^{-2}$; A satisfies $A(e^{\frac{2}{3}\pi i\mathbf{x}\cdot\mathbf{k}}) = |\mathbf{k}|^{31/2}e^{\frac{2}{3}\pi i\mathbf{x}\cdot\mathbf{k}}$. To simplify notation and work with a Gaussian centered at 0, we denote the full new i-o function we are seeking by $f_1(\mathbf{x})$. We will seek to estimate the change in the i-o function, i.e., $f = f_1 - f_0$. With this subtraction the function f we seek is centered at 0 and has a Gaussian distribution with covariance C. Our new i-o data are $y_i = f(\mathbf{x}_i) = f_1(\mathbf{x}_i) - f_0(\mathbf{x}_i)$, where $f_1(\mathbf{x}_i)$ are the measured debt to equity ratios, and are immediately normalized by subtracting the known $f_0(\mathbf{x}_i)$. Thus y_i sample the change $f(\mathbf{x}_i)$ in the i-o function.

We first illustrate the algorithm under the hypothesis that data $y_i = f(\mathbf{x}_i)$ are exact (the more realistic noisy case is handled below). In this exact information case the MAPN algorithm finds the maximizer of the density $\rho(f) = e^{-\|Af\|^2}$ (according to Theorem 1) restricted to the affine subspace $N^{-1}(\mathbf{y})$. This is equivalent to minimizing $\|Af\|$ subject to the constraint $\mathbf{y} = Nf = (f(\mathbf{x}_1), \ldots, f(\mathbf{x}_n))$, (where $f(\mathbf{x}_i)$ is the outcome for example $\mathbf{x}_i$), which yields the spline estimate

$$\widehat{f} = \sum_{j=1}^{n} c_j CL_j, \tag{3}$$

where for each j, the linear functional $L_j(f) = f(\mathbf{x}_j)$, and where $c_i = S\mathbf{y}$ is determined from $\mathbf{y}$ by a linear transformation S (see [9] for the construction of such spline solutions). We have (here δ denotes the Dirac delta distribution) $CL_j = C\delta(\mathbf{x} - \mathbf{x}_j) = C\left(a^2\sum_k e^{\frac{2}{3}\pi i\mathbf{k}\cdot(\mathbf{x}-\mathbf{x}_j)}\right) = \sum_k a^2 Ce^{\frac{2}{3}\pi i\mathbf{k}\cdot(\mathbf{x}-\mathbf{x}_j)}$ $= \sum_k \frac{a^2}{|\mathbf{k}|^{31}}e^{\frac{2}{3}\pi i\mathbf{k}\cdot(\mathbf{x}-\mathbf{x}_j)} = G(\mathbf{x} - \mathbf{x}_j)$ is a radial basis function (equivalently, a B-spline) centered at $\mathbf{x}_j$. So the estimated regression function is $\widehat{f} = \sum_{j=1}^n c_j G_j(\mathbf{x} - \mathbf{x}_j) = \sum_{j=1}^n c_j \sum_{\mathbf{k}} \frac{a^2}{|\mathbf{k}|^{31}}e^{\frac{2}{3}\pi i\mathbf{k}\cdot(\mathbf{x}-\mathbf{x}_j)}$. By comparison, a standard algorithm for forming a (Bayesian) estimate for f under the average case setting of information-based complexity theory using information $Nf = (y_1, \ldots, y_n)$ is to compute the conditional expectation $\phi(Nf) = E_\mu(f|N(f) = (y_1, \ldots, y_n))$. For a Gaussian measure this expectation is known also to yield the well-known spline estimate (3) for f [9,10]. The regularization al-

gorithm [12] can be chosen to minimize the norm $\|Af\|$ subject to $Nf = \mathbf{y}$, again yielding the spline solution (3).

Noisy information: It is much more realistic, however, to assume the information $Nf = (y_1, \ldots, y_n)$ in the above example is noisy, i.e., that if $f = f_1 - f_0$ is the sought change in the 2 year debt to equity ratio, then $y_i = f(\mathbf{x}_i) + \epsilon_i$ where ϵ_i is a normally distributed error term. In this case the MAP estimator is given by $\widehat{f} = \arg\sup_f \rho(f|\mathbf{y})$. However, note that (as always, up to multiplicative constants) $\rho(f|\mathbf{y}) = \frac{\rho_{\mathbf{y}}(\mathbf{y}|f)\rho(f)}{\rho_{\mathbf{y}}(\mathbf{y})}$ so that if the pdf of $\epsilon = (\epsilon_1, \ldots, \epsilon_n)$ is Gaussian, i.e., has density $\rho_\epsilon(\epsilon) = K_1 e^{-\|B\epsilon\|^2}$ with B linear and K a constant, then $\rho(f|\mathbf{y}) = K_2 \frac{e^{-\|B(Nf-\mathbf{y})\|^2} e^{-\|Af\|^2}}{\rho_{\mathbf{y}}(\mathbf{y})} = K_3 e^{-\|B(Nf-\mathbf{y})\|^2 - \|Af\|^2}$ where K_3 can depend on the data $\mathbf{y} = (y_1, y_2, \ldots, y_n)$. MAP requires that this be maximized, so

$$\widehat{f} = \arg\min \|Af\|^2 + \|B(Nf - \mathbf{y})\|^2. \tag{4}$$

This maximization can be done using Lagrange multipliers, for example. This again is a spline solution for the problem with error [7]. In addition, again, the minimization of (4) is the same as the regularization functional minimization approach in statistical learning theory [12]. It yields a modified spline solution as in (3), with modified coefficients c_j.

References

1. N. Friedman and J. Halpern (1996) Plausibility measures and default reasoning. In Thirteenth National Conf. on Artificial Intelligence (AAAI).
2. T. Hastie, R. Tibshirani and J. Friedman (2001) The elements of statistical learning: data mining, inference and prediction. Springer-Verlag.
3. M. Jordan. and M. Meila (2000) Learning with mixtures of trees. Journal of Machine Learning Research **1**, 1-48.
4. M. Kon (2004) Density functions for machine learning and optimal recovery. preprint.
5. C. Micchelli and T. Rivlin (1985) Lectures on optimal recovery. Lecture Notes in Mathematics, 1129, Springer-Verlag, Berlin, 1985, 21-93.
6. T. Mitchell (1997) Machine Learning. McGraw-Hill, NY.
7. L. Plaskota (1996) Noisy Information and Complexity. Cambridge Univ. Press.
8. T. Poggio and C. Shelton (1999) Machine Learning, Machine Vision, and the Brain. The AI Magazine **20**, 37-55.
9. J. Traub, G. Wasilkowski, and H. Wozniakowski (1988), Information-Based Complexity. Academic Press, Boston.
10. J. F. Traub and A. Werschulz (2001) Complexity and Information. Cambridge University Press, Cambridge.
11. J. Traub and H. Wozniakowski (1980) A General Theory of Optimal Algorithms. Academic Press, New York.
12. V. Vapnik (1998) Statistical Learning Theory. Wiley, New York.
13. G. Wahba (1999) Support vector machines, reproducing kernel Hilbert spaces and the randomized GACV. in B. Schoelkopf, C. Burges & A. Smola, eds, Advances in Kernel Methods Support Vector Learning, MIT Press, 69-88.

Autocovariance Based Weighting Strategy for Time Series Prediction with Weighted LS-SVM

Paweł Majewski

Gdansk University of Technology, Narutowicza 11/12, Gdańsk, Poland

Abstract. Classic kernel methods (SVM, LS-SVM) use some arbitrarily chosen loss functions. These functions equally penalize errors on all training samples. In problem of time series prediction better results can be achieved when the relative importance of the samples is expressed in the loss function. In this paper an autocovariance based weighting strategy for chaotic time series prediction is presented. Proposed method can be considered a way to improve the performance of kernel algorithms by incorporating some additional knowledge and information on the analyzed learning problem.

1 Introduction

The classic kernel methods (SVM, LS-SVM) use some arbitrarily chosen loss functions. These functions equally penalize errors on all training samples. Therefore all training examples are considered equally significant. However, in some situations, for instance in prediction of chaotic time series, better results can be achieved when some examples are considered more significant then others. In this paper we propose the autocovariance based weighting strategy for time series prediction with Weighted LS-SVM. The method allows to build a data driven loss function that automatically adapts to the learning problem.

2 Kernel Based Learning Algorithms

The kernel based learning algorithm [5] in the case of function estimation can be formulated as a problem of finding a function $f(x)$ that best "fits" given examples (x_i, y_i), $i = 1 \dots N$. To measure the quality of fitting one can introduce a loss function $V(y, f(x))$. The loss function can be any function that expresses the difference between obtained and desired value. In practice convex loss functions are used. The problem of finding the appropriate function is then equivalent to minimizing the functional: $I = \frac{1}{N} \sum_{i=1}^{N} V(y_i, f(x_i))$ and is usually reffered as *Empirical Risk Minimization* (ERM). In general ERM problem is ill-posed, depending on the choice of the hypothesis space. Therefore instead of minimizing ERM one can minimize its regularized version. Many techniques developed for solving ill-posed problems are possible but

the classic approach involves Tikhonov [8] regularization. Following Tikhonov one obtains:

$$I_{reg} = \frac{1}{N}\sum_{i=1}^{N} V(y_i, f(x_i)) + \gamma \|f\|_k^2 \tag{1}$$

where γ is a positive real regularization parameter and $\|f\|_k^2$ is a norm in Reproducing Kernel Hilbert Space (RKHS) defined by the chosen kernel function k. The second term in (1), usually called regularizator, corresponds to *Structural Risk Minimization* (SRM). The role of the regularizator is twofold. Firstly, it forces uniqueness and smoothness of the solution [5] and controls the complexity of the model with the regularization parameter γ. Additionally, the regularizator restricts the hypothesis space where we seek for the function f to RKHS induced by the selected kernel function k. The kernel function k can be any positive definite function that satisfies Mercer's conditions [9].

Regardless of the the loss function used, minimalization of (1) yields the solution of the form [5]: $f(\bullet) = \sum_{i=1}^{N} \alpha_i k(x_i, \bullet)$, where α_i are coefficients found during learning process. According to the choice of a particular loss function $V(y, f(x))$ one can obtain several learning algorithms [9].

2.1 Weighted LS-SVM

In the classic LS-SVM approach [6] errors on all training samples are equally penalized. In some situations, for instance in prediction of chaotic time series, better results can be achieved when some examples are considered more significant then others. Their significance can be expressed with the weights of Weighted LS-SVM (WLS-SVM) [6]. They use the adaptable loss function $V_{WLS}(y, f(x), z) = z_i(y_i - f(x_i))^2$ where the importance of every training sample can be adjusted by a weight parameter z_i. The modified learning problem (1) becomes now:

$$I_{WLS} = \frac{1}{N}\sum_{i=1}^{N} z_i(y_i - f(x_i))^2 + \gamma \|f\|_k^2 \tag{2}$$

where z_i denotes the weight of the squared error corresponding to the given sample. The optimal coefficients $\boldsymbol{\alpha}$ can be found by solving:

$$\left(\boldsymbol{K} + \frac{N\gamma}{2}\boldsymbol{Z^{-1}}\right)\boldsymbol{\alpha} = \boldsymbol{y} \tag{3}$$

where $\boldsymbol{K} = \{k_{ij} : k_{ij} = k(x_i, x_j); i, j = 1 \ldots N\}$ is the kernel matrix and $\boldsymbol{Z} = diag([z_1 \ldots z_N])$ is a weight matrix.

Depending on the selection of the weighting strategy training errors can be penalized according to their significance.

3 Weighting Strategy for Chaotic Time Series Prediction

The proposed weighting strategy for chaotic time series prediction is based on short-term correlation of the predicted series. Research made on relational datasets has shown that classification performance improves when autocorrelation of the data is present [1]. The task was to investigate weather such phenomena is valid in problem of chaotic time series prediction. Basing on the assumption that learning on correlated data is easier [3] we designed a strategy that weights the importance of the learning samples in accordance to their correlation. The strategy highly penalizes errors on low correlated samples therefore it imposes learning of hard-to-learn samples. The proposed strategy can control the overfitting of the learned models with a designated parameter d.

To calculate the relative importance of the samples we shift the series and calculate the correlation between the original and shifted series in a fixed-sized sliding window. With help of correlation analysis we measure whether or not a change in the shifted time series will result in a change in the original series. Obtained measure is then expressed in the weights of the samples. A low correlation coefficient suggests that the relationship between the shifted and the original series is weak or opposite therefore error penalty on that sample should be higher. A high correlation indicates that the shifted series will change when the original changes and for that reason error on that sample doesn't have to to be penalized that high.

To compute the weights we use the common unbiased covariance estimate based on W samples of the target set $\boldsymbol{y}$ [4]. We use a fixed time shift s and fixed-sized sliding window W ($W << N$) and obtain a function of sample identifier i:

$$A_{\boldsymbol{yy}}(i) = \frac{1}{W-s}\left(\sum_{n=i+s}^{i+W-1} y_{n-s}y_n\right) - {\mu_y}^2 \tag{4}$$

where μ_y denotes mean value of the target values y_i in window W. The time series is assumed to be stationary at least in the selected window W.

The formula (4) can be used to compute covariance coefficients for samples $(1 \ldots N-W+1)$. For the ramaining samples coefficients of the $i = N-W+1$ sample are used. The computed coefficients are then normalized so that they fall in the interval $\langle 1, d\rangle$. The higher value of d, the more "loose" is the weighting strategy on highly correlated samples. Since high values of d can result in overfitting of the model the optimal value of the parameter should be found with some validation procedure, e.g. cross-validation.

Since to solve (3) one needs the the reverted elements $1/z_i$ of the weights matrix $\boldsymbol{Z}$ then the application of obtained coefficients can be direct. Smaller values of the normalized $A_{\boldsymbol{yy}}(i)$ correspond to higher penalties z_i and bigger coefficients correspond to lower weights z_i.

4 Experiments

To examine the performance of the strategy a well-known reference problem of Mackey-Glass (MG) time series prediction [2] was chosen. We used the MG time series obtained with the constants set to $a = 0.2$, $b = 0.1$ and $\tau = 17$. The inputs were prepared as described in [7]. The training set of 400 training examples was made by taking samples of t ranging in values from 196 to 595. The next 400 MG values were used as validation data. Trained models were evaluated on the test set of 100 following samples. The samples were shaded with zero-mean gaussian noise with standard deviation equal to 1/4 of the standard deviation of the original series. The optimal values of the parameters W, s and d were obtained by grid search. We used RBF kernels with bandwidth $\sigma^2 = 2$ in all models.

Additionally, the proposed method was analyzed on another reference time series used in The SantaFe Competition [10]. We chose the data set A (Laser generated data) for the tests. The time series is relatively easy and very predictable on the shortest time scales (relatively simple oscillations) but has global events that are harder to predict. The inputs were formed by taking $L = 50$ last samples spaced a period from each other $\boldsymbol{x_k} = [z(k-1), z(k-2), \ldots, z(k-L)]$ and the targets were chosen to be $y_k = z(k)$ so that a step ahead prediction was tested. The whole data set of 1050 samples was split into the training set of the first 400 samples, validation set of 450 following samples and the test set of the last 100 samples. The optimal values of the parameters W, s and d were obtained by grid search. We used RBF kernels with bandwidth $\sigma^2 = 182$ in all models.

Weighted LS-SVM outperformed classic LS-SVM in both MG and SanteFe problems (see Table 1 and Table 2). Test error was significantly lower when proposed weighting strategy was applied. Sparse WLS-SVM models also performed better then the unweighted LS-SVM. It appears that due to high noise WLS-SVM were more prune to overfitting then unweighted algorithm in the MG problem. In case of SantaFe time series the weighting strategy automatically selected the training samples close to the rapid decay of the oscillations that are hard-to-learn for other learning algorithms. This resulted in the improvement on the test set which consisted of samples forming a similar rapid decay (as in the original competition). Better performance was achieved at the price of higher training error because the algorithm became slightly more "loose" on the rest of the samples.

5 Conclusions

In this paper an autocovariance based weighting strategy for chaotic time series prediction was presented. Proposed method can be considered a way to improve the performance of kernel algorithms by incorporating some additional knowledge and information on the analyzed learning problem. The

Table 1. Training, validation and test Mean Squared Error in the experiments with Mackey-Glass time series (WLS-SVM parameters d = 3.1; W = 30; s = 9)

Method	Train	Validation	Test
LS-SVM (full solution)	9.89×10^{-4}	1.72×10^{-3}	7.41×10^{-4}
WLS-SVM (full solution)	6.23×10^{-4}	1.87×10^{-3}	7.13×10^{-4}
WLS-SVM (pruned 10% of SV)	7.87×10^{-4}	1.75×10^{-3}	7.36×10^{-4}
WLS-SVM (pruned 30% of SV)	8.84×10^{-4}	1.73×10^{-3}	7.14×10^{-4}
WLS-SVM (pruned 50% of SV)	1.01×10^{-3}	2.08×10^{-3}	9.83×10^{-4}

Table 2. Training, validation and test Mean Squared Error in the experiments with Santa Fe Laser time series (WLS-SVM parameters d = 5; W = 42; s = 11)

Method	Train	Validation	Test
LS-SVM	64.22	325.23	744.25
WLS-SVM (full solution)	103.04	318.86	707.75
WLS-SVM (pruned 10% of SV)	123.70	333.79	714.05
WLS-SVM (pruned 20% of SV)	133.44	350.23	744.36
WLS-SVM (pruned 30% of SV)	176.72	394.04	928.46

results of the experiments show that predictor constructed according to the described weighting strategy outperforms the classic kernel algorithms.

References

1. Jensen, D., Neville, J. (2003) Autocorrelation and Linkage Cause Bias in Evaluation of Relational Learners. LNCS **2583**, 101–116, Springer-Verlag
2. Mackey, M. C., Glass, L. (1977) Oscillation and chaos in physiological control systems. Science **197**, 287–289
3. Neville, J., Simsek, Ö., Jensen, D. (2004) Autocorrelation and Relational Learning: Challenges and Opportunities. Proceedings of SRL 2004
4. Niedzwiecki, M. (2000) Identification of Time-Varying Processes in Signal Processing, Wiley
5. Poggio, T., Smale, S. (2003) The Mathematics of Learning: Dealing with Data. Noticies of AMS 50, **5**, 537–544
6. Suykens, J. A. K., Van Gestel, T., De Brabanter, J. (2002) Least Squares Support Vector Machines. World Scientific
7. Svarer, C., Hansen, L. K., Larsen, J., Rasmussen, C. E. (1993) Designer networks for time series processing, Proceedings of the III IEEE Workshop on Neural Networks for Signal Processing, 78–87
8. Tikhonov, A. N., Arsenin, V. Y. (1977) Solution of Ill-posed problems. W. H. Winston, Washington
9. Vapnik, V. N. (1998) Statistical Learning Theory, Wiley, New York
10. Weigend, A. S., Gershenfeld, N. A. (1994) Time Series Prediction: Forecasting the Future and Understanding the Past, Addison-Wesley

Workflow Mining Alpha Algorithm — A Complexity Study

Bolesław Mikolajczak and Jian-Lun Chen

Computer and Information Science Department
University of Massachusetts Dartmouth
Dartmouth, MA 02747, USA
bmikolajczak@umassd.edu

Abstract. Workflow mining algorithms are used to improve and/or refine design of existing workflows. Workflows are composed of sequential, parallel, conflict and iterative structures. In this paper we present results of experimental complexity study of the alpha workflow mining algorithm. We studied time and space complexity as dependent on workflow's internal structure and on the number of workflow tasks.

1 Introduction

In workflow mining algorithm we do not know explicitly the workflow structure; however, we apply multiple cases to the workflow mining "black box" and we observe outcomes of each case. Workflow execution for all cases is recorded in a log file. Workflow mining alpha algorithm is based on workflow log treated as input [2,3]. Problem of workflow mining is similar to the problem of software process discovery [1]. We use Petri nets to specify workflows.

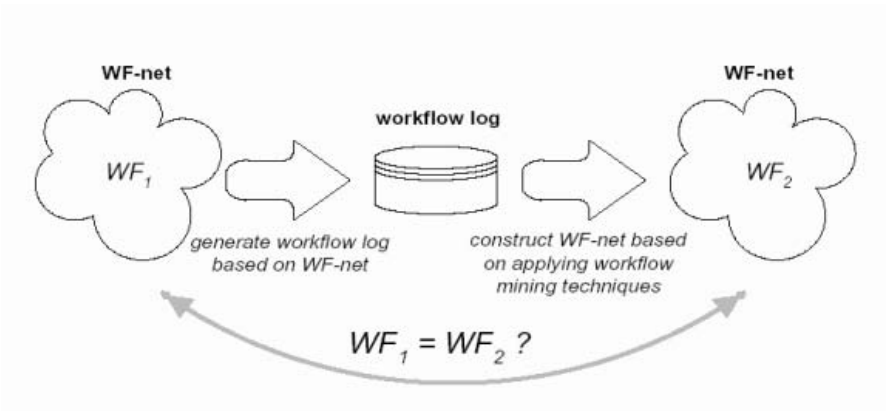


Fig. 1. An idea of workflow mining problem

Let $N' = (P, T, F, s)$ be a marked P/T-net. A transition $t \in T$ is live in (N, s)

iff for every reachable marking $s' \in [N, s >$ and $t \in T$, there is a reachable marking $s'' \in [N, s' >$ such that $(N, s'')[t >$. Let $N = (P, T, F)$ be a P/T net and t' an identifier not in $P \cup T$. N is a *workflow net* (WF-net) iff: it contains an input place i such that $\bullet i = \emptyset$, it contains an output place o such that $o\bullet = \emptyset$, and $N'=(P, T \cup \{t'\}, F \cup \{(o, t'), (t', i)\})$ is strongly connected. Let $N = (P, T, F)$ be a WF-net with input place i and output place o. N is *sound* iff the following properties are satisfied: workflow net $(N, [i])$ is *safe*, for any marking $s \in [N, [i] >$, $o \in s$ implies $s = [o]$, there is an option to complete: for any marking $s \in [N, [i] >$, $[o] \in [N, s >$, and absence of dead tasks: $(N, [i])$ contains no dead transitions. Let T be a set of tasks, $\sigma \in T^*$ is a workflow trace and $W \in P(T^*)$ is a workflow log, where $P(T^*)$ is the power set of T^*, i.e., $W \subseteq T^*$. Let W be a workflow log over T, i.e., $W \in P(T^*)$. Let $a, b \in T$: $a >_w b$ iff there is a trace $\sigma = t_1 t_2 t_3 ... t_{n+1}$ and $i \in \{1, ..., n-2\}$ such that $\sigma \in W$ and $t_i = a$ and $t_{i+1} = b$, $a \rightarrow_w b$ iff $a >_w b$ and $b \not>_w a$, $a \#_w b$ iff $a \not>_w b$ and $b \not>_w a$, and $a ||_w b$ iff $a >_w b$ and $b >_w a$. Let A be a set, $a \in A$, and $\sigma = a_1 a_2 .... a_n \in A^*$, a sequence over A of length n, *first*, *last* are defined as follows: $a \in \sigma$ iff $a \in \{a_1 a_2 .... a_n\}$, $first(\sigma) = a_1$, and $last(\sigma) = a_n$. Let $N = (P, T, F)$ be a sound WF-net, i.e., $N \in W$. W is a workflow log of N iff $W \in P(T^*)$ and every trace $\sigma \in W$ is a firing sequence of N starting in state $[i]$, i.e., $(N, [i])[\sigma >$. W is a workflow log of N iff for any workflow log W' of N: $>_{w'}, \subseteq_w$, and for any $t \in T$ there is $\sigma \in W$ such that $t \in \sigma$. Let $N = (P, T, F)$ be a sound WF-net, i.e., $N \in W$, and let α be a mining algorithm which maps workflow logs of N onto sound WF-nets, i.e., $\alpha : P(T^*) \rightarrow W$. If for any workflow log W of N the mining algorithm returns N (modulo renaming of places), then α is able to rediscover N. Let $N = (P, T, F)$ be a P/T-net with initial marking s. A place $p \in P$ is called implicit in (N, s) iff, for all reachable markings $s' \in [N, s >$ and transitions $t \in p\bullet$, $s' \geq \bullet t \backslash \{p\} \Rightarrow s' \geq \bullet t$. A WF-net $N = (P, T, F)$ is an SWF-net (Structured workflow net) iff for all $p \in P$ and $t \in T$ with $(p, t) \in F$:$\mid p\bullet \mid > 1$ implies $\mid \bullet t \mid = 1$ and for all $p \in P$ and $t \in T$ with $(p, t) \in F$: $\mid \bullet t \mid > 1$ implies $\mid p\bullet \mid = 1$ and there are no implicit places. If there is no choice and synchronization next to each other then there are no implicit places. This type of WF-net is called SWF-net. Let $N = (P, T, F)$ be a sound WF-net and let W be a workflow log of N. For any $a, b \in T : a \rightarrow_w b$ implies $a\bullet \cap \bullet b \neq \emptyset$. Let $N = (P, T, F)$ be a sound SWF-net and let W be a workflow log of N. For any $a, b \in T : a\bullet \cap \bullet b \neq \emptyset$ implies $a >_w b$. Based on SWF-net definition, if the workflow log is complete and a node resides between task A and B, then this implies that A comes before B. Let $N = (P, T, F)$ be a sound SWF-net and let W be a workflow log of N. For any $a, b \in T : a\bullet \cap \bullet b \neq \emptyset$ and $b\bullet \cap \bullet a = \emptyset$ implies $a \rightarrow_w b$. Let $N = (P, T, F)$ be a sound SWF-net such that for any $a, b \in T : a\bullet \cap \bullet b = \emptyset$ or $b\bullet \cap \bullet a = \emptyset$ and let W be a workflow log of N. Then if $a, b \in T$ and $a\bullet \cap \bullet b \neq \emptyset$ then $a \#_w b$. If $a, b \in T$ and $\bullet a \cap \bullet b \neq \emptyset$ then $a \#_w b$. If $a, b, t \in T$, $a \rightarrow_w t$, $b \rightarrow_w t$, and $a \#_w b$, then $a\bullet \cap b\bullet \cap \bullet t \neq \emptyset$. If

$a, b, t \in T$, $t \rightarrow_w a$, $t \rightarrow_w b$, and $a \#_w b$, then $\bullet a \cap \bullet b \cap t\bullet \neq \emptyset$. For SWF-net with a workflow log, we can derive the following implications:

1. if a transition comes after task A and task B then there is a choice between task A and task B.
2. if a transition comes before task A and task B then there is a choice between task A and task B.
3. if A causes T, B causes T, and there is a choice between A and B, then there is a transition that comes after task A and task B, but before task T.
4. if T causes A, T causes B, and there is a choice between A and B, then there is a transition that comes before task A and task B, but after task T.

For sound SWF-nets, $a||_w b$ implies $a\bullet \cap \bullet b = \bullet a \cap \bullet b = \emptyset$. Moreover, $a \rightarrow_w t$, $b \rightarrow_w t$, and $a\bullet \cap b\bullet \cap \bullet t = \emptyset$ implies $a||_w b$. Similarly, $t \rightarrow_w a$, $t \rightarrow_w b$, and $\bullet a \cap \bullet b \cap t\bullet = \emptyset$, also implies $a||_w b$.

2 Alpha Workflow Mining Algorithm

We make two assumptions: workflow net is a SWF net and workflow log is complete. Let W be a workflow log over T. The mining algorithm constructs a net (P_w, T_w, F_w). Knowing T_w, it is possible to find initial transition T_i and final transition T_o. Besides the input place i_w and the output place o_w, we have to find other places in the form $p(A, B)$, i.e. place between transition A and B. The subscript refers to the set of input and output transitions, i.e., $\bullet p(A, B) = A$ and $p(A, B)\bullet = B$. A place is added in-between task A and task B iff $A \rightarrow_w B$. For this purpose the relations X_w and Y_w are constructed. $(A, B) \in X_w$ if there is a causal relation from each member of A to each member of B and there exist choices between members of A or members of B, the members of A or B never occur next to one another. Relation Y_w is derived from X_w by taking only the largest elements with respect to set inclusion. Let us consider workflow log: *case 1*: A, C, D, E; *case 2*: A, D, C, E; *case 3*: B, F, G, H, I, J, K; *case 4*: B, F, G, I, H, J, K; *case 5*: B, F, G, I, J, H, K; *case 6*: B, F, I, J, G, H, K.; *case 7*: B, F, I, G, H, J, K; *case 8*: B, F, I, G, J, H, K.

1. T_w = {ACDE, ADCE, BFGHIJK, BFGIHJK, BFGIJHK, BFIJGHK, BFIGHJK, BFIGJHK}.
2. T_i : A and B, there is a selection between A and B: $A \#_w B$
3. T_o : E and K, there is a OR-joint at the end that comes from E and K: $E \#_w K$
4. Based on Tw, we get the following relations: *Causality relation*: $A \rightarrow_w C$, $A \rightarrow_w D$, $A \rightarrow_w E$, $C \rightarrow_w E$, $D \rightarrow_w E$, $B \rightarrow_w F$, $B \rightarrow_w G$, $B \rightarrow_w H$, BwI, $B \rightarrow_w J$, $B \rightarrow_w K$, $F \rightarrow_w G$, $F \rightarrow_w H$, $F \rightarrow_w I$, $F \rightarrow_w J$, $F \rightarrow_w$

K, $G \rightarrow_w H$, $G \rightarrow_w K$, $H \rightarrow_w K$, $I \rightarrow_w J$, $I \rightarrow_w K$, $J \rightarrow_w K$; *Parallel relation*: $C||_w D$, $G||_w I$, $G||_w J$, $H||_w I$, $H||_w J$; *Conflict relation*: $A\#_w B$, $A\#_w F$, $A\#_w G$, $A\#_w H$, $A\#_w I$, $A\#_w J$, $A\#_w K$, $C\#_w B$, $C\#_w F$, $C\#_w G$, $C\#_w H$, $C\#_w I$, $C\#_w J$, $C\#_w K$, $D\#_w B$, $D\#_w F$, $D\#_w G$, $D\#_w H$, $D\#_w I$, $D\#_w J$, $D\#_w K$, $E\#_w B$, $E\#_w F$, $E\#_w G$, $E\#_w H$, $E\#_w I$, $E\#_w J$, $E\#_w K$

5. Based on these relations we construct workflow system as in Fig.2.

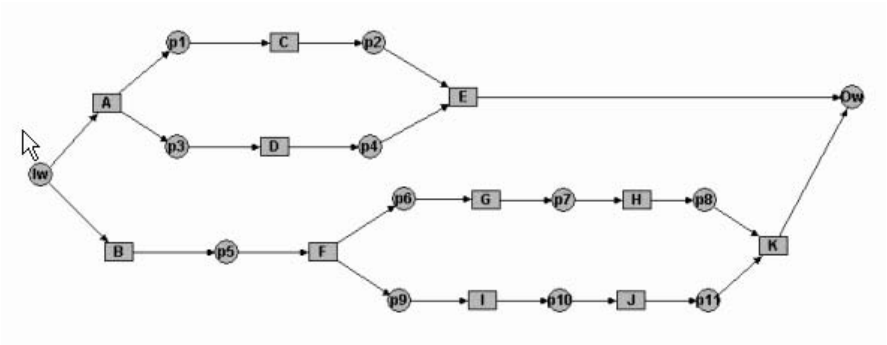


Fig. 2. Workflow constructed using alpha mining algorithm

3 Results of Experiments

In experiments we used Java implementation of the alpha workflow mining algorithm working in computer environment with Windows XP,Intel 2GHz, RAM 1GB,Java 2 SDK. From our analysis it is clear that cyclomatic complexity is not appropriate to describe complexity of the alpha workflow mining algorithm. In parallel workflow structure all concurrent transitions could be fired in arbitrary order. The number of traces depends on the number of transitions as follows: $NumberofTraces = (NumberofTransitions - 2)!$. Number of traces and size of workflow log increase with number of transitions. Size of workflow log will effect time and space consumption of mining algorithm. With increasing number of transitions the time and space consumption grow as well. We can use the Ccycle metrics to measure behavioral complexity. Ccycle defined by the difference of the total number of connections and total number of net elements. The cyclomatic complexity $C_{cycle} = F-(T+P)-Q$. Our results indicate that workflow net with the same number of $(P+T)$ could have different performance. This indicates that the cyclomatic number is not adequate to analyze complexity of workflow mining algorithm. Number of traces in parallel and conflict structures is not the same. Fig.3 illustrates number of traces in combined sequential and conflict structure.

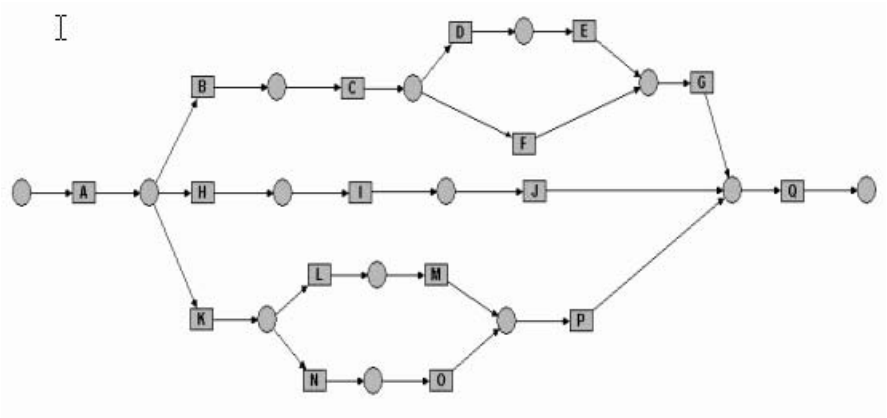


Fig. 3. Sequential and conflict control structure

4 Conclusions

The alpha mining algorithm can be used to design workflow nets and to refine the existing workflow net. The workflow has to be sound and noiseless. Different workflow structures have different effect on time and space complexity of the algorithm. When analyzing complexity in sequential and conflict workflow structures one can adopt cyclomatic complexity measure. To analyze sequential or conflict structures one can use the following formula: $C_{cycle} = F - (T + P) - Q + O$, where F (arcs) stands for the total number of connections, T the number of transitions, P the number of places. Parameter Q is the number of OR-splits in workflow net. Proper metric to present this effort is the number of traces in workflow log, therefore we can derive the following formula to valuate the complexity in parallel structure. $C_{parallel} = (Numberoftransitions - 2)!$

References

1. Cook J.E., Wolf A.L.(1998) Discovering Models of Software Processes from Event-Based Data. ACM Trans. on Software Engineering and Methodology, 7(3), 215-249, Springer, Berlin Heidelberg.
2. van der Aalst W.M.P., Weijters A.J.M.N., Maruster L.(2002) Workflow Mining: Which Processes can be Rediscovered? WP 74, Eindhoven University of Technology.
3. van der Aalst W.M.P., Weijters A.J.M.M., Maruster L, (2004) Workflow Mining: Discovering Process Models from Event Logs. IEEE Trans. on Knowledge and Data Engineering.

Recommendation Rules — a Data Mining Tool to Enhance Business-to-Customer Communication in Web Applications

Mikołaj Morzy

Institute of Computing Science
Poznań University of Technology, Piotrowo 3A, 60-965 Poznań, Poland
Mikolaj.Morzy@cs.put.poznan.pl

Abstract. Contemporary information systems are facing challenging tasks involving advanced data analysis, pattern discovery, and knowledge utilization. Data mining can be successfully employed to sieve through huge amounts of raw data in search for interesting patterns. Knowledge discovered during data mining activity can be used to provide value-added services to users, customers, and organizations.

The adoption of the Web as one of the main media for business-to-customer (B2C) communication provides novel opportunities for using data mining to personalize and enhance customer interfaces. In this paper we introduce the notion of recommendation rules — a simple knowledge model that can be successfully used in the Web environment to improve the quality of B2C relationship by highly personalized communication. We present the formalism and we show how to efficiently generate recommendation rules from a large body of customer data.

1 Introduction

Rapid scientific, technological, and social development witnessed in recent years has resulted in significant increase of the volume of data processed by computer systems. Data mining, also referred to as knowledge discovery in databases, aims at the discovery of hidden and interesting patterns from large data volumes. Discovered patterns can be used, e.g., to provide additional insight into the data, to allow prediction of future events, or to assist marketing operations. The main drawback of data mining systems is the high computational cost of knowledge discovery algorithms which disqualifies many data mining methods from straight utilization in on-line Web applications.

The Web is quickly becoming an important channel for sales and customer relationship management. Several businesses, including banks, insurance companies, retail and multimedia stores, are interacting with their customers on-line. This transition to the on-line communication channel introduces new, unprecedented requirements. Among others, user expectations of response times shrink to seconds, user identification becomes difficult, security and privacy become key issues. Increasing customer satisfaction requires precise mechanisms of B2C communication. A simple approach of broadcasting all messages to all customers, contemptuously referred to as the "spray

and pray" method, is unacceptable. Messages must be highly relevant to customers with respect to customer behavior and characteristics. Assessing the relevance is difficult, because customer personal data is often limited. On the other hand, highly personalized communication is very important in marketing applications.

Let us consider an example. A bank wants to inform its customers about a new credit card. The marketing department decides that the main addressee of the message is a young person who lives in a mid-sized city and who has a medium income. Direct translation of these constraints into a database query can be error-prone. For example, subjective choice of attribute value thresholds can lead to false positives (addressing customers who should not be bothered with the message), or false negatives (failing to address customers who should be notified).

An attempt to solve this problem using traditional data mining techniques leads to the utilization of clustering and classification. Unfortunately, both techniques are not appropriate here. Clustering is not helpful at all, because addressees of messages do not form distinct clusters. The set of addressees is determined dynamically upon the formulation of a new message and the constraints for message delivery are vague. In other words, clusters representing addressees of messages would have to be strongly overlapping, irregular, having different shapes and sizes. Classification is not helpful either, because it is impossible to acquire high-quality training and testing sets. We propose to tackle this problem from the association rule discovery perspective. In our solution conditions that determine the relevance of a message to a given customer are not formulated arbitrarily. Rather, conditions represent frequent combinations of attribute values and can be chosen by the user from a precomputed set of possible combinations. Additionally, each customer is approximated by frequent combinations of attribute values present in customer data. In other words, every customer is mapped to a point in a multidimensional space of frequent attribute values. Messages are also mapped to the multidimensional space as subregions. The inclusion of a customer point in a given message subregion implies that the customer is a potential recipient of the message.

In this paper we present recommendation rules — a simple knowledge representation model that allows high personalization of B2C communication. We introduce the notion of a recommendation rule and we present an adaptation of the well-known Apriori algorithm to pre-compute frequent sets of attribute values. The results presented in this paper originate from a prototype implementation of the system developed within the Institute of Computing Science of Poznań University of Technology. The paper is organized as follows. Section 2 presents related work. In Section 3 we present basic definitions and we formally introduce the notion of a recommendation rule. Section 4 contains the description of the mining algorithms. We conclude in Section 5 with a brief summary.

2 Related Work

The problem of mining association rules was first introduced in [1]. In [2] Agrawal et al. proposed the Apriori algorithm that quickly became the seed for several other frequent itemset mining algorithms. The original formulation of the association rule mining problem was generalized into the problem of mining quantitative association rules in [6].

Tightly coupled with quantitative association rules are clustered association rules first presented by Lent et al. in [5]. The idea behind clustering was to combine quantitative association rules for which rule antecedents or consequents corresponded to adjacent ranges of attribute values. Another similar problem was the problem of finding profile association rules, first presented by Aggarwal et al. in [3]. An exhaustive study of the subject can be found in [4]. Profile association rules correlate patterns discovered in user demographics data with buying patterns exhibited by users.

3 Definitions

Given a set of attributes $A = \{A_1, A_2, \ldots, A_n\}$. Let $dom(A_i)$ denote the domain of the attribute A_i. Let the database D consist of a relation R with the schema $R = (A_1, A_2, \ldots, A_n)$. For each tuple $r \in R$ let $r(A_j) = a_j$ denote the value of the attribute A_j in tuple r. The support of the value a_j of the attribute A_j is the ratio of the tuples $r \in R$ having $r(A_j) = a_j$ to the number of tuples in R. Given a user-defined minimum support threshold denoted as *minsup*. The value a_j of the attribute A_j is called *frequent*, if $support_R(A_j, a_j) \geq minsup$. The support of the set of values $\{a_j, \ldots, a_m\}$ of the attributes $A_j, \ldots, A_m$ is the ratio of the tuples $r \in R$ having the values of the attributes $A_j, \ldots, A_m$ equal to $a_j, \ldots, a_m$, respectively, to the number of tuples in R. The set of values $\{a_j, \ldots, a_m\}$ of the attributes $A_j, \ldots, A_m$ is *frequent*, if $support_R(A_j, a_j, \ldots, A_m, a_m) \geq minsup$. We say that a set of attribute values $\{a_j, \ldots, a_m\}$ *satisfies* the tuple r if $\forall k \in \langle j, \ldots, m \rangle : r(A_k) = a_k$. Let L denote the collection of all frequent sets of attribute values appearing in the relation R. Given a set of messages $M = \{m_1, m_2, \ldots, m_p\}$, where each message m_i is a string of characters. A *recommendation rule* is an implication of the form: $a_j \wedge a_k \wedge \ldots \wedge a_m \rightarrow k_i$, where $a_j \in dom(A_j) \wedge a_k \in dom(A_k) \wedge \ldots \wedge a_m \in dom(A_m) \wedge \exists l_q \in L : \{a_j, a_k, \ldots, a_m\} \subseteq l_q$. The left-hand side of the rule is called the antecedent and the right-hand side of the rule is called the consequent.

Each tuple $r \in R$ can be approximated using frequent sets of attribute values appearing in the tuple. The processing of recommendation rules consists in finding, for a given tuple r, all recommendation rules which apply to r, i.e., in finding all recommendation rules having the antecedent satisfying the tuple r. It is worthwhile noticing that this formulation of data mining task is fundamentally different from previous approaches. Previous approaches concentrated on efficient discovery of associations between attribute values. Our

approach takes the opposite direction, i.e., the knowledge is known *a priori* (we assume that the user has a vague notion of constraints that should be satisfied in order to send a message to a given customer), but the formulation of the knowledge is difficult. Therefore, we propose to reverse the process. Instead of formulating constraints and looking for customers satisfying those constraints, we begin with the in-depth analysis of the customer data and we derive frequent sets of attribute values to serve as descriptors for large user communities. Next, we force the user to formulate the constraints for message delivery only in terms of frequent sets of attribute values discovered during the first step. In this way, the user can not introduce arbitrary conditions that cut across communities of similar customers and the constraints for message delivery become "natural" in the sense that they represent natural clustering of attribute values present in customer data.

4 Mining Algorithms

Require: D, *minsup*, P
  Normalize(D, P);
  Discretize(D, P);
  RemoveCorrelatedAttributes(D, P);
  L_1 = set of frequent attribute values;
  $L = Apriori(D, L_1, minsup)$;
  for all tuples $t \in D$ **do**
    $L_t = subset\,(L, t)$;
    for all sets $l \in L_t$ **do**
      $\langle t.t\_id, l.s\_id \rangle \rightarrow metadata$
    end for
  end for

Require: L, M
  $rhs = \{m_i : m_i \in M\}$;
  $lhs = \emptyset$;
  $L_{LHS} = L$;
  while (*notFinished*) **do**
    $lhs = \{l : l \in L_{LHS}\}$;
    $L_{LHS} = L_{LHS} \setminus \{l : l \in L_{LHS} \wedge l \not\supseteq lhs\}$;
    $notFinished \leftarrow$ *user input*
  end while

Fig. 1. Algorithms for generation of frequent sets and recommendation rules

Figure 1 presents the algorithms for generating frequent sets and recommendation rules. This algorithm is a minor modification of the Apriori algorithm [2]. Let D denote the database of customer data. Let $M = \{m_1, \ldots, m_k\}$ denote the set of user-defined messages. Finally, let P denote the set of user-defined preferences (e.g., the correlation factor for pruning correlated attributes, parameters for attribute normalization and discretization, etc.). The first algorithm begins by performing necessary data preprocessing, such as normalization of numerical attributes and discretization of numerical attributes into discrete bins. Next, the algorithm generates all frequent sets of attribute values using the Apriori technique. In the last step, all tuples describing customers are verified for the containment of frequent sets of attribute values. For every customer tuple the information about all frequent sets of attribute values contained in the given tuple is added to the metadata.

This step is necessary for achieving a satisfying performance during runtime. Let *lhs* and *rhs* denote the left-hand side and the right-hand side of the generated recommendation rule, respectively. The user first selects the messages to be communicated to customers and adds them to the right-hand side of the rule. Next, the user adds conditions to the left-hand side of the rule. Conditions are represented by frequent sets of attribute values. In each iteration the user is free to choose from the collection of available frequent sets of attribute values, where the collection consists of all supersets of frequent sets already chosen for the left-hand side of the rule. The rationale behind this is that it guarantees that every message will be communicated to at least *minsup* fraction of customers, since every left-hand side must necessarily belong to the collection of frequent sets of attribute values. Specialization of a given recommendation rules continues until the user is satisfied with the joint condition.

5 Summary

In this paper we have presented the idea of recommendation rules. It is a knowledge representation model that allows organizations to personalize messages addressed to their customers, avoiding the arbitrary choice of message delivery criteria. In addition to the formulation of the problem, we have presented an algorithm for efficient definition of recommendation rules. Our prototype implementation proves that this idea is applicable in real-world applications. The integration of data mining techniques with Web applications is a very promising research area. Recommendation rules and the prototype presented in this paper provide an interesting step into this domain.

References

1. Agrawal R., Imielinski T., Swami A. (1993) Mining Association Rules between Sets of Items in Large Databases. ACM Press, Proc. of the ACM SIGMOD International Conference on Management of Data, Washington, D.C., 207–216
2. Agrawal R., Srikant R. (1994) Fast Algorithms for Mining Association Rules in Large Databases. Morgan Kaufmann, Proc. of 20th International Conference on Very Large Data Bases, Santiago de Chile, Chile, 487–499
3. Aggarwal C. C., Sun Z., Yu P. S. (1998) Online algorithms for finding profile association rules. ACM Press, Proc. of the 7th International Conference on Information and Knowledge Management, Bethesda, Maryland, 86–95
4. Aggarwal C. C., Sun Z., Yu P. S. (2002) Fast Algorithms for Online Generation of Profile Association Rules. IEEE Transactions on Knowledge and Data Engineering **14(5)**, 1017–1028
5. Lent B., Swami A. N., Widom J. (1997) Clustering Association Rules. IEEE Computer Society, Proc. of the 13th International Conference on Data Engineering, Birmingham U.K., 220–231
6. Srikant R., Agrawal R. (1996) Mining Quantitative Association Rules in Large Relational Tables. ACM Press, Proc. of the ACM SIGMOD International Conference on Management of Data, Montreal, Canada, 1–12

Feasibility Studies of Quality of Knowledge Mined from Multiple Secondary Sources I. Implementation of generic operations

Wiesław Paja and Zdzisław S. Hippe

Department of Expert Systems and Artificial Intelligence, University of Information Technology and Management in Rzeszow, Sucharskiego 2, Rzeszow, 35-225 Poland

Abstract. In the paper continuation of our research described earlier is shortly discussed. The main goal of current investigation is to build combined learning model (probably quasi-optimal) merging some knowledge models, developed by means of different machine learning algorithms. In order to reach this goal, a set of generic operations were implemented and tested on melanocytic dataset.

1 Introduction

Until now, in our research [1] the new conception of knowledge extraction from data has been developed and extensively studied [2,3]. The characteristic feature of our approach was the multiple (i.e. by means of different machine learning tools) analysis of a *primary source* of knowledge (e.g. decision table [4]), which supplied multiple learning models, called by us *secondary sources* of knowledge. In the next step of our approach, all developed learning models, always in the form of IF ... THEN rules, have been merged together into one combined learning model, called by us **Q**uasi-**O**ptimal **L**earning **M**odel (**QOLM**), and finally optimized using a set of generic optimization operations. Some of these operations have been already described elsewhere [1,5–7], more details are given in Section

2 Methodology used

Two different methods were used in the research. The first one relies on a separate optimization of each developed learning model. The second method was distinctly different. In the first step, as it was stated in the previous section, all developed learning models (production rules) were merged together and then, the entire joined (large) model, was optimized using the same set of generic operations. To generate learning models (in the form of decision rules) the following machine learning algorithms were used: (1) a pathway from general description of a problem to its specific description (**g**eneral-**t**o-**s**pecific, GTS [8]), (2) development of rules using belief networks [9], and (3)

standard R. Quinlan's C4.5 procedure, implemented in [10]. Improved models for both methods, were then evaluated — via testing the classification accuracy of unseen examples (see Section 4).

3 Implementation of selected generic operations

The full set of generic optimization operations is as follows:

1. *finding and removing redundancy:* the data may be overdetermined, that is, some rules may explain the same cases. Here, redundant (excessive) rules were analyzed, and the redundant rule (or some of the redundant rules) was (were) removed, provided this operation did not increase the error rate;
2. *finding and removing of incorporative rules:* another example when the data may be overdetermined. Here, some rule(s) being incorporated by another rule(s) were analyzed, and the incorporative rule(s) was (were) removed, provided this operation did not increase the error rate;
3. *merging rules:* in some circumstances, especially when continuous attributes were used for the description of objects being investigated, generated learning models contained rules that are more specific than they should be. In these cases, more general rule(s) were applied, so that they cover the same investigated cases, without making any incorrect classifications;
4. *finding and removing of unnecessary rules:* sometimes rules developed by the systems used were unnecessary, that is, there were no objects classificated by this rules. Unnecessary rule(s) was (were) removed, provided this operation did not increase the error rate;
5. *finding and removing of unnecessary conditions:* sometimes rules developed by the systems used contain unnecessary conditions, that were removed, provided this operation did not increase the error rate;
6. *creating of missing rules:* sometimes developed models didn't classify all cases from learning set. Missing rules were generated using a set of unclassified cases;
7. *discovering of hidden rules:* this operation generates a new rule by combination of similar rules, containing the same set of attributes and the same — except one — attribute values;
8. *rule specification:* some rules caused correct and incorrect classifications of selected cases, this operation divides considered rule into few rules by adding additional conditions;
9. *selecting of final set of rules:* there were some rules that classify the same set of cases but have different composition, simpler rule stayed in a set;

In our experiments, only the first five generic operations were applied.

Classification process of unseen objects consists in importing of testing set and categorization of investigated objects using the set of developed rules.

Results of this procedure are presented in Table 1, taking into account the number of classified cases, number of incorrect classifications, number of not classified cases, and the error rate. Additionally, the confusion matrix is generated.

4 Results and discussion

The initial results of improvement of learning models are gathered in Table 1. It should be stressed, that accuracy and quality of learning model developed for the same primary source are very different. Learning model developed with the first algorithm (Model 1) contains **116** decision rules, whereas the average error rate equals to **24,1%**. The second learning algorithm (Model 2) contains only **27** decision rules, however, the error rate is much larger (**42,9%**). Learning model developed using the third algorithm (Model 3) contains only **38** decision rules, testing cases with error rate equal **19,9%**. Quasi-Optimal Learning Model, obtained by simple merging together all multiple models, contains **178** decision rules, which classify testing cases with quite satisfactory error rate equal **15,7%**. After optimization (using the generic operations), the number of rules decreased to: **102** (Model 1), **16** (Model 2) and **38** (Model 3). In the case of QOLM the number of rules decreased from **178** to **137**, what have to be emphasized, without increasing of error rate. Also, the average rule strength [11], defined as: The bibliography shall be ordered alphabetically. Follow the examples at the end of this paper. The bibliography items should receive symbolic names like:

$$\frac{\Sigma Strength(R)}{n} \tag{1}$$

where: *Strength(R)* — is the total number of cases correctly classified by the rule during training, n — is number of rules in the model being investigated, was also calculated.

To summarize results obtained, it might be stated that the optimization of learning models, using only first five generic operations, yielded quite interesting and satisfactory results. Namely, the overall number of rules decreased about **27,3%**, average number of conditions decreased about **43,4%**, average value of rule strength increased about **38,2%**, and error rate increased about **1,6%**. The results obtained can be additionally interpreted in somewhat different way: the improvement of learning models will play a significant role in a case of very extended models, i.e. models which contain very large set of rules. However, the incident decrease of the overall accuracy of the improved model can be observed. In the future research, the untouched generic operations (#6 — #10) will be dealt with.

Table 1. Learning models and theirs parameters, before and after optimization

	Model 1	Model 2	Model 3	QOLM
Before optimization				
Number of rules	116	27	38	179
Number of conditions in a set of rules	622	98	129	715
Number of classificated cases *	49	35	55	54
Number of incorrect classifications	7	4	11	9
Number of unclassificated cases	6	20	0	0
Error rate [%]	24,1	42,9	19,9	15,7
Average value of rule strength	8,3	12,6	12,6	11,2
After optimization				
Number of rules	103	17	38	137
Number of conditions in a set of rules	489	56	98	512
Number of classificated cases *	49	36	55	54
Number of incorrect classifications	7	4	12	9
Number of unclassificated cases	6	19	0	0
Error rate [%]	23,7	42,9	21,5	15,7
Average value of rule strength	11,5	25,2	18,5	19,6

* (correctly and erroneously)

References

1. Hippe Z.S., Knap M., Paja W.: Feasibility Studies of Quality of Knowledge Mined from Multiple Secondary Sources, In: Klopotek M.A., Wierzchon S., Michalewicz M. (Eds.); Intelligent Information Systems '2002; Springer-Verlag, pp. 361-364
2. Hippe Z.S., Bajcar S., Blajdo P., Grzymala-Busse J.P., Grzymala-Busse J.W., Knap M., Paja W., Wrzesien M.: Diagnosing Skin Melanoma: Current Versus Future Directions, TASK Quarterly, 2003 (**7 No 2**) pp. 289-293
3. Hippe Z.S., Paja W.: Qasi-Optimal Learning Models. I. Research on Generic Operations in Process of Merging of Secondary Sources, In: Tadeusiewicz R., Ligeza A., Szymkat M. (Eds.) 4th National Conference: Computer Systems and Methods in Science and Engineering, Edit Office of AGH University of Science and Technology, Cracow 2003, pp. 485-490 (in Polish)
4. Pawlak Z.: Knowledge and Rough Sets In: Traczyk W. (Ed.) Problems in Artifical Intelligence, Wiedza i ycie, Warsaw 1995, pp. 9-21 (in Polish)
5. Jagielski J.: Knowledge Engineering in Expert Systems, (in Polish) Lublin Science Society, Zielona Gra 2001, pp. 79-83
6. Hippe Z. S., Knap M., Paja W.: Research on Quality of Knowledge Based on Analysis of Different Source of Knowledge, in: Nycz M., Owoc M. L. (Eds.), Knowledge Acquisition from Data Bases; Edit Office of Wroclaw University of Economics, Wroclaw 2002, pp. 233-244 (in Polish)
7. Wang J.: Data Mining: Opportunities and Challenges, Idea Group Publishing, Hershey 2003, pp. 57-59
8. Hippe Z.S.: Machine Learning — a Promising Strategy for Business Information Systems?, In: Abramowicz W. (Ed.), Business Information Systems'97, Academy of Economics, Poznan 1997, pp. 603-622

9. Hippe Z.S., Grzymala-Busse J.W., Mroczek T.: Rules from Belief Networks: A Rough Set Approach, In: Tsumoto S., Sowinski R., Komorowski J., Grzymala-Busse J.W. (Eds.) 4th International Conference, Rough Sets and Current Trends in Computing, Springer-Verlag Berlin Heidelberg 2004, pp. 483-487
10. http://www.aitech.com.pl
11. Bajcar S., Grzymala-Busse J.W., Hippe Z.S.: A comparison of Six Discretization Algorithms Used for Prediction of Melanoma, In: Klopotek M.A., Wierzchon S., Michalewicz M. (Eds.); Intelligent Information Systems '2002; Springer-Verlag, pp. 3-12

Modern Metaheuristics for Function Optimization Problem

Marek Pilski[1], Pascal Bouvry[2], and Franciszek Seredyński[1,3,4]

[1] Institute of Computer Science, Academy of Podlasie, Sienkiewicza 51, 08-110 Siedlce, Poland
[2] University of Luxembourg, Faculty of Sciences, Technology and Communication, 6, rue Coudenhove Kalergi, L1359 Luxembourg-Kirchberg, Luxembourg
[3] Polish-Japanese Institute of Information Technology, Koszykowa 86, 02-008 Warsaw, Poland
[4] Institute of Computer Science, Polish Academy of Sciences, Ordona 21, 01-237 Warsaw, Poland

Abstract. This paper compares the behaviour of three metaheuristics for the function optimization problem on a set of classical functions handling a lot number of variables and known to be hard. The first algorithm to be described is Particle Swarm Optimization (PSO). The second one is based on the paradigm of Artificial Immune System (AIS). Both algorithms are then compared with a Genetic Algorithm (GA). New insights on how these algorithms behave on a set of difficult objective functions with a lot number of variables are provided.

1 Introduction

There is a wide range of problems, classified as NP-hard, that appear impossible in practice to solve. In such case, various heuristics may be useful, which, however, do not always guarantee that the optimal solution is found. Nevertheless, a ,,good" solution within a reasonably short period of time can be found. The multi-variable optimization field for non-linear functions includes in particular many intractable problems.

In the paper we examine two new algorithmic techniques belonging to the ,,nature–inspired techniques" class: i.e. Particle Swarm Optimization (PSO) [4] and Artificial Immune System (AIS) [1]. Next we compared them to the classical Genetic Algorithm (GA) [3] on 6 different problems described by multi-variables functions.

In to compare the behaviour of three different algorithms on a given set of optimization functions with a lot number of variables and that are known to be extremely difficult to solve and to provide new insights on using these algorithms for the proposed objective functions.

We will compare all three heuristics in a common testing environment of well-known functions, which constitute the experimental firing ground for many optimization methods.

The paper is organized in the following way. The next section presents PSO algorithm. Section 3 describes AIS. Section 4 presents a set of test functions and section 5 shows resutlts of experiment study. Last section contains conclusion.

2 Particle Swarm Optimization

Initial position of individuals is chosen at random from the solutions space. Next, a single particle may move in the direction described by the equation:

$$\begin{cases} v^{t+1}[k] = c_1 r_1 v^t[k] + c_2 r_2 \big(y^t[k] - x^t[k]\big) + c_3 r_3 \big(y^{*t}[k] - x^t[k]\big) \\ x^{t+1}[k] = x^t[k] + v^{t+1}[k], \end{cases}$$

where v^t – speed of a molecule at a time t; x^t – position of a molecule at a time t; y^t – the best position found so far by a molecule for a time t; y^{*t} – the best position fund so far by the neighbours for a time t; c_1, c_2, c_3 – weight coefficients defining the influence of each of the three elements of the trade-off, which respectively define how much a molecule trusts itself at a time, its own experience and its neighbours and their experience; $[k]$ – k-th vectors coordinates x, v and y of the length equal to the number of dimensions of the space of solutions.

Coefficients c_1, c_2, c_3 are multiplied by random values r_1, r_2 et r_3, which belong to $\langle 0, 1\rangle$ range and are defined for every generation.

The value of the vector of speed can vary, which prevents the travelling of an individual through a space of solutions, along the straight line. The change in the vector's value is calculated in the following way:

$$v_i(t) = \chi\Big(v_i(t-1) + \rho_1\big(p_i - x_i(t-1)\big) + \rho_2\big(p_g - x_i(t-1)\big)\Big),$$

where v_i – vector of speed of an individual in an iteration i; p_i – best position of a given individual ($pbest_i$ – value for the position p_i); p_g - position of the best individual in the whole population ($gbest$ – its value). Moreover, parameters ρ_1 and ρ_2 may influence the vector of speed of a molecule. First of them influences p_i value, the second on p_g. A change in these parameters changes the force of influence of the best values found so far on molecules. Influence on speed has parameter called inertia weight χ.

$$\chi = \frac{\kappa}{abs\left(\frac{1-\frac{\rho}{2}-\sqrt{abs(\rho^2-4\rho)}}{2}\right)},$$

where κ – coefficient, $\kappa \in (0, 1\rangle$; ρ – coefficient, which is the sum of ρ_1 and ρ_2. A similar solution was proposed in paper [2] [4] [5].

3 Artificial Immune System

The idea of the algorithm presented in this paper was originally proposed by L.N. de Castro i F. J. Von Zuben'a [1] who called it CLONALG. To use the algorithm it was necessary to assume, that it does not refer to a predetermined, public set of antigens because the set of antigens is constituted

by unknown maxima of function $f(x)$. As affinity of antibody p and antigen we shall take the value of function $f(p)$. In the algorithm, an antibody is a vector of floating point numbers. The following changes have been made to the algorithm for experimental purposes. Not every variable in the vector of solutions (antibody) undergoes mutation – the probability of mutation of each variable equals 50% Single-ended crossover operator has been added – crossover follows mutation, and it is possible to influence its probability.

Moreover, in each iteration of the algorithm, the following operations are carried out sequentially:
Cloning, two ways. Linear – the best individual is cloned until the number of clones reaches the number of individuals selected for cloning, in other individuals the number of clones made is decreased by one. Constant – the best individual is cloned $N/2$ and the remaining individuals $N/4$, where N – the number of individuals selected for cloning.
Hipermutation. For vector $x = [x_1, x_2, ..., x_i]$ output vector $z = [z_1, z_2, ..., z_i]$ is calculated from: $z_i = x_i \pm \Delta x = X \cdot \alpha \cdot Rand\,\langle 0, 1\rangle$, where X – absolute value from the range of the function in question; α – parameter defining the degree of mutation calculated from: $\alpha = exp(-\rho \cdot D)$, where ρ – mutation coefficient; D – fitness coefficient equal $1 - n/N$; n – position of an antibody in a vector of solutions of the length N sorted according to affinity.
Crossover. Two antibodies and a point of section are chosen at random. The first part is taken from one parent and the second form the other one. Then, the parts are joined together after we have chosen the order of the parts in a new body.

4 Test functions

Six test functions were used to carry out the experiments. The functions to be minimized are presented below:

- sphere model, $x \in R^n$, a minimum $x^* = (0, ..., 0)$, $f_1(x^*) = 0$, $f_1(x) = \sum_{i=1}^{n} x_i^2$,
- Ackley's function, $x \in R^n$, a minimum $x^* = (0, 0, ..., 0)$, $f_2(x^*) = 0$, $f_2(x) = -20e^{\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right)} - e^{\left(\frac{1}{n}\sum_{i=1}^{n} cos 2\pi x_i\right)}$,
- Griewank's function, $x \in R^n$, a minimum $x^* = (0, 0, ..., 0)$, $f_3(x^*) = 0$, $f_3(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$,
- Rastrigin's function, $x \in R^n$, a minimum $x^* = (0, 0, ..., 0)$, $f_4(x^*) = 0$, $f_4(x) = \sum_{i=1}^{n} \left(x_i^2 - 10 \cdot \cos(2\pi x_i) + 10\right)$,
- Rosenbrock's function, $x \in R^n$, a minimum $x^* = (1, 1, ..., 1)$, $f_5(x^*) = 0$, $f_5(\mathbf{x}) = \sum_{i=1}^{n} \left(100\left(x_i^2 - x_{i+1}\right)^2 + (1 - x_i)^2\right)$,
- Schwefel's function, $x \in R^n$, a minimum $x^* = (0, 0, ..., 0)$, $f_6(x^*) = 0$, $f_6(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j\right)^2$.

5 Experimental study

The algorithms discussed (PSO, AIS and GA) were tested using functions presented in section 4. In the experiments, which were carried out, the precision of results was 10^{-6}. Experiments were carried out for the following number of variables $n = 5, 10, 20, 30$ but the results were presented only for $n = 30$. All the algorithms were run for 200 generations.The experiments, which were not presented in this paper aimed at setting parameters of the algorithms. For the sake of the presentation, each experiment was repeated 25 times, with parameter values predefined, and the best result was presented for each generation. Figure 1 show mean results of experiments.

This comparison is not purely linear because we do not take an important parameter – time of execution of one generation of the algorithm – under consideration. It is worth noticing that AIS took 50 times longer to execute than that of PSO, which results form computational complexity of the compared algorithms. AIS algorithm performs multiple loops over the whole population of antibodies (Ab and Abn) and it at the same time carries out costly/expensive operations of reproduction and sorting.

Moreover, it is worth noticing, that finding an optimum of a function in PSO algorithm after not more than 20 generations decreases rapidly and stays on a certain level of values of the functions tested and further populations do not bring a noticeable improvement of the result. AIS and GA behave in a different way. As the number of generations increases, they gradually and linearly approach the optimum of the functions tested, with AS being faster as far as reaching minimum is concerned.

6 Discussion and Conclusion

New insights on how 3 evolutionary algorithms (i.e. PSO, AIS and GA) behave and compare with each other on a set of difficult objective functions with a large number of variables have been provided.

Experiments carried out in the same testing environment proved that AIS gave better results for them than PSO and GA in terms of finding the minimum of multiple variables function. Classical GA was to the least efficient in the majority of tests. It is also worth noting that PSO is the fastest heuristic among those presented in the paper. It results from the fact that no genetic operators are used in this method and the whole population is used in the next generation. In order to define linear effectiveness of optimization methods compared in this paper, experiments assessing time of execution of the algorithms should be carried out. Furthermore, it is possible to find the best available modifications improving the efficiency of the algorithms discussed and carry out experiments again in order to establish the best method of optimization for a given class of functions.

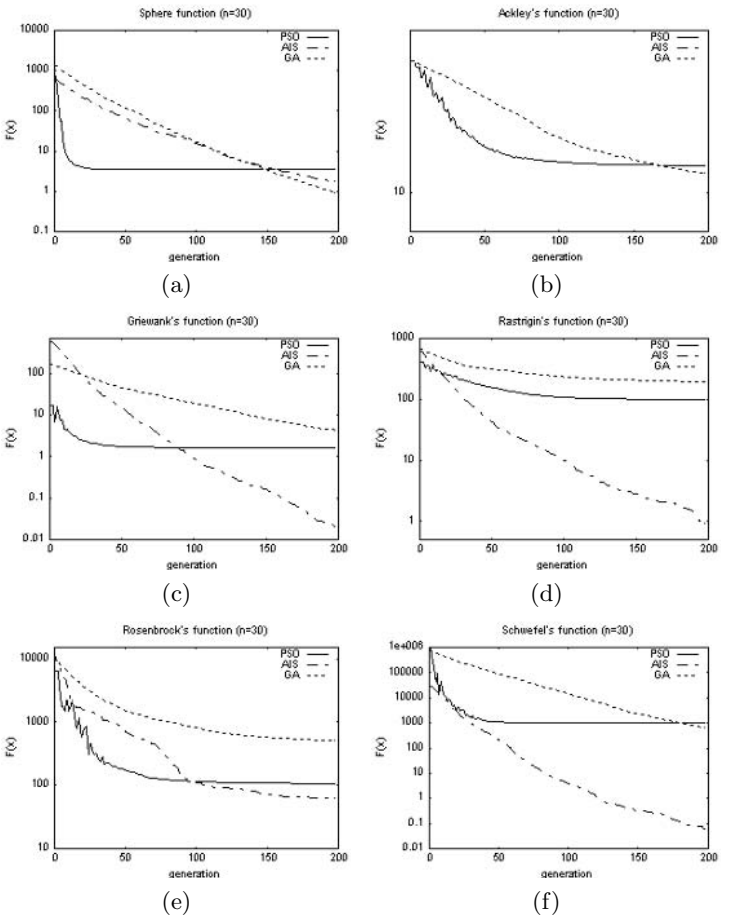


Fig. 1. Minimization of function: (**a**) Sphere (**b**) Ackley's (**c**) Griewank's (**d**) Rastrigin's (**e**) Rosenbrock's (**f**) Schwefel's.

References

1. Leonardo N. de Castro, Fernando J. Von Zuben (June 2002) ,,Learning and Optimization Using the Clonal Selection Principle", IEEE Transaction on Evolutionary Computation, vol. 6. no. 3
2. Liping Zhang, Huanjun Yu, and Shangxu Hu (2003) ,,A New Approach to Improve Particle Swarm Optimization", GECCO 2003
3. Michalewicz Z. (1996) ,,Genetic Algorithms + Data Structures = Evolution Programs", Springer
4. Shi Y. and Eberhart, R. (2000) ,,Experimental study of particle swarm optimization", Proc. SCI2000 Conference, Orlando, FL
5. Shi Y. and Eberhart R. (2001) ,,Fuzzy adaptive particle swarm optimization", Proceedings of the 2001 Congress on Evolutionary Computation

Property Driven Mining in Workflow Logs

Ella E. Roubtsova

TU Eindhoven, Den Dolech 2, P.O. Box 513, 5600MB Eindhoven, The Netherlands.

Abstract. We present a language for property specification for workflows and a tool for property checks. The language is based on the Propositional Linear Temporal Logic and the structure of workflow logs. These language and tool help companies to diagnose business processes, react to changes in business environment and collect formal definitions of business properties. We give examples of specifications of business properties that set relations between events of business processes

1 Introduction

Recording logs of events is normal practice of any well-organized business. These logs represent the real workflows of an enterprise and its workflow management is interested in log diagnostics. However, workflow logs are huge files and their effective diagnostics is not possible without formal specification of properties and tool support for property checks.

Most of current approaches make diagnostics of structural properties of processes recorded in logs, such as soundness of the workflow that is constructed from the log. Other approaches investigate the equivalence between a logged process and a standard workflow (We refer the reader to a good survey [1]). Monitoring and filtering of event logs using the database techniques is usual practice in the world of electronic payment systems and web-services. However, the properties that set relations between events of business processes are usually presented only informally. The analysis or the audit of logs have no support for automatic checks of such properties. Modern management needs precise techniques to make diagnostics of business properties faster.

In this paper we present a property diagnostics for workflows based on a declarative language for workflow property specification. The language has been developed to help analysts in formulating log properties in such a way that the properties can be checked automatically. The language is based on the structure of logs and the Propositional Linear Temporal Logic (PLTL) [3]. The standard structure of logs is used when building algorithms for property checks. Our tool for property driven workflow mining combines a tool-wizard for property construction, property parsers for syntax checkers and a verifier for property verification. Section 2 of the paper formalizes a workflow log. Section 3 shows the grammar and semantics of our workflow property language and presents examples of workflow properties. Section 4 concerns

the implementation issues of constructing, parsing and checking of property expressions. Section 5 concludes the paper.

2 Formal representation of a log

The formats of logs produced by workflow management systems have common elements [4]. To define a language for workflow properties specification we formalize a log. An event is represented in a log by an *audit trail entry*

	Data	WorkflowModelElement	EventType	Timestamp	Originator
1	Data	A	schedule	1899-12-30T00:04:00.0 00+01:00	Jan
2	Data	A	start	1899-12-30T00:05:00.0 00+01:00	Sara
3	Data	A	suspend	1899-12-30T00:07:00.0 00+01:00	Sara
4	Data	A	resume	1899-12-30T00:09:00.0 00+01:00	Sara
5	Data	A	suspend	1899-12-30T00:14:00.0 00+01:00	Rit
6	Data	A	resume	1899-12-30T00:15:00.0 00+01:00	Rit
7	Data	A	complete	1899-12-30T00:17:00.0 00+01:00	Rit

Fig. 1. An example of a process instance

(ate). An ate is a record with the following fields: $WE : String($ *workflow element*$)$, $ET : String$ *(event type)*, $TS :$ *double (time stamp)* and $O : String$ *(originator)*: $ate = \{WE, ET, TS, O\}$.

A sequence of $ate's$ in a log is called a *process instance* (Figure 1). A process instance defines a total order relation on the set ATE of audit trail entries $T = \{(ate_1, ate_2) |\ ate_1, ate_2 \in ATE\ \wedge\ ate_2\ follows\ ate_1\}$:

$$p = (pName, (WE, ET, TS, O), T).$$

A set P of process instances defines a workflow $w = (wName, P)$.
A set W of workflows is grouped into a log: $L = (LogName, W)$.

3 A language to specify properties of workflows

Properties of an audit trail entry.
By default any appearance of a field identifier for a specific ate derives the value of the field from this ate. To compare the values of the string-fields WE, ET, O with some site-defined values we use operations equal $=$ and not equal $!=$. To work with the time stamp field TS we use the complete set of comparison operators and some functions to derive, for example, the hour from the time stamp. For the sake of simplicity, we do not define here the complete set of such functions. So, an *elementary property* of an audit trail

entry is represented by a property of a field in form of one of the following expressions:
$< element >::= true \mid false \mid WE\ =< string > \mid WE\ != < string > \mid$
$ET\ =< string >\ \mid ET\ != < string >\ \mid O\ =< string >\ \mid O\ != < string > \mid$
$TS\ =< double > \mid TS\ != < double > \mid TS\ < < double >\ \mid TS\ > < double >\ \mid$
$TS\ >= < double >\ \mid TS\ <= < double >\ \mid HOURS(TS) =< double >$.
We also use variables to memorize field values and the arithmetic and the comparison operations on variables. To assign values of fields to variables at the level of elementary properties we define the function $assign :=$ which always returns value $true$: $< element >::=< Variable >:= WE \mid$
$< Variable >:= ET \mid\ < Variable >:= TS \mid\ < Variable >:= O)$.

Elementary properties are combined into logical expressions by means of logical operations (the operations are represented in order of decreasing priority):
$< expr >::=< element > \mid \neg < expr > \mid < expr > \& < expr > \mid$
$< expr > \mid\mid < expr >$.

Properties of a process instance. A process instance has twofold nature: it is both a sequence of *ate's* and a relation. Business properties are usually relate events of a process instance to each other: one event causes other events or prevents them. To express relations of events in the process instance desired by business we adopt the set of the linear temporal operators expressing properties of sequences [2,3]:
$< tempexpr >:=< expr > \mid \neg < tempexpr > \mid < tempexpr > \& < tempexpr > \mid$
$< tempexpr >\ \mid\mid\ < tempexpr > \mid$ NEXT $(< tempexpr >) \mid$
IN_FUTURE $(< tempexpr >) \mid$ALWAYS $(< tempexpr >) \mid$
$(< tempexpr >)$ EXISTS_UNTIL $(< tempexpr >) \mid$
$(< tempexpr >)$ ALWAYS_UNTIL $(< tempexpr >)$.

The semantics of the properties is defined by a satisfaction relation. To define the semantics we construct a Kripke structure [2]: $M_{tempexpr} = (ATE, T, \nu)$, where ATE is a finite set of audit trail entries being states of a process instance; T is a binary ordering relation on audit trail entries which defines the initial audit trail entry and a single transition from each *ate* to the next one; $\nu : ATE \rightarrow 2^{tempexpr}$ assigns true values of a temporal property to each *ate* in the process instance:
1. (ATE,T,ate_i) $\models expr$ iff $expr \in \nu(ate_i)$.
2. (ATE,T, ate_i) $\models \neg tempexpr$ iff $expr \notin \nu(ate_i)$.
3. (ATE,T,ate_i) $\models tempexpr \wedge tempexpr_1$ iff $ate_i \models tempexpr$ and $ate_i \models tempexpr_1$.
4. (ATE,T,ate_i) $\models tempexpr || tempexpr_1$ iff $ate_i \models tempexpr$ or $ate_i \models tempexpr_1$.
5. (ATE,T,ate_i) $\models$ NEXT $(tempexpr)$ iff *for* $(ATE, T) : ate_i, ate_{i+1}, ...$
$ate_{i+1} \models\ tempexpr$.
6. (ATE,T, ate_i) $\models$ ALWAYS $(tempexpr)$ if *for* $(ATE, T) :\ ate_i, ate_{i+1}, ...$ for all $j \geq i$ $ate_j \models\ tempexpr$.
7. (ATE,T, ate_i) *models* IN_FUTURE $(tempexpr)$ iff *for* $(ATE, T) :\ ate_i, ate_{i+1}, ...$ for some $j \geq i$ $ate_j \models\ tempexpr$.
8. (ATE,T, ate_i) $\models$ $((tempexpr)$ EXISTS_UNTIL $(temexpr_1))$ iff *for* $(ATE, T) :$ $ate_i, ate_{i+1}, ...$ for some $j \geq i$ $ate_j \models\ tempexpr_1$ and for some k, $k < j$,

$ate_k \models tempexpr$.
9. (ATE,T, ate_i) $\models$ (($tempexpr$) ALWAYS_UNTIL ($temexpr_1$)) iff for (ATE, T) :
$ate_i, ate_{i+1}, ...$ for some $j \geq i$ $ate_j \models tempexpr_1$ and for all k, $k < j$,
$ate_k \models tempexpr$.

The definition of the language allows analysts nesting of properties, which means changing the position of the *ate* in the log for which the property must hold. In such a way the relation of several events in a business process can be specified.

Properties of a log. The language for specification of a log properties has to be completed by the specification of the scope of a property. A property of a log can cover one, several or all process instances of a workflow and also one, several or all workflows of a log:
$logproperty$::=$< LOGQ >< PROCESSQ >< INSTANCEQ >< tempexpr >$
$< LOGQ >$::=FOR-ALL-LOGS | EXISTS-LOG | FOR-LOG $< name >$;
$< PROCESSQ >$::= FOR-ALL-PROCESSES | EXISTS-PROCESS |
FOR-PROCESS $< name >$;
$< INSTANCEQ >$::=FOR-ALL-INSTANCES | EXISTS-INSTANCE |
FOR-INSTANCE $< name >$;
$< name >$::=$< string >$.

Property description	Property specification
The four eyes principle dictates that at least two different persons must witness certain activities. This helps to protect an organization from dishonest individuals and unintended mistakes. Our property tells that the first workflow element is *authentication* and the second workflow element is *authentication* but the originators of these events are different.	FOR-LOG "organization" FOR-PROCESS "logon" FOR-INSTANCE "network logon 18.09.2004" IN FUTURE ((WE="authentication" & or1:=O) & NEXT (WE="authentication" & or2 :=O) & (or1! = or2))
Absolute deadline Workflow element A of the process instance should be completed until 18.00. The property holds for the process instance shown in **Figure 1.**	FOR-LOG "pn-ex-15.xml" FOR-PROCESS "main-process" FOR-INSTANCE "experiment" IN FUTURE (WE="A" & ET="complete" & HOURS(TS) = 18.00).
Relative deadline Workflow element *A* should be completed in ten hours from the start. The property does not hold for the process instance shown in **Figure 1.**	FOR-LOG "pn-ex-15.xml FOR-PROCESS "main-process" FOR-INSTANCE "experiment" ((WE="A".& ET="start" & t1:=Hours(TS)) EXISTS UNTIL (WE="A" & ET="complete" t2:=HOURS(TS)) & t2-t1<= 10)).

Fig. 2. Examples of properties

Examples of properties of a process instance. Sometimes it is not trivial to specify a property. Companies can overcome this little disadvantage of the proposed methodology by employing specialists responsible for correct specification of properties. Those specifications can be saved under recognizable names in order to be used by personnel doing the everyday monitoring of logs. Figure 2 contains some examples of such properties.

4 A tool for automatic property checks

To implement the tool for automatic property checks we have solved the following tasks: property constructing, property parsing and property checking.

Property constructing has been implemented as a wizard-tool which shows the set of the temporal operators, the lists of fields and logical connectors which allow constructing a property and choosing its scope.

Property parsing has been implemented using the Java Compiler Compiler (JavaCC) [5] for the grammar presented in Section 3. There have been developed parsers for temporal expressions, for expressions, and for simple logical expressions to build property checkers on their basis.

Property checking of expressions is performed as interpretation of expressions during parsing. An expression *expr* is evaluated for one *ate* in a process instance. The temporal property checking could not be done during parsing because of the parser backtracking problem. So, each kind of temporal operator is evaluated by the corresponding function. Recursive application of these functions is used to evaluate temporal logical expressions for a process instance. The tool for property checks has been developed as an independent component-prototype to be integrated into the mining tool created by the IS group of the Technology Management Faculty at the TU Eindhoven.

5 Conclusion

In this paper we have presented a language for specification of workflow properties and a tool for automatic property checks. Our approach helps companies to be more flexible, react faster and reengineer business processes in response to new business requirements. This approach allows collecting the best practices of log analysis in the form of formal properties and leads to the standardized understanding of business requirements.

Acknowledgement The author thanks Prof. Wil van der Aalst for sharing insights and Peter van den Brand for advises on programming in Java.

References

1. Aalst van der W.M.P., van Dongen B.F., Herbst J., Maruster L., Schimm G., and Weijter A.J.M.M.(2003) Workflow Mining: a Survey of Issues and Approaches. Data and Knowledge Engineering.**47(2))**, 237–267
2. Alur R., C. Courcoubetis, D.L. Dill.(1993) Model-Checking in Dense Real-Time. Information and Computation. **104(1)**,2–34
3. Bernard B., Bidoit M., Finkel A., Laroussinie F., Petit A., Petrussi L., Schnoebelen Ph., McKezie P. (2001) Systems and Software Verification. Model-Checking Techniques and Tools. Springer-Verlag
4. Dongen B.F.van, Aalst W.M.P.van der.(2004) EMiT: A Process Mining Tool. ICATPN 2004. Ed. J.Cortadelle and W.Reisig. LNCS 3099. 454–463
5. JavaCC. (2004)A parser/scanner generator for java, http://javacc.dev.java.net

Logical Design with Molecular Components

Filomena de Santis and Gennaro Iaccarino

Department of Informatica e Applicazioni "R.M. Capocelli" University of Salerno, Via S. Allede, 84081 — Baronissi, Italy

Abstract. We propose a theoretical model to realize DNA made circuits based on *in-vitro* algorithms, to perform arithmetic and logical operations. The physical components of the resulting Arithmetic-Logic Unit are a variety of elements such as biochemical laboratories, test tubes and human operators. The advantage of the model is the possibility to perform arithmetic operations with huge binary numbers.

1 Introduction

As it is well known, arithmetic and logical operations are done in a conventional computer by the Arithmetic-Logic Unit that often incurs in overflow or underflow problems, due to the minimal and maximal size of the representable numbers. We present some basic instruments to realize simple circuits based on DNA algorithms that constitute the bio-hardware of a DNA Arithmetic-Logic Unit overcoming overflow and underflow arithmetic limitations. Table 1 summarizes the notations and the chemical operations used in the sequel.

Table 1. Notations and Chemical Operations

Symbols	
x, ¬x	Generic DNA sequence and its complement
x^i	i repetition of x sequence
↑ x, ↓ x, ↕ x	Upper, lower and double strand x
Chemical Operations	
Synthesis:	Generating of DNA single strands in vitro.
Annealing:	Bounding of two complementary DNA single strands
Cutting:	Cutting a DNA double strand by restriction enzyme
Ligation:	Pasting two DNA double strands by restriction enzyme
Polymerization:	Generating a complete double strand from a portion of it
Gel Electrophoresis:	Separating DNA strands by electric charges
PCR:	Cloning and amplifying a particular DNA piece in solution

2 Representing Binary Strings

Each binary number can be encoded by a set of integers indicating the positions where bits are set to 1 [2]; correspondently, its biochemical representation can be done by a set of DNA double strands test tubes $T[\alpha]_m ... T[\alpha]_1$

associated to the positions where bits are set to 1 [3]. An example of DNA double strand, representing an integer, is the following:

$$\updownarrow\underbrace{(aagctct^5)^i}_{S^i}\,aagctt\,\underbrace{(ctgcatg^5)^k}_{X^k}\,ctgcag\,\underbrace{(gaattgc^5t^5g^5c)^j}_{Y^i}\,\underbrace{gaattc}_{E_0}$$

where S^i encodes the test tube containing the strand and X^k the byte in which the molecular bit is contained; Y^j represents the offset into the byte and E_0 the end of each DNA strand. All the subsequences are linked by the restriction sites *aagctt*, *ctgcag* and *gaattc*, respectively for HandIII, PstI and EcoRI enzymes. According with this schemes, the first position is encoded with $\updownarrow\ (aagctct^5)^1aagctt(ctgcatg^5)^1ctgcag(gaattgc^5t^5g^5c)^1gaattc$ and so on for the successive ones. In order to simplify the biological operations we need, each sequence has different size and Y has to be $l-1$ times longer than X, where l is the maximum value of k. Thus, a generical value α is encoded by the set of test tube $T[\alpha]_m...T[\alpha]_1$ that contain all the integers of $X[\alpha]$.

3 Logical and Arithmetic Operations

3.1 Logical Operations

In [5] Weiss and Basu report *in-vivo* experimental results which examine the steady state behavior of cellular logic gates, with mRNA support. Here we propose the logical design for *in-vitro* logic gates, using DNA algorithms. Let $T[\alpha]_m...T[\alpha]_1$ and $T[\beta]_m...T[\beta]_1$ be the test tubes encoding binary strings α and β respectively, with m<<n. The **OR**, $\alpha \vee \beta$, is executed synthesizing the test tubes $T[\alpha]_i$ and $T[\beta]_i$, $\forall i = 1..n$, and mixing them together. The **XOR**, $\alpha \oplus \beta$, is executed extracting all the double strands that belong exclusively to $T[\alpha]_i$ or $T[\beta]_i$. The **AND**, $\alpha \wedge \beta$, is executed extracting all molecular bits that belong contemporarily to $T[\alpha]_i$ and $T[\beta]_i$. The **NOT**, $\neg\alpha$, is executed synthesizing a set of test tube $T_m...T_i$, applying the AND operation with $T[\alpha]_m...T[\alpha]_1$ and eventually extracting these sequences from the solutions. Each operation is performed efficiently whatever is the size of the input binary strings. Each algorithm requires a constant number of bio-steps [3], related to the number of test tubes used in the representation of binary numbers. Thus the expected number of bio-steps for each logical operation is $O(m)$, where m is the unfixed number of test tubes requested by molecular bits. If the number of bits is fixed the complexity is $O(1)$. Previous analyses carry out theoretical schemes where each gate has input and output strings composed by DNA test tubes, circuits are constituted by bio-steps, and the computational site is a biological laboratory.

3.2 Arithmetic Operations

As it is shown in [1], we can implement addition and multiplication operations by recursive procedures. Let $\alpha = \alpha_n...\alpha_1$ and $\beta = \beta_n...\beta_1$ be two binary

strings, and $X[\alpha] = \{i : \alpha_i = 1\}$ and $X[\beta] = \{j : \beta_j = 1\}$, it results:

$$\mathbf{Add}(\alpha, \beta) = Val(RecursiveAdd(X[\alpha], X[\beta])); \tag{1}$$

where

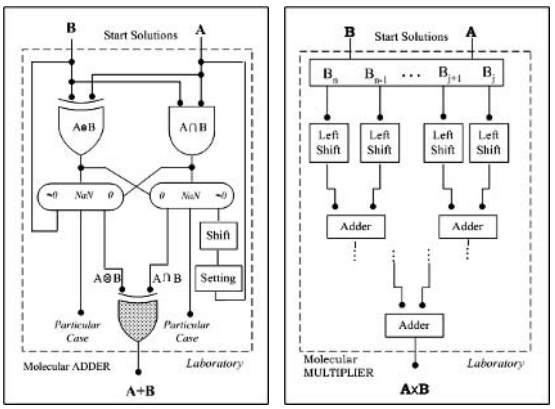
$$ReacursiveAdd(Y, Z) = \begin{cases} X & if \quad Z = \emptyset \\ Z & if \quad Y = \emptyset \\ RecursiveAdd((Y \oplus Z)(Y \cap Z)^{+}) & otherwise \end{cases}$$

The multiplication procedure can be realized using progressive additions of values, left shifted. So the multiplication operation results in:

$$\mathbf{Mul}(\alpha, \beta) = \mathbf{Add}(\{\, Val\ (X[\alpha] + (j-1))\}\beta_j = 1) \tag{2}$$

Subtraction and division are trivial consequences of them.

For each $T[\alpha]_i$ and $T[\beta]_i$ with $i = 1...m$, the addition is performed as follows:
Step1. Divide molecular bits in two different test tubes $T[\alpha \oplus \beta]_i$ and $T[\alpha \cap \beta]_i$. Check whether the set $T[\alpha \oplus \beta]_i$ or $T[\alpha \ \cap \ \beta]_i$ is empty. If that's true, then the set of test tubes $T[\alpha + \beta]_i$ are equal to the not empty test tubes. Else go to step2.
Step2. Shift on the left all the bits contained in $T[\alpha \cap \beta]_i$, that is increase by one position all the double strands in solution, producing $(X[\alpha] \cap X[\beta])^{+}$. Repeat this two simple steps until one of the set of test tube in step 1 is empty. Figure 1(a) shows a logical circuit that faithfully reproduces the molecular algorithm steps. The procedure is realized as follows. With the help of restriction enzyme *EcoRI*, cut all the double strands at their $3'$ end. Add the upper strands $\uparrow$ $attgc^5t^5g^5cgaattc$ with the ligation enzyme and attend that the polymerization process forms double strands of this kind: $\updownarrow (aagctct^5)^i aagctt(ctgcatg^5)^k ctgcag(gaattgc^5t^5g^5c)^{j+1} gaattc$. Thus each bit position in $T[\alpha \ \cap \ \beta]_i$ has been shifted on the left. To reorganize the solution in bytes, move surplus bits from a byte to the next and from a test tube to the next [3]. Restriction enzymes and ligation are used, enzyme *SalI* to increase X^k and *HandIII* to increase S^i. At the end of this process, the surplus bits will be: $\updownarrow\ (aagctct^5)^i aagctt(ctgcatg^5)^{k+1} ctgcag\ \ (gaattgc^5t^5g^5c)^1 gaattc$ and $\updownarrow (aagctct^5)^{i+1} aagctt(ctgcatg^5)^1 ctgcag(gaattgc^5t^5g^5c)^1 gaattc$.
For the multiplication first calculate all shifted values of α bits, in comparison to 1-bit$\in$ $X[\beta]$, then sum them as in a tree data structure. Thus progressive additions are not made with the same solutions but, concurrently with successive pairs of tubes. This procedure, besides to improve complexity, avoids that biological errors might affect DNA in solutions. Figure 1(b) shows a simple multiplier circuit. As shown in [3], the expected number of bio-steps, for the addition, is $O(m \cdot \log_2 n)$ and becomes $O(\log_2 n)$, if the number of bits is finite and limited to one test tube. Multiplication complexity is $O(m \cdot (\log_2 n)^2)$; it depends on the logarithmic number of addition in the tree. Also for multiplication it became $O(\log_2 n)^2)$ in the best case. Arithmetic operations can be described as conventional circuits.



(a) (b)

Fig. 1. Logical design of molecular circuits. (**a**) Molecular Adder: dark gates represent not DNA made logical selections. (**b**) Molecular Multiplier.

4 Floating Point Arithmetic

The DNA representation of floating numbers we propose is comparable to the IEEE 754 [4]. Using the DNA representation presented above, we can divide the set of test tubes as follow: $T[\alpha]_{sign}T[\alpha]_{exp}T[\alpha]_m...T[\alpha]_1$, where $T[\alpha]_{sign}$ encodes the sign of α. $T[\alpha]_{exp}$ the molecular bits for the exponent and $T[\alpha]_m...T[\alpha]_1$ the mantissa f. As it happens in IEEE 754 standard, we need to choose a set of DNA strings to represent a few of mathematical significative values ($0, \pm\infty$ and NaN) [3].

For the addition, compare α and β exponents (by *gel electrophoreses*) and increase the mantissa of the greater of $|exp_\alpha - exp_\beta|$ positions. The result is the left shift of the mantissa toward greatest positions. Perform integer addition with α and β mantissas and then normalize the result [3] comparing the new most significant bit with the old, and shifting it if smaller.

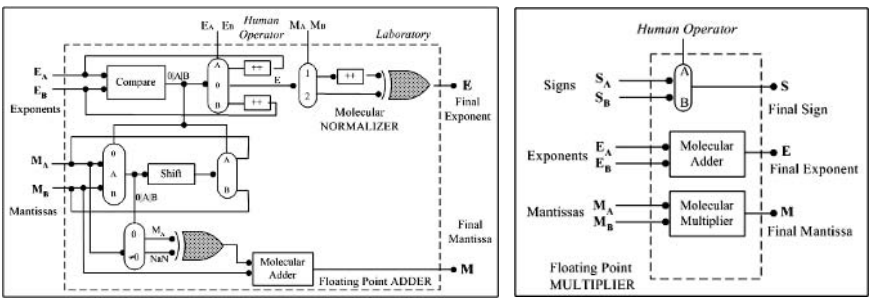
Multiplication is realized by adding α and β exponents and multiplying their mantissas (integer multiplication). At the end of this process compare $T[\alpha]_{sign}$ and $T[\beta]_{sign}$ and determine resulting sign as shown in Table 2. The

Table 2. Sign Choice

Choice	$T[\alpha \cdot \beta]_{sign}$	Sign
$T[\alpha]_{sign} = T[\beta]_{sign} = \phi$	$T[\alpha \cdot \beta]_{sign} = \phi$	Positive
$T[\alpha]_{sign} = T[\beta]_{sign} \neq \phi$	$T[\alpha \cdot \beta]_{sign} = \phi$	Positive
$T[\alpha]_{sign} \neq T[\beta]_{sign}$	$T[\alpha \cdot \beta]_{sign} \neq \phi$	Negative

expected bio-steps for the addition depend on the bio-steps in each computational step. Only one gel electrophoresis is required to choose the exponent, so the complexity is $O(1)$. The integer subtraction requires $O(\log_2 q)$ [3] and

the mantissa left shift $O(q)$, where q is the number of bits in the representation of the exponent. The integer addition takes $O(m \cdot \log_2 n)$ and the final gel electrophoresis and the increment $O(\log_2 q)$. In conclusion, the expected bio-steps are: $O(1) + O(\log_2 q) + O(q) + O(m \cdot \log_2 n)$ that is $O(m \cdot \log_2 n)$. They become $O(\log_2 n)$, if the bits of the mantissa are fixed and limited to one test tube. The expected bio-steps for the multiplication depends on the integer addition and multiplication: $O(\log_2 q) + O(m \cdot (\log_2 n)^2)$ namely $O(m \cdot (\log_2 n)^2)$. They become $O(\log_2 n)^2$, if the bits of the mantissa are fixed. Logical circuits for floating point adder and multiplier are respectively shown in Figure 2.



(a) (b)

Fig. 2. Floating Point Adder (**a**) and Multiplier (**b**). Dark sections represent logical selections, not implemented with Dna molecules.

5 Conclusion

The purpose of this paper was to introduce a concrete approach for the logical design of a real DNA-ALU, defeating limitations of results presented in [1], such as the fixed number of bits available for the user, and in [5], such as difficulties in building real biological circuits due to the *in-vivo* nature of its experimental basis.

References

1. Barua R. (2002) Binary Arithmetic for DNA Computer, 8^{th} International workshop on DNA-Based Computers: Dna Computing, pp. 124-132
2. Biswas S. (1998) Computing with Bio-Molecules. Theory and Experiment, Ed G. Paun
3. de Santis F., Iaccarino G. (2004) A DNA Arithmetic Logic Unit, WSEAS Transactions on Biology and Biomedicine, vol. **1**, October 2004, pp. 436-440
4. Kahan W. (1996) IEEE Standard 754 for Binary Floating Point Arithmetic, Lecture Notes on status of IEEE 754, University of California Berkeley CA.
5. Weiss R., Basu S. (2002) The Device Physics of Cellular Logic Gates, First Workshop on Non-Silicon Computing, Cambridge, Mass.

A New Programming Interface for Reinforcement Learning Simulations

Gabriela Şerban

Faculty of Mathematics and Computer Science, "Babeş - Bolyai" University, Cluj-Napoca, Romania

Abstract. The field of Reinforcement Learning, a sub-field of machine learning, represents an important direction for research in Artificial Intelligence, the way for improving an agent's behavior, given a certain feed-back about its performance. In this paper we propose an original interface for programming reinforcement learning simulations in known environments. Using this interface, there are possible simulations both for reinforcement learning based on the states' utilities and learning based on actions' values (Q-learning).

1 Introduction

Artificial Intelligence, one of the most recent disciplines appeared in the field of computer science, tries to understand the intelligent entities. But, unlike philosophy and psychology, which deal with the study of intelligence, too, the aim of Artificial Intelligence is building and as well understanding intelligent entities.

So, the field of machine learning represents one of the most recent challenges of the Artificial Intelligence, opening new perspectives for research in the field, and, of course, a way for achieving intelligent systems.

In our opinion, the most challenging field of Machine Learning is Reinforcement Learning, dealing with the problem of improving the behavior of an artificial agent based on feedback of its performance. Reinforcement Learning, seen as a method to solve problems of machine learning, based on new knowledge acquisition [1], is one of the ways that may form the basis for proving the machines' performances, being, in the same time, a field open to future researches.

2 The programming interface

The interface is realized in JDK 1.4, and is meant to facilitate to develop software for reinforcement learning in known environments.

There are three basic objects:agent, environment and simulation.

The agent is the learning agent and the environment is the task that it interacts with. The simulation manages the interaction between the agent and the environment, collects data and manages the display, if any.

In a reinforcement learning task, the interaction between the agent and the environment is continuous.

Generally, the inputs of the agent are perceptions about the environment, the outputs are actions, and the environment offers rewards after interacting with it.

In our case, because the agent acts in a known environment, its perceptions are, in fact, states from the environment.

In our model the reward received by the agent is a number; the environment, the actions and perceptions are instances of classes derived from the *IEnvironment*, *IAction* and *IState* interfaces respectively. The implementation of actions and perception can be arbitrary as long as they are understood properly by the agent and the environment. It is obvious that the agent and the environment has to be chosen to be compatible with each other in this way.

The interaction between the agent and the environment is handled in discrete time. We assume we are working with simulations. In other words there are no real-time constraints enforced by the interface: the environment waits for the agent while the agent is selecting its action and the agent waits for the environment while the environment is computing its next state.

We assume that the agent's environment is a finite Markov Decision Process.

For using the interface, the user has to define the specialized object classes *HisState*, *HisEnvironment* and *HisAgent*, by creating instances for each. The agent and the environment are then passed to a simulation object (*CSimulation*), that initializes and interconnects them. Then, CSimulation::init() will initialize and execute the simulation.

If the agent learns the states' utilities, it has to be derived from the *AgentUtility* class, otherwise, if it learns the actions' values (Q-learning) it has to be derived from the *AgentQValues* class.

For lack of space a prototypical example for a concrete agent may be found at the following URL: http://www.cs.ubbcluj.ro/ ~gabis/uml.pdf.

We have to mention that the learning algorithms used for implementing the behavior of the agent are the URU algorithm [6] for learning the states' utilities (values), respectively the SARSA algorithm [7] for Q-learning.

The SARSA algorithm is considered one of the most efficient methods for Q-learning; the URU algorithm (Utility-Reward-Utility) is an algorithm for learning the states' values, a variant of reinforcement learning based on Temporal Differences (we have proposed this algorithm in [6]).

3 The Design of the Interface

In this section we give a short description of the main entities used for designing the programming interface (for lack of space the UML diagram of

the interface may be found at the following URL: http://www.cs.ubbcluj.ro/~gabis/uml.pdf).

AGENT

The agent is the entity that interacts with the environment, receives perceptions (states) from it and selects actions. The agent learns by reinforcement and could have or not a model of the environment.

The main classes corresponding to agent entities are:

- *RLAgent*
  Ia an abstract class, the basic class for all the agents. The specific agents will be instances of subclasses derived from *RLAgent*. The main methods of this class are
  1. **void** *learning*(**double** *alpha*, **double** *gamma*, **double** *epsilon*, **int** *episodes*, **IEnvironment** *m*)
     Is the basic method which implements the learning algorithm of the agent in the environment *m*. There are given: the learning rate (*alpha*), the reward factor (*gamma*), the value for *epsilon* for the ϵ-Greedy selection mechanism, the number of training episodes (*episodes*).
     This method is not abstract, is concretely defined in the class *RLAgent* (indifferent what is the learning's type, the learning method is the same).
  2. **QValues** *next*(**IState** *s*, **IEnvironment** *m*) **ABSTRACT METHOD**
     This function gives the agent's policy for moving after learning. If the object having the type *QValues* returned by the method contains the state *snext* and the action *a*, it means that the agent's policy is the following: from the state *s*, the agent will choose the action *a* and will move to the state *surm*.
     This function has specific definition according to the agent's type, too.
- *AgentUtility*
  Is an abstract class, a subclass of *RLAgent*, being the entity which defines the behavior of an agent that learns by reinforcement based on the states' utilities. This class specializes the methods (2), (3) and (4) (defined in the superclass) according to learning based on states' utilities.
  The method *actions()*(from the superclass *RLAgent*) is not defined in this class (that is why the class is abstract), but will be defined in the class corresponding to the specialized agent created by the user (and who can be an instance of a class derived from *AgentUtility*).
- *AgentQValues*
  Is an abstract class, a subclass of *RLAgent*, being the entity which defines the behavior of an agent that learns by reinforcement based on the actions' values. This class specializes the methods (2), (3) and (4) (defined in the superclass) according to the Q-learning method.

The method *actions()*(from the superclass *RLAgent*) is not defined in this class (that is why the class is abstract), but will be defined in the class corresponding to the specialized agent created by the user (and who can be an instance of a class derived from *AgentQValues*).

ENVIRONMENT

The environment basically defines the problem to solve. It determines the dynamic of the environment, the rewards and controls, the ending of the training process. In our approach, the environment will have an implicit representation as a space of states (*IState*).

The main classes corresponding to the environment entity is:

- *IEnvironment*
  Is an interface, the basic class for all environments. The specific environments will be instances of subclasses derived from *IEnvironment*. The environment classes defined by the user (subclasses of *IEnvironment*) will give specialized definitions for the methods defined in the interface. The main method of the interface is the abstract method
  1. **IState** *next*(**IState** *s*, **IAction** *a*, **Integer** *r*)
     This method will be called by an instance of the class that simulates the learning (*CSimulation*), at each step of the simulation.
     This function determines the environment to make a transition from the current state *s* to a next state *surm*, after executing the specific action *a*. The state *surm* will be returned, the function supplying in the same time the reward *r* obtained after the transition.
     In the case that the action *a* could not be applied in the state *s*, the method returns *null*.

SIMULATION

The basic object is the interface *CSimulation*, that manages the interaction between the agent and the environment. Defines the *heart* of the interface, the uniform usage that all agents and environments are meant to conform to.

An instance of the simulation class is associated with an instance of an agent and an environment at the creation moment. This is made in the constructor of the class *CSimulation*. The class *CSimulation* keeps references to the instances of the agent and the environment. This facilitates cross-references of instances in case it is need.

The methods of this class are:

1. **void** *init*(**double** *alpha*, **double** *gamma*, **double** *epsilon*, **int** *episodes*)
   Is the method that initializes the simulation with the given parameters (is the function that starts the learning process of the agent).
2. **void** *policy*(**PrintStream** *ps*)
   Is the method that gives the moving policy for the agent, obtained at the end of the simulation (after the training process).

```
public class CSimulation
{
    private RLAgent a; //reference to the agent's instance
    private IEnvironment m; //reference to the environment's instance
    ...
}
```

4 Experiment

For experimenting the use of the interface, we have considered the problem of a path-finding robot, whose goal is to learn (by reinforcement) to come out from a maze (moving from an initial to a final state).

We mention that we made experiments for different rectangular environments (with more states) and the learning process works well.

5 Further Work

Further works for generalizing the interface would be made in the following directions:

- the study of the case in which the environment in which the agent acts is unknown (the situation in which the agent has to develop also a model of the environment);
- the study of the case in which the agent's environment is a Hidden Markov Model [5];
- to generalize the interface for multi-agent systems (the situation in which several agents are learning by reinforcement a certain goal).

References

1. Bergadano, F., Giordana, A. (1991) Machine Learning. Ellis Horwood Limited, Chichester, England
2. Fiechter, C.N. (1994) Efficient Reinforcement Learning. Proceedings of the 7th annual ACM Conference on Computer Learning theory, 88–97
3. Harmon, M., Harmon, S. (2000) Reinforcement Learning - A Tutorial. Wright State University, www-anw.cs.umass.edu/∼mharmon/rltutorial/frames. html
4. Serban, G. (2001) A Reinforcement Learning Intelligent Agent. Studia Universitatis "Babes-Bolyai", Informatica **XLVI(2)**, 9–18
5. Serban, G. (2001) Training Hidden Markov Models - a Method for Training Intelligent Agents. Proceedings of the Second International Workshop of Central and Eastern Europe on Multi-Agent Systems, Krakow, Poland, 267–276
6. Serban, G. (2003) A New Reinforcement Learning Algorithm. Studia Universitatis "Babes-Bolyai", Informatica **XLVIII(1)**, 3–14
7. Sutton, R., Barto, A., G. (1998) Reinforcement learning. The MIT Press, Cambridge, England

Anomaly Detection System for Network Security: Immunity-based Approach

Franciszek Seredyński[1,2], Pascal Bouvry[3], and Dawid R. Rutkowski[4]

[1] Polish-Japanese Institute of Information Technology, Koszykowa 86, 02-008 Warsaw, Poland
[2] Institute of Computer Science, Polish Academy of Sciences, Ordona 21, 01-237 Warsaw, Poland
[3] Faculty of Sciences, Technology and Communication, Luxembourg University 6, rue Coudenhove Kalergi, L-1359 Luxembourg-Kirchberg, Luxembourg
[4] Warsaw University of Technology, Nowowiejska 15/19, 00-665 Warsaw, Poland

Abstract. In this paper we present architecture of recently built experimental anomaly detection system based on the paradigm of artificial immune system and working in a network environment. We show how network traffic data are mapped into antibodies or antigens of artificial immune system and how similarities between signatures of attackers and antibodies are measured. We present an example of the work of the system in the real network environment.

1 Introduction

Designing intrusion detection systems (IDS) to ensure security of information and computational resources is one of current research activities in the area of computer networks. Anomaly detection, which is one of major approaches in intrusion detection and is a subject of this study, relays on building models from network data and discovering variations from the model in the observed data.

Currently used commercial IDS systems are usually signature-based (see, e.g. [6]). However, this approach does not prevent attacks, which were not described earlier in training data. For this reason another approaches are currently studied to design IDS. A statistical approach, data mining and machine learning methods or using nature inspired algorithms (see, e.g. [2]) are examples of recently applied techniques for designing IDS.

One of the promising nature-inspired techniques used for IDS are artificial immune systems (AIS) [5,1,3] and we follow this approach. The purpose of this project was to build an experimental IDS working in a real environment and serving as a tool to study AIS techniques suitable for anomalies detection.

The remainder of the paper is organized as follows. The next section presents a background on AIS. Section 3 contains a description of AIS-based anomaly detection system (ADS) including issues of traffic data coding and the choice of matching functions. Section 4 describes methods used for generation of antibodies. Section 5 presents results of an experiment conducted in a real network environment and last section concludes the paper.

2 Artificial Immune Systems

AIS is a computational technique inspired by ideas coming from immunology. It has recently become one of the most popular tools applied in the field of computer security [5], to solve problems in domains of function optimization or combinatorial optimization [4].

While a general computational scheme of AIS is currently a subject of a research, two basic notions - *antigen* and *antibody* are well established. Antigens are foreign invaders, which attack a system. Antibodies are a part of the system, responsible for detection of antigens. Antibodies detect antigens by matching them. A number of antibodies is much smaller than the number of antigens, so matching in never perfect. One of purposes of AIS-based system is to develop relatively small number of antibodies, which are able to detect a big number of antigens, including antigens that have never been seen before.

3 AIS-based Anomaly Detection System

3.1 A General Overview of the System

The general scheme of AIS-based ADS working in a single node of a network is presented in Fig. 1. It is assumed that external users can have an access to resources and services which are opened to them, using network communication system by sending and receiving messages, which constitute *network traffic*. Incoming traffic is only the source, which contains both an authorized access and attempts to unauthorized access to resources and services.

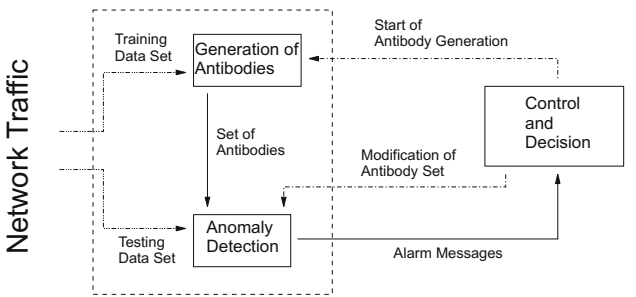


Fig. 1. General scheme of anomaly detection system

Incoming traffic should be parameterized in such way to be able to define patterns of both malicious behavior corresponding to invaders and authorized behavior. A part of network traffic containing only authorized behavior (*training data set*) should be available. We assume that the system is able to work in two modes: *learning mode* and *normal operating mode*. In the

learning mode data from training data set are used to generate a set of antibodies, which model authorized behavior of users. In the normal operating mode *testing data set* - a part of regular incoming traffic are used to tune the system, detect anomaly and inform *Control and Decision System* about suspicious behavior.

It is assumed that the node of the system is a part of TCP/IP network, and messages of a network traffic consist of a number of packets created and sent according to TCP/IP protocols.

3.2 Coding Antibodies and Antigens

What to analyze in incoming network traffic to be able to detect intruders is an open research issue. In the system we have applied two recently emerged ideas. The first one [1] is to use a set of traffic parameters such as *a number of bytes per second* (*Bps*), *a number of packets per second Pps*, and *a number of ICMP packets per second*. These values are calculated and averaged using small time intervals, using the idea of moving window. It means that both antibodies and antigens can be represented in the form of vectors in 3-D space.

The second one is to use [3] full headers of TCP/IP protocols. In the implementation of the system this idea was reduced to the analysis only TCP SYN packets. Such packets are sent to establish TCP communication, and can be considered as good representatives of TCP packets. Under this approach both antibodies and antigens will be represented by binary arrays corresponding to headers of TCP SYN packets.

3.3 Similarity Measures Between Antibodies and Antigens

A decision about the degree of matching antibodies and antigens is taken on the base of evaluation of a distance between corresponding antibodies and antigens. Euclidean $d(x, y)$ or Hamming distances $f_H(X, Y)$ are used as measures of similarities between antibodies and antigens, where $x = \{x_1, ..., x_n\}$ and $y = \{y_1, ..., y_n\}$ are vectors from $n-$ dimensional Euclidean space, and X and Y are binary arrays.

If a distance calculated for a given pair of antigen and antibody exceeds some threshold, the part of network traffic corresponding to the antigen is considered as an intruder.

4 Generating Antibodies

An important issue for any AIS-based ADS is a generation of a set of antibodies, which will represent a legal network traffic. Such a set should be enough small, and each antibody should cover relatively large part of a space corresponding to the legal traffic. Three methods of generation of antibodies have been implemented in the system.

4.1 Positive Characterization Technique

The *positive characterization* (PC) technique [1] assumes that information about the legal traffic is directly used to create detectors (antibodies). A number of antibodies are equal to the number of information units describing the legal traffic. The method is simple and effective in detecting anomalies, but increasing the number of detectors may cause the increase the computational costs of detecting.

4.2 Random Generation of Antibodies

In this method [3] information about the legal traffic is used only as a test set to build a population of detectors, which are created in a random way. A candidate for antibody is created randomly and next compared with each representative of the legal traffic. If the candidate covers at least one example of the legal traffic than it is included into the created population of antibodies.

4.3 Negative Characterization Technique

Negative characterization (NC) technique [1] proposes to consider antibodies as rules of the following form:

R^k : **if** $x_1 \in \langle min_1^k, max_1^k \rangle$ **and ... and** $x_n \in \langle min_n^k, max_n^k \rangle$ **then** ***anomaly***,

where R^k is k-th rule, which can be interpreted as n-dimensional hypercube with min_i^k and max_i^k as boundaries of each i-th dimension, and $\{x_1, ..., x_n\}$ are parameters of a suspected traffic activity (antigen).

Rules-antibodies are generated in such a way to cover a part of space not belonging to the legal traffic. Examples of the legal traffic can be seen as hyper spheres in the space. Because not all points of the legal traffic are known, different hyper radiuses of self-examples are used and rules-hyper cubes are generated with different boundaries. They can intersect and create multilevel subspaces corresponding to different levels of anomalies.

5 Experiment

This experiment has been conducted with a network traffic data collected during 9 days in a new installed server in the local network of Institute of Computer Science PAS (IPI PAN), Warsaw. The data contained about 4.4 mln. of packets.

PC method and Euclidean distance were used for analysis of the traffic. The best results obtained using analysis of TCP SYN packets in 10-minute periods. The distances between parameters of the traffic considered as legal (11 detectors) and the whole traffic (1198 antigens) are shown in Fig. 2. The frequency of attacks is high. There are visible periods of activities in midday, and two attacks conducted from a private computer (the distance equal to 31.5 and 42.4 in two subsequent 10 min. periods of time).

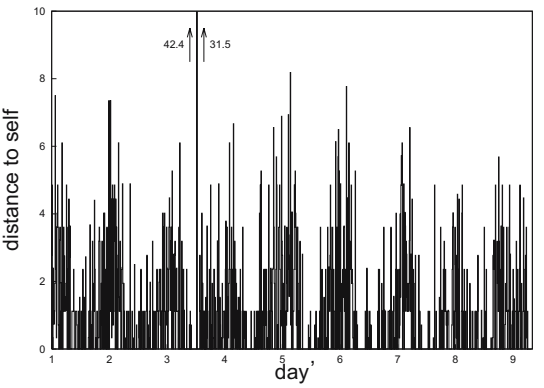


Fig. 2. Discovery of anomalies in a local network of IPIPAN

6 Conclusions

The architecture of a recently built IDS has been presented in the paper. The system was designed using AIS principles, what assumes applying corresponding methods of coding traffic and methods of generation of antibodies. Results of the experiment conducted in a real network environment have shown promising capabilities of the system to detect anomalies.

References

1. D. Dasgupta, F. Gonzales, An Immunity-based Technique to Characterize Intrusions in Computer Networks, *IEEE Trans. on Evolutionary Computation*, vol. 6, N3, June 2002, pp. 281-291
2. G. V. Dozier, D. Brown, J. Hurley, C. Cain, Vulnerability Analysis of AIS-Based Intrusion Detection Systems via Genetic and Particle Swarm Red Tearns, in *CEC 2004*, pp. 111-116
3. P. K. Harmer, P. D. Wiliams, G. H. Gunsch, G. B. Lamont, An Artificial Immune System Architecture for Computer Security Applications, *IEEE Trans. on Evolutionary Computation*, vol. 6, N3, June 2002, pp. 252-279
4. E. Hart, P. Ross, J. Nelson, Producing robust schedules via an artificial immune system, *Proc. IEEE Int. Conf. Evolutionary Computing*, May 1998, pp. 464-469
5. S. A. Hofmeyr, S. Forrest, Architecture for an artificial immune system, in *IEEE Trans. on Evolutionary Computation*, vol. 8, N4, 2000, pp. 443-473
6. M. Roesch, Snort - Lighweight Intrusion Detection for Networks, in *13th Systems Administration Conference - LISA 99*, 1999

Importance of TDS Attribute in Computer Assisted Classification of Melanocytic Skin Lesions

Aleksander Sokołowski

Rzeszów University of Technology, ul. W. Pola 2, 35-959 Rzeszów, Poland
alex5@prz.edu.pl

Abstract. In the paper the new algorithm and results of its application to designation the importance of the TDS attribute in identifying the melanocytic skin lesions are described. The algorithm consists of decision table, TDS belief net, and the nearest neighbor method applied to the bases, which was obtained by reduction the number of attributes from thirteen to four. The obtained results show that this algorithm is very promising.

Keywords: decision tables, belief networks, nearest neighbor method, classification of skin lesions, ABCD formula, TDS attribute

1 Introduction

The approach mentioned in the title of this paper has been applied to databases being collections of cases describing melanoma data[1]. Melanoma is routinely diagnosed with help of so-called **ABCD** formula (**A** indicates **A**symmetry, **B** – **B**orders, **C** – **C**olors, and **D** – **D**iversity of structure) [1,2]. The above databases were used in the research on optimization of the above formula with help of data mining system LERS [3,4]. The results were reported in [5–8]. Additionally, there were presented the results of the classification of melanocytic skin lesions (using the same very bases), based on neural nets [9], and on belief nets [10]. The databases include the attributes that are divided into five categories: $\langle \boldsymbol{Asymmetry} \rangle$, $\langle \boldsymbol{Border} \rangle$, $\langle \boldsymbol{Color} \rangle$, $\langle \boldsymbol{Diversity} \rangle$, and $\langle \boldsymbol{TDS} \rangle$ (Total Dermatoscopy Score). The domain of $\langle \boldsymbol{Asymmetry} \rangle$ encloses three different values: symmetric spot, single axial symmetry, double axial symmetry. $\langle \boldsymbol{Border} \rangle$ is the numerical attribute, with value changing from 0 to 8 (integer values). The attributes $\langle \boldsymbol{Asymmetry} \rangle$ and $\langle \boldsymbol{Border} \rangle$ are the functions of single variable. The attributes $\langle \boldsymbol{Color} \rangle$ and $\langle \boldsymbol{Diversity} \rangle$ are the functions of several variables. $\langle \boldsymbol{Color} \rangle$ is the function of six variables: black, blue, dark brown, light brown, red, and white [2]. Similarly, $\langle \boldsymbol{Diversity} \rangle$ is

[1] Investigated databases were supplied from Chair of Expert Systems and Artificial Intelligence, University of Information Technology and Managements in Rzeszow, Poland.

the function of five variables: pigment dots, pigment globules, pigment network, structureless areas, and branched streaks. There are thirteen attributes in the aggregate. ⟨***TDS***⟩ is an additional attribute defined as follows:

$$TDS = 1.3*Assymetry+0.1*Border+0.5*\sum Colors+0.5*\sum Diversities, \tag{1}$$

where for the ⟨**Asymmetry**⟩ attribute the value of symmetric spot counts as 0, single axial symmetry counts as 1, and double axial symmetry counts as 2. The symbol $\sum Colors$ represents the sum of all six variable values of color attribute, and $\sum Diversities$ represents the sum of all five variable values of diversity attribute. The variables of the color and diversity attributes take the value 0, when there isnt suitable feature and the value 1, when there is suitable feature. There are fourteen attributes this way. The ⟨**TDS**⟩ is the auxiliary attribute. ⟨**TDS**⟩ is the important attribute for generating decision trees, particularly. It appears that the ⟨**TDS**⟩ attribute significantly reduces the decision trees generated on the ground of the above databases [11], for example. The aim of the paper is to verify the validity of ⟨**TDS**⟩ in other algorithm than generating decision trees one. In order to simplify calculations and the structure of databases the new approach is proposed. New approach consists in reduction of the attribute number, as the first step. Two attributes ⟨***Color***⟩, and ⟨***Diversity*** ⟩ undergo the reductions. Six variables of the ⟨***Color***⟩ reduce to single variable as follows:

$$f_c(< Color >) = \sum_{i=1}^{6} 2^{i-1} c_i, \tag{2}$$

where f_c is the function transforming ⟨***Color***⟩ as six dimensional variable to integer value. c_1, c_2, c_3, c_4, c_5, and c_6 represent the six binary values of the *black*, *blue*, *dark brown*, *light brown*, *red*, and *white* variables, respectively. The shape of the equation (2) shows that the value of ⟨***Color***⟩ treated as the value of binary code and the function f_c transforms this value from binary to decimal codes. The attribute ⟨***Diversity*** ⟩ is treated in the same manner:

$$f_c(< Diversity >) = \sum_{i=1}^{5} 2^{i-1} c_i, \tag{3}$$

where d_1, d_2, d_3, d_4, and d_5 represent the five binary values of the *pigment dots*, *pigment globules*, *pigment network*, *structureless areas*, and *branched streaks* variables, respectively. This means that thirteen attributes are reduced to four attributes, and ⟨***TDS***⟩ is taken into account as additional attribute.

2 Numerical studies

It can be applied Bayesian network method to such database. The simple network including the four above attributes and disregarding any interdependence between nodes has been used in numerical studies. Nevertheless this network works properly. Every pathological case is correctly recognized, if it exists already in the database, otherwise the case is not classified. In order to classify them it can be applied, for example, the Dirichlet probability distribution to calculate the approximation of all the above attributes distributions [10]. The decision table is however simpler method to classify the melanoma cases. It works in the same way as the Bayesian net. It means that every unseen case is not classified. The additional attribute $\langle \boldsymbol{TDS} \rangle$ has been applied to classify all such cases. In order to relate the value of TDS to the particular decision the simple belief network has been constructed. Additional to this scheme it has been applied minimum distance method. It means that when the melanoma case isn't classified with TDS belief net, then the decision is just the same as for the nearest value of TDS. This algorithm was applied to the two modified databases. The base E410144 including 410 cases, 14 attributes, and 4 concepts was transform to the base E410044 (410 cases; 4 attributes, and 4 concepts). The base E410044 has been used as the base for learning. The second base E112144 was translated to the base E112044. This base was used to testing this algorithm. Table 1 presents the results obtained with this algorithm. It follows from it that the error rate is equal to 14.3 %. It is pretty large value. I mean that the error rate is important as the quantity estimating the quality of the algorithm used for the bases including the melanoma cases. But the possibility of designating the error sources is matter of major importance. The sixteen errors inserted in Table 1 indicate thirteen cases of wrong recognize of the blue-nevus as the benign-nevus categories, three cases of wrong recognize of the malignant-nevus as the suspicious-nevus categories, and one case of wrong recognize of the benign-nevus as the suspicious-nevus categories. All above errors are not important for the patients, because they point out the not dangerous illness in the case of first thirteen errors, and they force to visit the doctor in the case of the remaining errors. But it remains the question about the sources of all above errors.

In order to find the error sources it is proper to look at the distributions of the TDS values for the E410144 base. Table 2 shows some values of TDS which indicate more than one category. Column TDS contains the values of TDS. Columns for categories Benign-nevus, Blue-nevus, Suspicious-nevus, and Malignant-nevus categories consist the values, which denotes the number of cases for particular category. The first row of Table 2 denotes the value of 2 for TDS, two cases for Benign-nevus category, and two cases for Blue-nevus category. This means that there are two cases for Benign-nevus category, and ten cases for Blue-nevus category, which have the same value of TDS. Most of such cases are for the two categories: Benign-nevus and Blue-nevus. As it has

Table 1. The results of classification melanoma cases by means of decision table and decision table supported with TDS belief network and with nearest TDS value method for unseen cases.

Algorithm	**Number of correctly classified cases**	**Number of incorrectly classified cases**
Decision table	16	96*
Decision table + TDS belief network + nearest TDS value method	96	16**

* not classified cases
** incorrectly classified cases

been earlier written such errors isnt dangerous for patients. Very dangerous instance is presented in the last row of Table 2. The value of TDS is equal to 6.4, and there are one case for Benign-nevus and three cases for Malignant-nevus. The error resulting from it can be very dangerous for patients.

Table 2. Distribution of selected values of TDS for the E410144 database.

TDS	**Benign-nevus**	**Blue-nevus**	**Suspicious-nevus**	**Malignant-nevus**
2	2	10	0	0
2.5	12	18	0	0
5.6	0	0	2	10
6.4	1	0	0	3

3 Conclusions

The decision tables method for the melanoma database works correctly, when all possibly cases for values of attributes have been included in the database. If not, additional algorithms should be applied to recognize the category of decision. The application of TDS parameter gives very good results in some cases, but it is the source of potential errors. It results from the definition of the TDS attribute (1). Two series of the values of the thirteen attributes (without TDS), for example: 1, 6, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, and 1, 6, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1 give both the same value of TDS equal to 6.4, but they lead to two different decisions. It seems that TDS is very powerful tool in artificial intelligence research, but it requires further study. TDS is the matter of research now, and the results of optimization of the ABCD formula have been already published [12].

4 Acknowledgements

The author would like to thank Prof. Z. S. Hippe for access to all above melanoma databases, and for help in preparing this paper.

References

1. R. J. Friedman, D. S. Rigel, A. W. Kopf, Early detection of malignant melanoma: the role of physician examination and self-examination of the skin, CA Cancer J. Clin., 35 (1985) 130-151.
2. J. W. Stolz, O. Braun-Falco, P. Bilek, A. B. Landthaler, A. B. Cogneta, Color Atlas of Dermatology, Blackwell Science Inc., Cambridge, MA (1993).
3. J. W. Grzymala-Busse, LERS A system for learning from examples based on rough sets, in Intelligent Decision Support. Handbook of Application and Advances of the Rough Sets Theory. R. Slowinski (ed.), Kluwer Academic Publishers, Dordrecht, Boston, London (1992) 3-18.
4. J. W. Grzymala-Busse, A new version of the rule induction system LERS, Fundamenta Informaticae 31 (1997) 27-39.
5. A. Alvarez, F. M. Brown, J. W. Grzymala-Busse, and Z. S. Hippe, Optimization of the ABCD formula used for melanoma diagnosis, Proc. of the II PWM2003, Int. Conf. On Intelligent Information Processing and WEB Mining Systems, Zakopane, Poland, June 2-5 (2003) 233-240.
6. J. P. Grzymala-Busse, J. W. Grzymala-Busse, and Z. S. Hippe, Melanoma prediction using data mining system LERS, Proceedings of the 25th Anniversary Annual International Computer Software and Applications Conference COMPSAC 2001, Chicago, IL, October 8-12 (2001) 615-620.
7. J. W. Grzymala-Busse, and Z. S. Hippe, Postprocessing of rule sets induced from a mela-noma data sets, Proc. of the COMPSAC 2002, 26th Annual International Conference on Computer Software and Applications, Oxford, England, August 26-29 (2002) 1146-1151.
8. J. W. Grzymala-Busse, and Z. S. Hippe, A search for the best data mining method to pre-dict melanoma, Proceedings of the RSCTC 2002, Third International Conference on Rough Sets and Current Trends In Computing, Malvern, PA, October 14-16 (2002) Springer-Verlag, 538-545.
9. A. Sokołowski, A. Dereń, The study of hierarchy importance of descriptive attributes in computer assisted classification of melanocytic skin lesions, L. Rutkowski et al. (Eds.): Proceedings of ICAISK 2004, Zakopane, Poland, June 7 11, LNAI 3070, Springer-Verlag, 1050-1055.
10. T. Mroczek, J. W. Grzymala-Busse, Z. S. Hippe, Rulet from relief networks: a rough set approach, S. Tsumoto et al. (Eds.): Proceedings of RSCTC 2004, Uppsala, Sweden, June 1-5, LNAI 3066, Springer-Verlag 483-487.
11. J. W. Grzymala-Busse, Z. S. Hippe, S. Bajcar, A. Bak, A. Sokołowski, Decision Threes a method to support medical diagnoses exemplified by a case of melanocytic spots on the skin, Dermatologia Kliniczna, 5 (4) (2003), 201-209 (in Polish).
12. R. Andrews, S. Bajcar, J. W. Grzymala-Busse, Z. S. Hippe, C. Whiteley, Optimization of the ABCD formula for Melanoma diagnosis using C4.5, a data mining system, S. Tsu-moto et al. (Eds.): Proceedings of RSCTC 2004, Uppsala, Sweden, June 1-5, LNAI 3066, Springer-Verlag 630-636.

A Rules-to-Trees Conversion in the Inductive Database System VINLEN

Tomasz Szydło[1], Bartłomiej Śnieżyński[1], and Ryszard S. Michalski[2,3]

[1] Institute of Computer Science, AGH University of Science and Technology, Kraków, Poland
[2] Machine Learning and Inference Laboratory, George Mason University, Fairfax, VA, USA
[3] Institute of Computer Science, Polish Academy of Science
e-mail: tomsz@student.agh.edu.pl, sniezyn@agh.edu.pl, michalski@gmu.edu

Abstract. Decision trees and rules are completing methods of knowledge representation. Both have advantages in some applications. Algorithms that convert trees to rules are common. In the paper an algorithm that converts rules to decision tree and its implementation in inductive database VINLEN is presented.
Keywords: machine learning, decision tree learning, decision making, inductive databases.

1 Introduction

Decision trees and decision rules are very common knowledge representation used in machine learning and data mining. Algorithms for learning decision trees are simple to implement, and relatively fast. Algorithms for learning decision rules are more complex, but resulting rules may be easier to interpret.

To exploit advantages of both knowledge representations, learned decision trees are often converted into rules. Opposite conversion is also possible. The AQDT-2 method proposed by Imam and Michalski is able to convert rules into a decision tree that is optimized according to a given criterion of decision tree optimality [1].

A major advantage of such a conversion is that it allows an adaptation of the decision tree to changing conditions. To explain this advantage, note that decision trees are built under certain assumptions. If these assumptions are not satisfied, e.g., the cost of measuring of some attributes changes, it is not possible to measure some attribute, new data become available, or the probability distribution of classes changes, the learned decision tree needs to be modified. Once a tree is built, however, such a modification is difficult or even impossible without learning the tree from scratch again.

The AQDT-2 program takes a set of rules learned from examples, and generates a decision tree tailored to the given decision task. Generating a tree from rules is faster than generating a decision tree from examples.

The AQDT-2 method was implemented by Imam as an independent program. Nowadays, we can observe a trend of building integrated machine learn-

ing environments (see [5,7,6]). Such an approach not only makes performing machine learning experiments easier, but also provides an opportunity to apply different methods by users who are not specialists in this domain.

This paper describes an implementation of the AQDT-2 method as a module of the inductive database system VINLEN that aims at integrating conventional databases with a range of inductive inference capabilities [2]. The following sections describe the AQDT-2 method, its implementation as a VINLEN module, and some preliminary results from testing the module as a VINLEN operator.

2 AQDT-2 as a Knowledge Generation Operator

As it was mentioned above, AQDT-2 is implemented as a part of inductive database VINLEN [2]. In this system database and knowledge base form knowledge system (KS), standard database operators are completed by knowledge generation operators (KGOs) that operate on elements of KS.

There are several KGOs implemented in the system so far. The most important is rule induction operator that is an implementation of AQ algorithm. It transforms database table into rulefamily (set of attributional rules expressed using Attributional Calculus [4]). AQDT-2 is another operator. It takes a rulefamily as an input and produces a decision tree that can be stored in a knowledge base. Classic decision tree induction algorithm – C4.5 is also implemented to make comparison with AQDT-2 in the future.

Generated decision tree is presented in a result window, where user has possibility to expand or collapse nodes of the tree, what is useful for watching some branches of the tree in different level of abstraction. There is also possibility to print and copy the tree to the clipboard.

3 AQDT-2 Algorithm

In this section the AQDT-2 algorithm is described. It builds a decision tree from a set of attributional rules instead of examples. Algorithm is presented below:

Input: A family of attributional rulesets.
Output: A decision tree.
Step 1. Evaluate each attribute occurring in the ruleset using the LEF attribute ranking measure (see below). Select the highest ranked attribute. Suppose it is attribute A.
Step 2. Create a node of the tree, and assign to it the attribute A. Create as many branches from the node, as there are legal values of the attribute A, and assign these values to the branches.

Step 3. For each branch, associate a group of rules from the ruleset context which contain a condition satisfied by the value assigned to this branch. If there are rules in the ruleset context that do not contain attribute A, add these rules to all rule groups associated with the branches.

Step 4. If all the rules in a ruleset context for some branch belong to the same class, create a leaf node and assign to it that class. If all branches of the tree have class assigned, stop. Otherwise, repeat steps 1 to 4 for each branch that has no class assigned.

The difference between AQDT-2 and methods of learning decision trees from examples is that it chooses attributes that are assigned to the nodes using criteria based on the properties of the input attributional rules. At each step, algorithm chooses the attribute from available set by determining the attribute utility in the given set of attributional rules. The attribute (test) utility is based on five elementary criteria: cost, disjointness, importance, value distribution, and dominance [1].

The above criteria can be combined into one general test ranking measure using the "lexicographic evaluation functional with tolerances" – LEF [3]. LEF combines two or more elementary criteria by evaluating them one by one (in the order defined by LEF) on the given set of tests. A test passes to the next criterion only if it scores on the previous criterion within the range defined by the tolerance. In AQDT-2 method the following LEF is used:

$$< C, \tau_1, D, \tau_2, I, \tau_3, V, \tau_4, Do, \tau_5 > \tag{1}$$

where C, D, I, V, Do represent cost, disjointness, importance, value distribution, and dominance; τ_1, τ_2, τ_3, τ_4 and τ_5 are tolerance thresholds.

The decision tree can be generated in *Compact Mode* or *Normal Mode*. In *Normal Mode* standard decision trees are generated: each branch has one specific attribute and value assigned. In *Compact Mode* a decision tree may contain nodes representing conditions expressed in attributional calculus (e.g. internal disjunction in the form $x = v_1$ or v_2) that are generated using selectors appearing in rules.

4 Example

In this section a simple example of AQDT-2 method application is presented. A robot domain with the following nine nominal attributes is used: *head*, *body*, *smile*, *holding*, *height*, *antenna*, *jacket tie*, and *group*. The last one is a target attribute with three possible values: *bad*, *do_not_know*, *good*. Rules generated by AQ21 KGO are presented below:

```
A robot is unfriendly, if
    - its head is square or triangular,
      and its body is square or round;      (r1)
A robot is unclassified, if
```

```
    - its head is pentagonal;                (r2)
A robot is friendly, if
    - its body is round, or                  (r3)
    - its head is square or triangular,
      and its body is triangular;            (r4)
```

From these rules two trees are generated using AQDT-2 method in compact mode and standard mode. First one has six nodes, four of them are leafs. It looks as follows:

```
IF robot's head is pentagonal THEN it is unclassified
IF robot's head is round THEN it is friendly
IF robot's head is square or triangular THEN
     IF robot's body is round or square THEN it is unfriendly
     IF robot's body is triangular THEN it is friendly
```

We show how it is constructed. At the beginning, attribute *head* with the highest rank is chosen (using LEF method mentioned above) and a node with this attribute assigned is created. Compact mode causes that whole selectors are used to assign values to branches. *head* appears in three selectors, therefore three new nodes are created with branches with the following values assigned: `pentagon` (r2), `round` (r3), and `square or triangle` (r1, r4). Rule labels in parenthesis show which rules are copied to the new nodes. Two of these nodes are recognized as leafs. Third one is processed recursively because rules copied there belong to different classes. Attribute *body* is chosen (from rules r1 and r4). There are two values in selectors containing this attribute, hence two nodes are created with values `round or square` (r1), and `triangle` (r4) assigned to branches. These nodes are leafs, therefore tree construction is finished.

In the normal mode there are always as many branches as there are values defined in an attribute domain. Therefore tree is bigger. It has eleven nodes and eight leaves. It is presented below:

```
IF robot's head is pentagonal THEN it is unclassified
IF robot's head is round THEN it is friendly
IF robot's head is square THEN
     IF robot's body is round THEN it is unfriendly
     IF robot's body is square THEN it is unfriendly
     IF robot's body is triangular THEN it is friendly
IF robot's head is triangle THEN
     IF robot's body is round THEN it is unfriendly
     IF robot's body is square THEN it is unfriendly
     IF robot's body is triangular THEN it is friendly
```

5 Conclusion and Further Research

The goals of this project are to develop an efficient and versatile program for transforming attributional rules learned to optimized decision trees and

to integrate it with the VINLEN inductive database system. The generated tree is stored in the VINLEN's knowledge system, and used as an input to other operators.

In the near future, we would like to add new features to the module, such as the ability for edit manually the decision tree and to test it on a database of examples. We would also like to develop an operator that produces code in a given programming language that implements the generated decision tree.

Interesting results of comparison of AQDT-2 and C4.5 programs can be found in [1]. Imam and Michalski tested both programs on several datasets. The following properties of the generated decision trees were measured: the number of nodes, accuracy, and execution time. Decision trees obtained using AQDT-2 tended to be simpler and had higher predictive accuracy. In further research, we plan to conduct more experiments that test the decision rules to decision tree conversion on a number of problem domains.

6 Acknowledgments

The authors thank Janusz Wojtusiak for help with using AQ21 and tuning it to produce better rules.

References

1. I. Imam and R. S. Michalski. An empirical comparison between learning decision trees from examples and from decision rules. In *Proc. of the Ninth International Symposium on Methodologies for Intelligent Systems (ISMIS-96)*, Zakopane, Poland, 1996.
2. K. A. Kaufman and R. S. Michalski. The development of the inductive database system VINLEN: A review of current research. In M. Kłopotek et al., editor, *Intelligent Information Processing and Web Mining*, Advances in Soft Computing, pages 393–398, Zakopane, Poland, 2003. Springer.
3. R. S. Michalski. AQVAL/1-computer implementation of a variable-valued logic system VL1 and examples of its application to pattern recognition. In *Proceeding of the First International Joint Conference on Pattern Recognition*, Washington, D.C., USA, 1973.
4. R. S. Michalski. *Attributional Calculus: A Logic and Representation Language for Natural Induction*. Reports of the Machine Learning and Inference Laboratory, MLI 04-2. George Mason University, 2004.
5. I. Mierswa, R. Klinkberg, S. Fischer, and O. Ritthoff. A Flexible Platform for Knowledge Discovery Experiments: YALE – Yet Another Learning Environment. In *LLWA 03 - Tagungsband der GI-Workshop-Woche Lernen - Lehren - Wissen - Adaptivitat*, 2003.
6. M. Staudt, J. U. Kietz, and U. Reimer. A data mining support environment and its application on insurance data. In *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*. AAAI Press, 1998.
7. I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, 1999.

Part VII

Invited Session: Syntactic Parsing and Machine Learning

Deep Parser for Free English Texts Based on Machine Learning with Limited Resources

Marek Łabuzek and Maciej Piasecki

Wrocław University of Technology, Computer Science Department,
Wybrzeże Wyspiańskiego 27, Wrocław, Poland

Abstract The paper presents an attempt at the construction of a wide scale parser for English based on Inductive Learning and limited resources. The parser loosely preserves the *shift-reduce* scheme, enriched with powerful actions, and a compound *Decision Tree* instead of the decision table. The attempt originates directly from Hermjakob's ideas [3], but an important goal was to analyse possible extensions to a wide scale solution. Several supporting heuristics, as well as the overview of the development process and experiments, are described in the paper.

1 Introduction

The 'mission almost impossible' to build a deep parser of English from scratch in several months relying on a very limited resources was the origin of the work presented here. The goal of the project was the construction of a wide-scale commercial machine translation (MT) system, developed by *Techland* company, used in the two products: *Internet Translator* (1.0 and 2.0) and *English Translator* 2.0. The low budget assigned to the projects made it impossible to use most of the existing ready-to-use components and linguistic resources, including parsers and *Penn Tree Bank*. Because of the commercial character of the project, we could not use most existing scientific solutions, e.g., Charniak's parser [1]. When one completes the picture with the need for deep parsing as the base for the next transfer phase, texts of any kind as the input of the system, and a small team working on the project, then one will get the idea how 'almost impossible' the mission was.

Looking for a starting point, while excluding all methods demanding a large corpus, we have chosen Hermjakob's method of parser learning [3] based on a generalised form of *Decision Trees* [6,7]. The parser follows the general shift-reduce scheme, but it is deterministic, and uses the hierarchical structure of *decision trees* instead of a control table. The decision structure is built on the base of rich context information expressed by 205 features: morphological, syntactic, semantic and "background knowledge" (a subcategorisation dictionary and a *concept hierarchy*). The tests of the parser were performed on the sub-corpus of *Wall Street Journal Corpus* with vocabulary of only 3000 lexemes. However, only 256 selected sentences were used in incremental

learning, while achieving: 98.4% for "part of speech tagging" (built into the parser), 89.9% for "labeled precission", and 56.3% "of test sentences without any crossing brackets" [3]. The important difference between Hermjakob's approach and other approaches to automatic parser induction proposed in the literature is that other approaches are based on statistical methods which assume the existence of a large tree bank, e.g. [1]. Automatic grammar acquisition from a tree bank in the style of [2] provides only a very large grammar (even compacted [4]) and an efficient parser still needs to be constructed. Other non-statistical approaches produce often shallow parsers or based on *Dependency Grammar*, e.g. [5].

The goal of the work presented here, was to extend Hermjakob's approach to the parsing of free texts and to reduce the amount of semantic information used by the parser, but still to keep the number of learning examples very small. Being realistic, we accepted the unavoidable decrease in the quality of the parser, but we wanted to receive an acceptable, even if sometimes only marginally so, analysis of every expression delivered to the parser.

2 Parser Architecture

The parser loosely follows the general scheme of *shift-reduce* parser[1], but there are additional types of powerful actions. In comparison to [3], the set of actions was reduced to five main types with three 'standard' types: *shift*, *reduce*, and *done*. The restrictions on reduce are weak, e.g. it can be applied to almost any elements on the stack. The 'non–standard' *add into* action is similar to reduce, but inserts one node into the other. The *empty-cat* action (creating gaps) produces an 'empty copy' of some node, i.e. a co-indexed clone of it. The *empty-cat* encompasses the functionality of Hermjakob's three actions: *empty-cat*, *co-index* (co-indexing the nodes), and *mark* (marking nodes with some extra-syntactic information). The last two are not used because of limited semantic information. Also, the *reduce* was not used for recognising multi-word lexemes, as in [3]. Instead, this task is done in the preprocessing phase the MT system. The sophisticated formal language of addressing the arguments of actions [3] was mainly preserved, e.g.:

1. `R (-3 -1) TO VP AS MOD PRED AT -2` — reduces the elements on the stack to VP node assigning them the roles, the result goes to the pos. 2,
2. `A (-2 -1) TO (NP -1 BEFORE -2) AS CONJ COMPL` — adds the two elements into the first NP-tree below the position 2 on the stack.

The parser cannot backtrack but can send any top element back to the input list by a *shift-out* action. The wrong choice can create an infinite cycle of actions sending elements to and fro. The *sanity checker*, controlling

[1] Firstly, elements, representing morphologically analysed words, are inserted on the *input list*, next, by *shift* actions can be moved to *the stack*, where several elements can construct a compound element (a sub-tree of the derivation tree) by a *reduce action*. The actions are being chosen on the base of decision table.

the parser, breaks loops by forcing an extra *shift in* of the next element. In our approach, because of the decreased quality of decision structures (constructed for free texts) the responsibility of the sanity checker were extended to controlling of the *proper application* of the actions. A set of hand-coded rules defining pre-conditions of particular actions were introduced, e.g.

```
   [ACTION] REDUCE 2 TO VP AS MOD SAME
[COND] (SYNT OF -1 AT VERB= VP) AND (
(SYNT OF -2 AT SYNT-ELEM= PP) OR (SYNT OF -2 AT SYNT-ELEM= ADV)
OR (SYNT OF -2 AT SYNT-ELEM= SNT) )
```

Above, the `[COND]` part defines the state of the stack, for which the `[ACTION]` part can be performed. The `[COND]` part is defined in terms of *features* and their values. The features describe partially the state of the parser (the stack and the input list) achieved by performing the previous action. The features can express any kind of information stored in elements:

1. values of morphological attributes such as number, gender, verb form etc. (some of them are ambiguous in most parse nodes),
2. the structure of nodes, e.g. presence of some branch described by the syntactic (and semantic) role or values of attributes of some role filler,
3. possible agreement on values of attributes between some nodes,
4. possible *matching*: between subcategorisation pattern of one node (possible head) and the second element as the possible complement of the head (the syntactic role and/or the category of the possible filler is returned).

The features use the same formal language of addressing elements like actions, i.e. features can access almost any element on the stack or the input list. If an element pointed out by a feature does not exist or have an attribute under question, the feature returns `UNAVAIL`. The values of the features form a *is-a* hierarchy. A level of generality is specified for each feature on which its value is read/tested. In comparison to [3], the number of syntactic roles and categories was reduced, and the features based on semantic properties of elements (the semantic class of an element, or semantic matching between two elements) had to be eliminated to large extent. The creation of a detailed ontology in a style of [3] was impossible for the open domain. But, the semantic features form more than 50% of 205 ones used in [3], while being of the great importance for his parser. As a kind of surrogate, features giving access to *WordNet* (WN) classes of the lexemes, were introduced in this approach e.g.:

- `classp of wn-en-building of mod of pp -1` — testing presence of the WN class in the modifier of the first PP on the stack.

Only some WN classes concerning: building, location, measure, person, quantity, time, and way (path), appeared to be useful in learning, especially in making decisions concerning PPs attachment. The general obstacles for increasing the number of WN features were: problems with coping with too

large number of too specialised features, and the lack of good association between nominal and verbal hierarchies of WN.

The features of the type 4, are based on a subcategorisation dictionary (SCD), and take into account the present filling of a head. The subcategorisation pattern is being chosen on the base of the heuristically calculated ranking of all possible ones for the head. The size of SCD for unlimited domain must be relatively very large (here: 18 000 entries). Unfortunately, increasing number of ambiguities brings decreasing quality of matching. Entries in SCD are tree structures with distinguished *leaf*, marked with the `PRED` role, representing the *head*. Besides the structure, each tree describes several complements (*sub-trees*) and their *syntactic roles*, obligatory values of morphological attributes, and even particular lexemes. The English subcategorisation patterns were acquired from *XTAG* grammar (lexical dictionary) [8], and automatically transformed from the *XTAG* format to the simplified grammar implicitly defined by the decision structure of the parser.

Some examples from the final set of the features:

1. `synt of -3 at verb` — reads the syntactic category of the element on the stack and returns one of the immediate subcategories of the verb,
2. `np-vp-match of 1 with 2` — checks whether elements on positions 1 and 2 on the input list match syntactically as subject and verb predicate,
3. `syntrole of vp -1 of -2` — checks whether the element on the stack matches the first not filled argument of some subcategorisation pattern of the first VP element on the stack,
4. `morphp of f-ger of mod of -1` — tests whether the value of the attribute of the sub-tree with the role `mod` (i.e. a kind of adjunct) of the position 1 equals to `f-ger`.

3 Learning Process

As in [3], the parser learns directly from examples of parsing actions recorded during hand-made parsing of a sentence. Thus, the learning corpus (called here *a log*) consists of pairs: a sentence and sequences of parsing actions, defining operationally the syntactic structure, e.g. (a fragment of the log)

```
SENTENCE The bar code on the product was blurred.
  @ACT SHIFT DET
  @ACT REDUCE 1 TO DP AS PRED
  @ACT SHIFT NOUN
```

During learning the recorded actions are performed on an example sentence. Next, *examples*, i.e. pairs: an action and a vector of values of the features, are used in the construction of the *Decision Structure*. The examples are always regenerated during learning according to a specified vector of features. In later versions of the parser, words in the log were annotated by the tagger in order to be consistent in learning with the parser application.

The *Decision Structure* (DS) is built by *Inductive Learning* and it is: a *simple Decision Tree* (in the sense of [6]), a *tree* of DSs, or a *list* of DSs.

As in [3], each 'sub-tree' in a compound DS is marked with a predicate defining a *similarity class* of examples. All examples in the given class "are treated as if they were the same" [3], any other belongs to the `OTHER` class. The `OTHER` examples are passed further on to the next DS in a list. Precise decisions for the examples of the given class can be made (ascription of a particular action) by the given DS or by DSs of the lower levels, e.g. a simplified part of the DS for the `ADD` class of examples:

```
Tree: SYNT OF ACTIVE-FILLER -1 AT SYNT-ELEM
      UNAVAIL Leaf: Other
      NOUN Tree: SYNT OF PRED* OF NP -1 AT NOUN
            PRON Leaf: Decision
                  ADD ACTIVE-FILLER -1 BEFORE -1 INTO -1 AS MOD
            NP Leaf: Other
            PN Leaf: Other
      PP Leaf: Other
```

The construction of all decision trees in DSs is based on the C4.5 algorithm [7]. The sophisticated structure of DSs facilitates introduction of some hand-coded knowledge (the predicates are manually assigned) to the learning process. The exact structure can strongly influence results of learning. Following [3], decision on more exceptional cases are forced to be done first, before more typical ones. However, because of the limited information available for our parser, examples were divided only into several classes by a list of DSs, i.e. `DONE`, `ADD`, `EMPTY-CAT`, `SHIFT OUT`, `R 4`, `R 3 AS DUMMY PRED`, `R 3 AS COMP CONJ PRED`, `R 3 AS SAME`, `R 1`, `R 2 AS PRED OR SAME DUMMY`, `R 2 TO PP`, `R 2 TO SNT`, `REDUCE`, `SHIFT IN`. Moving the `DONE` class to the beginning was motivated by its relatively rare occurrence and the strange dependency of the parser on the presence of the final period. Another very difficult actions appeared to be the actions of the `EMPTY-CAT` class (introducing a trace).

4 Experiments

The goal of the experiments was to construct a deep parser producing a useful result for any sentence. Firstly a direct extension of Hemrjacob's approach to wide scale was developed. The coverage of the parser was extended by increasing the number of examples and introduction of many specialised features (totally 234) for exceptional cases. A set of 912 distinct sentences was parsed manually. Selection of the examples appeared to be difficult according to the open domain. The initial naive strategy was to choose some sentences randomly, and next to add gradually the following on the base of types of errors made by the parser learned on the current set. After noticing that questions, sentences with many auxiliaries etc. cause a lot of mistakes, a

lot of sentences taken from a English grammar textbooks were systematically added. Finally, many sentences and longer expressions from Web pages (some of them including constructions like citations, brackets etc.) were randomly selected and added. Because of the parser being too dependent on the final period or question mark, the final manually prepared version of the log was doubled automatically (up to 1783 examples) by adding the versions without the final delimiter. A generalisation of this method can be applied to generation of partial or modified expressions on the base of examples.

According to [3], the operational character of the learning makes creation of a detailed grammar unnecessary. However, as one could observe, the increasing number of examples (almost four times more than in [3]) was rapidly increasing the probability of inconsistency in the log (two different actions done in almost the same state). The large number of the features caused, that each inconsistency resulted in creation of a 'strange rule', in which any feature from the vector could be used to solve an inconsistency, e.g. the reduction of an object into VP element could be activated by an adverb on some remote position of the stack. Even the sophisticated construction of DS could not prevent it. The inconsistencies were mostly caused by typical actions, not exceptional ones. They appeared when one teacher performed the same action in a slightly different situations, or in a different situation than the other teacher. The resulting 'strange rules' could not be eliminated by the machine learning algorithm itself. Their creation was the natural consequence of inductive learning algorithm trying to find some reasons for performing two different actions in the same state. The 'strange rules' decrease the quality of the parser, unexpectedly spoiling analysis of simple expressions.

Trying to find a remedy, a set of guidelines for teachers was written, defining implicitly a kind of semi-formal grammar! Moreover, a special tool was constructed, which enables browsing the log using some query language, and discovering inconsistencies. Another cause of inconsistency were misleading expectations of a teacher concerning the exact features used by the parser in a particular decision. In order to bring the perspective of a teacher closer to the perspective of the parser, the teacher could see, optionally, only the basic window of $\pm 5/3$ elements during manual parsing.

In the second phase we tried to improve generalisation of the parser, by its integration with a tagger and significant reduction of the number of the features (down to 135). Introduction of the tagger separated from the parser (in [3] the tagger is implicitly encoded by the `SHIFT IN` actions) helped reducing the number of the features. Next, the basic window of elements permanently tested (morpho-syntactic attributes) was reduced from ± 5 to ± 3. Thirdly, iteratively, importance of the features from outside of the basic window was being heuristically assessed, and after each reduction, the learning was being restarted. This process resulted in finding many inconsistencies and removing many too specialised features. Moreover, the parser quality improved significantly, maintaining the quality of the log became easier and the time of

learning decreased. However, in the case of the reduced and stable number of the features, some sentences as being ambiguous, could not be added to the log.

The initial hope that the development of the parser would be faster without the need to create a grammar first, was fulfilled only partially. The preparation of one example sentence containing 30 words takes more than 0.5 hour by an experienced person. The preparation of the good log, free of inconsistencies, takes a lot of time, increasing with the number of the examples. The initial fast improvement in syntax covering of the parser, slows down quickly. Fortunately, the large number of features do not necessarily enforces a huge number of examples. The number of examples should be rather chosen according to grammar coverage criterion.

The quality of the parser is far from being satisfactory. Finally, only 152 from 325 sentences (textbook examples and sentences from the Web) were parsed without errors, according to an expert. Taking a look into the wrongly analysed examples one can notice several sources of errors:

- a partial analysis caused by several stops, in which subsequent parts are parsed as isolated phrases,
- words misinterpreted by the tagger (97% accuracy), specially noun↔verb,
- bad segmentation: two sentences merged into one or wrong point chosen by heuristic splitting of a compound sentence,
- and obviously wrong decisions made by the parser, e.g. bad attachment of PP, or a main verb separated from the auxiliary verb.

However, a lot of badly parsed sentences had large, well parsed fragments and the errors did not influence the overall result of the translation. Moreover, there were several types of errors, appearing systematically, which could be neutralised during the transfer phase, e.g. multiplied trace (identical nodes generated by a `EMPTY-CAT` action, which could be reduced to one), PP(*of the police*) treated as an adjunct of NP instead being analysed as a kind of DP, or the lack of the top SNT node.

Many errors originate from stops in parsing. The stops can be caused by the sanity checker, but most of them result from the deterministic character of the parser (always only one solution). But such an idealistic assumption was very hard to be maintained in open domain without rich semantic information. As the result, the parser trying to find the analysis of some ambiguous construction, makes wrong decision and stops later on an unknown combination of feature values. A mechanism of 'pushing forward' by a special *shift in* action was devised. After 'pushing', some parts of a sentence following the problematic construction can be analysed properly. Very often, the stack contains a set of partial trees after the main parsing has been finished. Instead of a simple merge of all trees, like in [3], a *repairing parser*, based on an expert systems was introduced. The rules define repairing procedures for correcting and finalising the parsing. Another simple improving technique was to divide longer sentences into several shorter ones by simple heuristic

rules. It is more likely, that a shorter sentence would be parsed without stops. Anyway, the parser is extremely fast (several sentences per second on PC), consuming much less time than other parts of the MT system!

5 Conclusions

Our attempt to extend Hermjakob's approach to a wide scale application has shown how much of its accuracy depends on the limited domain and hand-coded rich syntactic-semantic information. Even multiplying the number of example sentences by four and using several heuristic methods, we could not come close to Hermjakob's results. The problem is not an insufficient number of examples, but insufficient information delivered to the parser, while keeping it strictly deterministic. In many cases, adding a sentence to the log needed an additional feature. However, the experiments showed that in open domain it is better to limit the number of the features, because of inconsistencies.

An interesting extension would be an automatic acquisition of examples from a tree bank, e.g. *Penn Tree Bank*, based on choosing for learning sentences less similar to the sentences already added to a log. This mechanism could be based on a kind of 'pre-parsing' of unseen sentences.

Besides the strong limitations being met, a practically useful parser for English, utilising small resources (i.e. less than two person-years, a small operational tree-bank of 912 parsed sentences, and a 'standard' tagger) was constructed. Any kind of statistic information (except in the tagger) was not applied. In the case of languages for which a large tree-bank does not exist, e.g. Polish, such an attempt seem to be a possible way to construct a parser.

References

1. E. Charniak. A Maximum-Entropy-Inspired Parser. In *Proceedings of NAACL-2000*, 2000.
2. R. Gaizauskas. Investigations into The Grammar Underlying The Penn Treebank II. Research Memorandum CS-95-25, Deparmtent of Computer Science,University of Sheffield, 1995.
3. U. Hermjakob. *Learning Parse and Translation Decisions From Examples With Rich Context*. PhD thesis, University of Texas, Austin, 1997.
4. A. Krotov, M. Hepple, R. Gaizauskas, and Y.Wilks. Compacting The Penn Treebank Grammar. In *Proceedings of the COLING/ACL98 Joint Conference*, pages 699 – 703, Montreal, 1998.
5. J. Nivre. Inductive Dependency Parsing. MSI Report 04070, School of Mathematics and Systems Engineering, Växjö University, 2004.
6. J.R. Quinlan. Induction of Decision Trees. *Machine Learning*, 1(1):81–106, 1986.
7. J.R. Quinlan. *C4.5: Programms for Machine Learning*. Morgan Kaufmann, San Mateo, 1993.
8. A Lexicalized Tree Adjoining Grammar for English. Technical Report, The XTag Group of Institute for Research in Cognitive Science, University of Pennsylvania, 1999.

Baseline Experiments in the Extraction of Polish Valence Frames

Adam Przepiórkowski[1] and Jakub Fast[2]

[1] Institute of Computer Science, Polish Academy of Sciences, ul. Ordona 21, Warsaw, Poland
[2] Collegium Invisibile, ul. Krakowskie Przedmieście 3 pok. 12, Warsaw, Poland

Abstract. Initial experiments in learning valence (subcategorisation) frames of Polish verbs from a morphosyntactically annotated corpus are reported here. The learning algorithm consists of a linguistic module, responsible for very simple shallow parsing of the input text (nominal and prepositional phrase recognition) and for the identification of valence frame cues (hypotheses), and a statistical module which implements three well-known inferential statistics (likelihood ratio, t test, binomial miscue probability test). The results of the three statistics are evaluated and compared with a baseline approach of selecting frames on the basis of the relative frequencies of frame/verb co-occurrences. The results, while clearly reflecting the many deficiencies of the linguistic analysis and the inadequacy of the statistical measures employed here for a free word order language rich in ellipsis and morphosyntactic syncretisms, are nevertheless promising.

1 Introduction

The aim of this paper is to report the results of preliminary experiments in the extraction of verbal valence frames, i.e., lists of verbs' arguments, from a corpus annotated morphosyntactically at word level.

Valence dictionaries are crucial resources for the operation of syntactic parsers, and yet, for many languages such resources are unavailable or they are available in paper form only. In case of Polish, there is only one large valence dictionary, namely, [5], although valence information is present also in the general dictionary of Polish [1]. Both dictionaries are available only in the traditional paper form. Additionally, there are two much smaller electronic valence dictionaries, created by Marek Świdziński and Zygmunt Vetulani, of about 1500 and 150 verbs, respectively.

The lack of large machine-readable valence dictionaries for Polish notwithstanding, there are many well-known arguments for constructing such dictionaries automatically, on the basis of naturally occurring texts. First of all, automatic methods of constructing valence dictionaries are much quicker and cheaper than the traditional manual process (e.g., the five volumes of [5] were published in the space of twelve years). Second, automatic methods are more objective than the traditional methods, based on potentially inconsistent intuitions of a team of lexicographers. Third, automatic methods may provide

not only the categorical information, but also statistical information about how often a verb occurs with a given frame, which is particularly useful for probabilistic parsers. Fourth, the same methodology may be applied, without any overheads, to different collections of texts, e.g., to create thematic or diachronic valence dictionaries. Moreover, automatic methods may be and have been used for extending and verifying existing valence dictionaries.

The textual material for the experiments reported in this paper was the IPI PAN Corpus of Polish [7], the first and currently the only large publicly available morphosyntactically annotated corpus of Polish (cf. `www.korpus.pl`). Since the corpus is rather large (over 300 million segments), two smaller corpora were used in the experiments: the 15-million segment (over 12 million orthographic words; punctuation marks and, in some special cases, clitic-like elements are treated as separate segments) `sample` corpus, as well as a less balanced 70-million segment `wstepny` subcorpus. The corpus does not contain any constituent annotation apart from sentence boundary markers, but it employs a detailed positional tagset providing information about parts of speech, as well as values of inflectional and morphosyntactic categories (cf. [8]). Morphosyntactic analyses are disambiguated by a statistical tagger with a rather high 9.4% error rate [3], but all the original tags provided by the morphosyntactic analyser are also retained in the corpus.

As in the case of similar experiments reported for English, German and other languages since [2] and [4], the current algorithm consists of two stages. First, a linguistic module identifies cues, i.e., observations of co-occurrences of verbs and apparent valence frames. This stage produces much noise, partly due to the low quality of the tagger, and partly because of various deficiencies of the shallow syntactic parser used for identifying nominal phrases (NPs) and prepositional phrases (PPs). Second, the statistical module applies inferential statistics to the output of the linguistic module, trying to determine which of the valence frames observed with a given verb can be, with a certain degree of confidence, classified as actual valence frames of this verb.

The rest of the paper is structured as follows. Section 2 describes the linguistic module of the learning algorithm, i.e., shallow syntactic processing and cue identification, while the next section, 3, briefly introduces the statistics employed here. The following section, 4, describes the experiments, presents their results and evaluates them. Finally, §5 concludes the article.

2 Syntactic Cues

The process of collecting valence cues consists of three steps. First, a simple shallow grammar is applied to the XML corpus sources, resulting in the identification of some NPs, PPs and verbs. Second, each sentence is split into clauses. Third, for each clause, all NPs and PPs identified in this clause are collected into an observed frame (OF), and the pair ⟨verb, OF⟩ is added to the set of observations. This section presents the details of the first two steps.

The shallow grammar employed in these experiments is particularly simple. Conceptually, it is a cascade of regular grammars with some added unification-like functionality for handling NP- and PP-internal agreement. There are only 9 rules for NPs and 4 rules for PPs. 5 of the NP rules are responsible for identifying very simple NPs containing a noun and possibly a sequence of pre- or post-modifying adjectives, and another 4 rules take care of personal pronouns and the strong reflexive pronoun SIEBIE. Moreover, 5 rules identify verbs, determining whether they are reflexive or negated; since in Polish valence to some extent depends on polarity, it makes sense to treat affirmative and negative verbal forms separately. Adjectival participles and gerundial forms are not considered to be verbal by this grammar.

The range of phrases that this grammar identifies is very limited: numeral phrases, adjectival phrases, adverbial phrases, clauses and infinitival verbal phrases are excluded from the consideration here, i.e., the task at hand is constrained to the identification of possible NP and PP arguments. Moreover, many NPs and PPs remain unidentified due to the fact that a very conservative approach to corpus annotation was adopted: only those forms are taken to be nominal, verbal, etc., whose all morphosyntactic interpretations are, respectively, nominal, verbal, etc. That is, the decisions made by the tagger are initially ignored. In the process of identifying NPs and PPs, all morphosyntactic interpretations of all forms belonging to an NP are unified; for example, if an adjective is syncretic between neuter singular and feminine plural, and a noun is syncretic between neuter singular and neuter plural, only the neuter singular interpretation is selected for the resulting NP. The outcome of the tagger is taken into account only in cases when this procedure results in a number of possible interpretations.

Also the second step of cue identification, i.e., finding minimal clauses, is particularly simple. Sentences are split into potential clauses on commas and conjunctions (disregarding those which have already been classified as belonging to an NP or a PP), and those potential clauses which contain a verb are selected as actual clauses. This, again, results in some noise.

Two versions of the grammar were used in the experiments, with the main difference between them being the treatment of genitive NPs (GNPs). As is well known, in Polish, GNPs often occur as modifiers of other NPs, which often results in attachment ambiguity. The shallow grammar employed here cannot resolve such ambiguities. If such GNPs were left in the output of the grammar, this would result in many false observed valencies involving GNPs. Both grammars currently deal with this problem via brute force: the first grammar (G1) ignores GNPs altogether (so there is no way to identify valence frames with genuine GNP arguments), while the second grammar (G2) attaches GNPs to immediately proceeding NPs and PPs, whenever possible.

The final simplification assumed here is that nominative NPs are ignored altogether, i.e., no attempt at distinguishing subject-taking verbs and subjectless verbs is made.

3 Basic Statistics

Three standard statistics were compared for rating each observed verb/frame combination in terms of how strong a piece of evidence it is for inferring the verb's valence requirements. The statistics were: binomial miscue probability testing, the t test for non-normal variables, and Likelihood Ratio.

The three metrics employ a common probabilistic model. The values characterising a given verb/frame combination $\langle v, f \rangle$ are interpreted as describing two binomial samples, m_X and m_Y, such that in m_X, the number of successes k_X is equal to the number of f's occurrences with v and the sample size n_X is equal to the total number of v's occurrences, and in m_Y, the number of succeseses k_Y is the number of occurences of f in clauses that *do not* contain v, and sample size n_Y is the total number of such clauses. The samples are interpreted as taken from two binomially-distributed random variables $X \sim \mathrm{Bin}(n_X, \pi_X)$ and $Y \sim \mathrm{Bin}(n_Y, \pi_Y)$, where π_X and π_Y are estimated on the basis of m_X and m_Y as $\hat{\pi}_X = \frac{k_X}{n_X}$ and $\hat{\pi}_Y = \frac{k_Y}{n_Y}$.

3.1 Binomial Miscue Probability Test

Assuming a certain maximal probability B_f that — due to parser or grammatical error — a given frame f occurs in a sentence irrespective of the verb it contains (in the experiments, B_f was set to 2^{-7}), the binomial test measures the probability of k_X or more occurences of f being generated in a sample of size n_X by a random variable with a distribution $\mathrm{Bin}(n_X, B_s)$. Its value is calculated as follows:

$$H(B_f) = \sum_{i=k_X}^{n_X} (B_f)^i \, (1 - B_f)^{n_X - i} \begin{pmatrix} n_X \\ i \end{pmatrix} \tag{1}$$

If this probability is sufficiently low (i.e., lower than an assumed null hypothesis rejection level), the probability that the observed number of co-occurrences is due solely to error can be neglected and the frame classified as valid for the verb.

3.2 *t* Test

The t test establishes the significance of a difference observed between the probabilities of success in two independent samples. For binomially-distributed variables, its value is given by the following equation:

$$t = \frac{\hat{\pi}_X - \hat{\pi}_Y}{\sqrt{\frac{\hat{\sigma}_X^2}{n_X} + \frac{\hat{\sigma}_Y^2}{n_Y}}} \tag{2}$$

where $\hat{\sigma}^2 = \hat{\pi}(1 - \hat{\pi})$ and is the variance in a single Bernoulli trial estimated on the basis of the respective samples. The null hypothesis for this test is that the

two samples come from the same population, and the alternative hypothesis is that m_X comes from a population characterised by a significantly lower probability of success. If the value of t is lower than an appropriate threshold value, it can be inferred that the probability of succes in m_X is significantly lower than that in m_Y, the null hypothesis is rejected and f is taken to be an invalid frame for v.

3.3 Likelihood Ratio

The Likelihood Ratio test measures the significance of the difference in the probability of a particular outcome given two different, nested classes of probabilistic models. Assuming a joint distribution $\langle X, Y\rangle$, the likelihood ratio is used to compare the quality of the best model in which X and Y have the same probabilities of success against the quality of the best joint binomial model that does not make this assumption. The value of the likelihood ratio is thus the following:

$$\lambda = \frac{\max_p P(\langle m_X, m_Y\rangle | \langle X, Y\rangle \sim \mathrm{Bin}(n_X, p) \times \mathrm{Bin}(n_Y, p))}{\max_{p_X, p_Y} P\left(\langle m_X, m_Y\rangle | \langle X, Y\rangle \sim \mathrm{Bin}(n_X, p_X) \times \mathrm{Bin}(n_Y, p_Y)\right)} \tag{3}$$

A sufficiently low value of λ implies that the theoretical values of π_X and π_Y are sufficiently different. In the experiments described below, λ was multiplied by -1 if the difference $p_X - p_Y$ was negative, in order to eliminate only such $\langle v, f\rangle$ combinations where f is significantly less likely to occur with v than with any other verb, and not the other way round. If the value of λ is negative and sufficiently close to 0, it can be concluded that f is an invalid frame for v.

4 Experiments and Evaluation

Altogether 16 experiments were performed: for the two subcorpora mentioned in §1, `sample` and `wstepny`, two versions of the grammar described in §2, G1 and G2, were used, and for each of the corpus/grammar combinations, four statistics were applied: the three statistics described in §3, as well as the baseline statistic, i.e., the Maximum Likelihood Estimate (MLE) of π_X, where frames with $\hat{\pi}_X \geq 0.01$ were selected as valid. The result of each of the experiments is a function from verb lemmata to sets of valence frames accepted for those verbs by a given statistic, on the basis of cues generated by a given grammar running on a given corpus. Frames are understood as multisets of arguments.

For example, the following 8 frames were selected for the verb DOSTRZEC (*discern*) by the binomial miscue probability statistic, on the basis of data generated by G2 from the `wstepny` corpus: `<empty>` (empty valence frame), `np/acc` (an accusative NP), `pp/w/loc` (a PP consisting of the preposition W and a locative NP), `np/acc+np/gen`, `np/acc+pp/w/loc`, `np/acc+pp/w/loc`, `np/gen`, `np/gen+pp/w/loc`. Valence dictionaries used for the evaluation (see

below) mention only one valence frame, `np/acc`, but — with the possible exception of `np/acc+np/gen` — other discovered valencies are reasonable sets of dependents of DOSTRZEC, with `pp/w/loc` being a common locative modifier of DOSTRZEC and `np/gen` being a possible effect of the (long-distance) genitive of negation or genitive modifiers, and with the observations `<empty>` and `pp/w/loc` reflecting the ellipsis of the object. 38 other observed frames for DOSTRZEC were rejected by this test.

For the purpose of the experiments reported here, the confidence level for all statistics was set to 99%. Moreover, only frames registered a certain minimal number of times were taken into account, and additionally a certain minimum verb/frame co-occurrence restriction was imposed. Four such ⟨frame occurrences, frame/verb co-occurrences⟩ cutoff points were tested, from $\langle 50, 3\rangle$ to $\langle 800, 10\rangle$.

Two 'gold standards' were adopted for the purpose of evaluating the results of these experiments, namely, the two dictionaries containing valence information mentioned in §1: [5] and [1]. From the set of verbs for which at least 100 observations were registered by the G1 grammar in the `sample` corpus, 48 verbs[1] were blindly selected, evenly distributed across the scale of the number of occurrences. For each of these verbs, valence information was manually extracted from the two valence dictionaries. It should be noted that in both cases, but perhaps especially in the case of [5], this was an interpretative process, due to the fact that both dictionaries refer to some of the arguments via their function (e.g., a temporal phrase) and not their morphosyntactic form. Those valence frames were narrowed down to valence frames containing only phrases identifiable by the grammars, i.e., NPs and PPs.

Bańko [1]

	`sample`		`wstepny`	
	G1	G2	G1	G2
MLE	40.17	42.23	38.33	36.94
binomial	**40.96**	42.80	**38.94**	**38.01**
t test	39.83	43.90	36.00	35.24
likelihood	39.93	**44.23**	35.69	35.06

Polański [5]

	`sample`		`wstepny`	
	G1	G2	G1	G2
MLE	40.07	42.32	37.35	36.47
binomial	**41.13**	43.28	**38.42**	**37.86**
t test	39.08	42.92	33.23	32.54
likelihood	39.11	**43.41**	33.48	32.65

Table 1. F values for the 16 tests, for two gold standards (based on [1] and [5]), for the cutoff point $\langle 400, 10\rangle$.

[1] BIĆ SIĘ, DOBIEC, DOCZEKAĆ SIĘ, DOSTRZEC, DŹWIGAĆ, NABIERAĆ, NAUCZYĆ SIĘ, OBSŁUGIWAĆ, ODBIJAĆ SIĘ, ODCZYTYWAĆ, OKAZAĆ SIĘ, OPOWIADAĆ, ORZEKAĆ, PŁAKAĆ, POPATRZEĆ, POSTĘPOWAĆ, POTRZEBOWAĆ, POWIĘKSZYĆ, PRZESTRZEGAĆ, PRZEŻYWAĆ, REALIZOWAĆ, ROZEGRAĆ, SIĄŚĆ, SPACEROWAĆ, STARTOWAĆ, SZANOWAĆ, UCZESTNICZYĆ, UDAWAĆ SIĘ, URZĄDZAĆ, USTANOWIĆ, WKRACZAĆ, WYBIERAĆ SIĘ, WYNIKAĆ, WYSTAWIAĆ, WYTWARZAĆ, WZDYCHAĆ, ZABRZMIEĆ, ZAJMOWAĆ, ZAJRZEĆ, ZAPOWIADAĆ, ZATRZYMYWAĆ SIĘ, ZAWIERAĆ, ZAŻĄDAĆ, ZDARZAĆ SIĘ, ZDAWAĆ SIĘ, ZJECHAĆ, ZRZUCIĆ, ZWRACAĆ.

Table 1 presents the summary of the results for the two gold standards. The numbers in the table are the average values of F-measure (the harmonic mean of precision and recall) for the 48 verbs, where only those frames were taken into account which were observed at least 400 times, and co-occurred with a given verb at least 10 times. The best statistic for each corpus/grammar pair is indicated by bold face.

As can be seen in Table 1, the binomial miscue probability test is the best statistic in six of the eight corpus/grammar/gold standard combinations. As can be seen from the comparison of Tables 1 and 2, this test also turns out to be the most stable, i.e., least dependent on cutoff points. Another interesting observation is that, surprisingly, the results for the much smaller `sample` corpus are uniformly better than the results for `wstepny`. The decrease in F-measures is almost entirely due to reduced precision; for example, the 41.89/35.99 difference in Table 2 for the binomial test applied to the results of G2 as evaluated on the basis of [1] corresponds to recall values of 74.07/73.72 and precision values of 34.31/27.37. Why this should be so is still not fully clear. Finally, it should be noted that the baseline MLE test mentioned above almost always fares better than the more sophisticated t test and likelihood ratio.

Bańko [1]

	`sample`		`wstepny`	
	G1	G2	G1	G2
MLE	34.77	35.47	31.16	29.87
binomial	**41.79**	**41.89**	**38.14**	**35.99**
t test	30.28	32.32	22.31	18.62
likelihood	30.28	32.69	22.01	18.57

Polański [5]

	`sample`		`wstepny`	
	G1	G2	G1	G2
MLE	36.02	36.37	32.55	31.39
binomial	**40.21**	**41.67**	**37.52**	**37.83**
t test	30.50	32.15	23.06	20.65
likelihood	30.92	32.51	23.00	20.51

Table 2. As Table 1, but for the cutoff point $\langle 50, 3 \rangle$.

The numbers in Table 1 may appear to be disappointing, especially as compared with F-measures given in the literature, e.g., F-measures of around 80 reported in [9]. However, these numbers are not comparable, as the results usually reported in the literature are F-measures based on so-called token precision and recall, i.e., the evaluation is based on the manual annotation of frames in a test corpus, usually performed by the authors. In the current paper, on the other hand, automatically extracted frames are evaluated on the basis of two valence dictionaries given *a priori*. Moreover, the two dictionaries differ significantly in the kind of valence information they contain. In fact, when the information in one dictionary is evaluated against the other, the F-measure is around 65.5, so — in a sense — this is the upper limit of what can be achieved using the methodology adopted here.[2]

[2] For [5] treated as extracted data and [1] as gold standard, the average precision, recall and F for the 48 verbs (narrowed down to frames consisting of NPs and PPs

5 Conclusion

To the best of our knowledge, the experiments reported here are the first attempt at the extraction of valence frames for a free word order Slavic language from a corpus containing no syntactic constituency annotation, rather than from a treebank, cf., e.g., [9].

The manual inspection of the results suggests that many 'errors' stem from the phenomenon of ellipsis, from erroneous classification of adjuncts as arguments, and from various errors and inconsistencies in gold standards. Taking into consideration the fact that the cross-gold standard agreement, as given by the F-measure, is around 65.5, and given the high noise production at each level of linguistic processing (morphological analysis, tagging, shallow parsing) and the basic nature of statistical models involved, the results reported here are highly encouraging.

References

1. Mirosław Bańko, editor. *Inny słownik języka polskiego*. Wydawnictwo Naukowe PWN, Warsaw, 2000.
2. Michael R. Brent. From grammar to lexicon: Unsupervised learning of lexical syntax. *Computational Linguistics*, 19(2):243–262, 1993.
3. Łukasz Dębowski. Trigram morphosyntactic tagger for Polish. In Mieczysław A. Kłopotek, Sławomir T. Wierzchoń, and Krzysztof Trojanowski, editors, *Intelligent Information Processing and Web Mining*, Advances in Soft Computing, pages 409–413. Springer-Verlag, Berlin, 2004.
4. Christopher D. Manning. Automatic acquisition of a large subcategorization dictionary from corpora. In *Proceedings of the 31st ACL*, pages 235–242, 1993.
5. Kazimierz Polański, editor. *Słownik syntaktyczno-generatywny czasowników polskich*. Zakład Narodowy im. Ossolińskich / Instytut Języka Polskiego PAN, Wrocław / Kraków, 1980–1992.
6. Adam Przepiórkowski. On the computational usability of valence dictionaries for Polish. IPI PAN Research Report 971, Institute of Computer Science, Polish Academy of Sciences, 2003.
7. Adam Przepiórkowski. *The IPI PAN Corpus: Preliminary version*. Institute of Computer Science, Polish Academy of Sciences, Warsaw, 2004.
8. Adam Przepiórkowski and Marcin Woliński. The unbearable lightness of tagging: A case study in morphosyntactic tagging of Polish. In *Proceedings of the* 4th International Workshop on Linguistically Interpreted Corpora (LINC-03)*, EACL 2003*, pages 109–116, 2003.
9. Anoop Sarkar and Daniel Zeman. Automatic extraction of subcategorization frames for Czech. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING 2000)*, pages 691–697, 2000.

only) are 74.77, 65.78 and 64.54, respectively, and for the opposite evaluation — 65.65, 76.59 and 64.36.

Part VIII

Invited Session: New Trends in Data Mining and Knowledge Discovery in Uncertain, Nonstationary Spatio-Temporal Data

Gene Expression Clustering: Dealing with the Missing Values

Alicja Grużdź, Aleksandra Ihnatowicz, and Dominik Ślęzak

Department of Computer Science, University of Regina
Regina, SK, S4S 0A2 Canada
Polish-Japanese Institute of Information Technology
Koszykowa 86, 02-008, Warsaw, Poland

Abstract. We propose a new method to deal with missing values in the gene expression data. It is applied to improve the quality of clustering genes with respect to their functionality. Calculations are run against real-life data, within the framework of self-organizing maps. The applied gene distances correspond to the rank-based Spearman correlation and entropy-based information measure.

Keywords: Gene Expression Data, Self-Organizing Maps, Rank-Based Distances

1 Introduction

The DNA microarray technology provides enormous quantities of biological information about genetically conditioned susceptibility to diseases [2]. The data sets acquired from microarrays refer to genes via their expression levels. Comparison of the gene expressions in various conditions and for various organisms can be very helpful while formulating biomedical hypotheses.

Thousands of gene expressions have to be interpreted properly to retrieve valuable information. Dealing with huge amount of data and understanding the functional associations between genes is a major challenge. The appropriate choice of the data-based similarity/distance functions is crucial with that respect, particularly for applications of the gene clustering algorithms. By identifying the gene clusters, we can hypothesize that the genes grouped together tend to be functionally related. Thus, the gene expression clustering may be useful in identifying mechanisms of gene regulation and interaction, which can help in understanding the function of a cell. There are many algorithms for this purpose, such as hierarchical clustering [3], Bayesian clustering [10], fuzzy c-means clustering [4], or self-organizing maps (SOM) [18].

While grouping genes together, it is important to provide visualization enabling the experts to interact with the algorithmic framework. In [6] we focused on the SOM-based method mapping the similarities onto the two-dimensional grid. The implemented *Gene-Organizing Map* system (*GenOM*) modeled relationships between genes using various measures, including the statistical correlations [5] and the information functions [7]. Calculations indicated a particularly promising performance of the rank-based Spearman correlation and the *rough entropy* information distance [16].

A serious problem with the gene clustering corresponds to the missing values occurring in microarray data. This is mostly due to complicated, microscale process of their manufacturing. The experiments are very sensitive to spotting and hybridization conditions like temperature, relative humidity, image corruption, dust or scratches on slides, resolution and mechanical preciseness. There are many estimation methods which deal with this issue. A naïve approach is to fill the empty spaces with zero or the row average. More advanced methods refer to the k-nearest neighbor technique [3], singular value decomposition [19], or Bayesian principal component analysis [11].

We deal with the above-mentioned problem in a different, statistically less invasive way. We do not propose any method for filling the gaps in data because, for real-life data, the available expression levels are not always reliable enough to estimate the missing ones. Instead, we measure the distances between the incomplete gene expression vectors. We develop and compare the following exemplary approaches, extending the measures reported in [6]:

1. The Spearman correlation corresponds to the Euclidean distance between the vectors of ranks of original values, ordered with regards to particular genes. We calculate the *expected ranks*, as if the missing values were known. Then we use distances between the vectors of such expected ranks.
2. Rough entropy averages the entropies of binary variables, obtained using *discretization* induced by every particular record of observations. Given a cutting point, corresponding to some observation, we evaluate the probability that a missing value would be below/above it. Then we estimate the required probabilistic distributions for the (pairs of) genes-variables.

We test both above approaches against the data set related to a selected type of soft tissue tumor [1,14][1], further referred to as to the *Synovial Sarcoma* data. We provide simple examples for the pairs of functionally related genes. We run massive clustering calculations using the GenOM system as well.

The presented method easily catches the similar gene expression profiles in the considered data. It may lead to determination of the groups of genes with congruent activity responses to the specific tumor type in future. We can check if some genetic assumptions tend to be true. Also, as reported in [6], the GenOM interface enables us to presuppose the known gene correlation laws to achieve better clustering. It may be interesting for genetics and biologists as it gives opportunity to interact with the learning process.

The paper is organized as follows: Section 2 presents basics of the DNA microarrays; Section 3 – functions reflecting the gene expression similarities; Section 4 reconsiders the microarray data in terms of the expression ranks. Sections 5, 6 provide foundations for handling the gene distances given the missing values. Section 7 contains the results of calculations for the *Synovial Sarcoma* data. Section 8 summarizes our work.

[1] Downloaded from http://genome-www.stanford.edu/sarcoma/. We would like to thank the data suppliers for opportunity to run calculations using this data set.

2 The DNA microarrays

The DNA microarray technology [2] enables simultaneous analysis of characteristics of thousands of genes in the biological samples. It is automated, quicker, and less complicated than the previous methods of molecular biology, allowing scientists to study no more than a few genes at a time.

Microarrays have different formats, but all of them rely on DNA sequences fabricated on glass slides, silicon chips or nylon membranes. Each slide (DNA chip) contains samples of many genes at fixed locations. Each spot on the slide corresponds to a single gene. The microarray production starts with preparing two samples of mRNA – the sample of actual interest and a healthy control one. They are marked with fluorescent labels and repeatedly mixed for each of genes on the slide. The obtained color intensities of every spot indicate to what extent particular genes are expressed in the sample.

We obtain the gene expression data system $\mathbb{A} = (U, A)$, where U is the set of experiments and A – the set of genes.[2] Every gene $a \in A$ corresponds to the function $a : U \to \mathbb{R}$ labeling experiments with their expression levels. The data system is often transposed as displayed in Fig. 1, where genes correspond to the rows and experiments – to the columns.

A	u_0	u_1	u_2	u_3	u_4	u_5	u_6	u_7	u_8
a_0	2.174	-0.4405	1.363	0.6984	0.5569	*	-0.0658	0.6932	0.138
a_1	-0.2319	0.4421	1.317	2.911	0.5031	-0.2014	2.389	-0.339	0.2009
a_2	0.4963	0.084	0.9453	2.385	0.5654	-0.2638	2.743	-0.369	0.1344
a_3	-1.453	0.6558	-1.585	-1.009	-0.4974	-0.4548	-1.787	-0.4217	-0.9868
a_4	-2.352	-1.068	*	-2.342	-2.488	-2.874	-3.53	-0.9623	-1.241
a_5	-0.2695	-1.554	-0.3601	-1.199	-1.776	-2.678	-0.0756	*	-1.163
a_6	*	-0.3473	*	0.3414	0.1508	-0.9562	-1.794	0.8462	1.172
a_7	1.064	1.305	0.8207	1.323	0.9923	2.205	-0.3863	0.8436	-0.7809
a_8	-1.991	2.363	-0.2841	-1.32	-0.6655	0.1326	-1.219	-0.684	-1.201
a_9	-1.711	2.133	*	-0.9629	-0.8462	0.0897	-1.388	-0.6773	-1.103
a_{10}	-0.0803	1.036	0.7597	0.3875	-1.319	-1.582	0.8526	-1.215	-0.5528
a_{11}	0.7802	-0.442	-0.8912	0.193	0.6379	0.4672	-0.6572	0.0321	0.2658
a_{12}	-0.0594	-0.3767	-0.4723	0.0975	1.355	1.196	-0.3392	0.6019	0.4246
a_{13}	-0.702	-0.6721	-0.3448	-0.6236	0.6979	0.8698	-1.081	0.0611	-0.0976
a_{13}	0.895	0.6363	0.6441	1.04	1.387	0.113	1.017	1.241	1.538
a_{15}	-0.1608	0.4728	1.278	2.392	0.9721	0.5273	1.831	2.4	0.1117
a_{16}	-0.1888	0.6483	1.079	2.097	1.729	1.543	1.118	-0.3707	-0.0714
a_{17}	2.517	0.5933	0.8259	*	3.459	3.723	3.394	-0.4586	-0.8334
a_{18}	0.6119	0.0602	*	1.814	1.721	2.099	0.5186	-2.028	-0.5893
a_{19}	-0.0498	-0.6261	-1.352	-1.013	-1.729	-2.016	-1.911	-0.3923	-0.5924

Fig. 1. *Synovial Sarcoma* data: 9 experiments and 20 (out of 5520) genes.

[2] The notation $\mathbb{A} = (U, A)$, where U is the universe of objects and A is the set of attributes, is taken from the theory of rough sets and information systems [12].

3 Gene expression similarities

The process of retrieving meaningful cluster-based information demands the appropriate distance metrics. This non-trivial problem becomes even more complicated in the case of a model so unexplored and complex as genes. Defining biologically natural and understandable similarities is very challenging, mostly from the fact that we still know little about genes' functions.

At the current level of genetic knowledge we can just compare the analytical results with the experts' directions and commonly known medical facts. In [6] we tested the Euclidean distance, the Pearson and Spearman (rank order) correlations [5], as well as the *rough entropy distance* $\varrho_H : A \times A \to [0, 1]$, referring to the well known information-theoretic measures [7].

In its simplest form, the value of $\varrho_H(a, b)$ equals to $1/2 \cdot (H(a|b) + H(b|a)) = H(a, b) - H(a)/2 - H(b)/2$, where $H(a)$, $H(b)$ are entropies of genes $a, b \in A$, and $H(a, b)$ is the entropy of the product variable (a, b). Using the standard entropy formula for qualitative data, we obtain that $\varrho_H(a, b) \geq 0$, where equality holds if and only if a and b determine each other.

Normally, $\varrho_H(a, b)$ would require *discretization* of quantitative expression data or proceeding with parameterized probabilistic density estimates. Both approaches, however, seem to be too sensitive to the parameter changes, especially for relatively unreliable data such as those obtained from microarrays. Therefore, we consider *rough entropy* [16]. Given $u \in U$, we discretize the data with respect to its values $a(u)$, $a \in A$. Every gene $a : U \to \mathbb{R}$ is transformed to the binary attribute $a_u : U \to \mathbb{R}$ defined by formula

$$a_u(e) = \begin{cases} 1 \text{ if and only if } a(e) \geq a(u) \\ 0 \text{ if and only if } a(e) < a(u) \end{cases} \tag{1}$$

Given $B \subseteq A$ and $u \in U$, we denote by B_u the set of u-discretized genes taken from B. Entropy of B_u, denoted by $H(B_u)$, is calculated from the product probabilities of binary variables $a_u \in B_u$. Entropies in $\varrho_H(a, b)$ are calculated using the following formula for rough entropy (for $B = \{a\}$, $\{b\}$, and $\{a, b\}$):

$$H(B) = \frac{1}{N} \sum_{u \in U} H(B_u) \tag{2}$$

where N is the number of observations in U. $\varrho_H(a, b)$ equals zero, if and only if a and b *roughly determine* each other, i.e. the a's and b's value ranks are directly or inversely proportional. The way of defining the rough entropy assures that $\varrho_H(a, b) \leq 1$, where equality holds if and only if for every $u \in U$ the binary variables a_u and b_u are statistically independent. We have also $\varrho_H(a, b) \leq \varrho_H(a, c) + \varrho_H(b, c)$, where equality holds if and only if for every $u \in U$ the variables a_u and b_u determine together c_u but, on the other hand, remain statistically independent given c_u.

4 Ranking operations

Rough entropy keeps information-theoretic properties while not focusing too much on the precise expression values. Actually, it would not change when calculated for the ranking values instead of the original ones, if we assume that there are no equal expressions for every given gene in data (which happens very likely – see e.g. Fig. 1). The formulas for rough entropy are based entirely on inequalities between the attribute values, which can be equivalently rewritten for their ranks.

The same can be observed about the Spearman correlation, which is proportional to the normalized Euclidean-like distance

$$\varrho_S(a,b) = \sqrt{\frac{\sum_{u\in U}\left(\sigma(a)(u) - \sigma(b)(u)\right)^2}{N(N-1)}} \tag{3}$$

between permutations $\sigma(a), \sigma(b) : U \to \{0,\ldots,N-1\}$, $a, b \in A$. For the sake of simplicity, we will refer to (3) as to the *Spearman distance*. Let us that $\varrho_S(a,b) \in [0,1]$, where $\varrho_S(a,b) = 0$, if and only if $\sigma(a)$ and $\sigma(b)$ are identical, and $\varrho_S(a,b) = 1$, if and only if they are fully reversed to each other.

The analysis of the gene expression data through the rank-related distances turns out to be equally or even more efficient than in case of the actual values. It confirms intuition that we should focus on catching the global gene transcription profile changes rather than on local differences in expressions. Global changes can be captured by statistical correlations, although, in case of operating with the exact values, they may be inaccurate because of imprecise measurements and diversified data origin [1,14]. The rank-related correlations/distances assume less information and, therefore, may be more reliable in approximating the gene relationships.

In [6] we report that clustering based on the Spearman correlation (distance) is comparable to the Pearson correlation and much better than in case of the standard Euclidean distance. An additional advantage of the rough entropy distance is that it does not differ between the direct and inverse rank correlations. It may turn out to be very important while searching for the genes with significant functional relationships.

Throughout the rest of the paper we assume that the gene expressions are provided in a rank-based form, obtained by the original data preprocessing. We will not distinguish between the gene expressions $a : U \to \mathbb{R}$ and their corresponding rankings $\sigma(a) : U \to \{0,\ldots,N-1\}$. We will continue studying the Spearman and rough entropy distances for the rank-based attributes.

5 Handling the missing values

As mentioned in Section 1, the gene expression data may contain a substantial number of missing values, which is related to the microarray manufacturing

procedures. We propose a new method for measuring distances between the rank-based attributes, for such partially missing data. As mentioned in Section 1, it does not need estimation of the missing values. Instead, it predicts their (relative) ranking positions in a very simple way, with no advanced statistical models required.

Let us denote by $M(a)$ the number of missing values for $a \in A$. As shown in Fig. 2, the non-missing values are ranked from 0 to $N - M(a) - 1$. We treat a rank-based attribute as a function $a : U \to \{0, \ldots, N - M(a) - 1\} \cup \{*\}$, where every value different than $*$ occurs exactly once for the whole U.

A	u_0	u_1	u_2	u_3	u_4	u_5	u_6	u_7	u_8
a_0	7	0	6	5	3	*	1	4	2
a_1	1	4	6	8	5	2	7	0	3
a_2	4	2	6	7	5	1	8	0	3
a_3	2	8	1	3	5	6	0	7	4
a_4	3	6	*	4	2	1	0	7	5
a_5	6	2	5	3	1	0	7	*	4
a_6	*	2	*	4	3	1	0	5	6
a_7	5	6	2	7	4	8	1	3	0
a_8	0	8	6	1	5	7	2	4	3
a_9	0	7	*	3	4	6	1	5	2

A	u_0	u_1	u_2	u_3	u_4	u_5	u_6	u_7	u_8
a_{10}	4	8	6	5	1	0	7	2	3
a_{11}	8	2	0	4	7	6	1	3	5
a_{12}	3	1	0	4	8	7	2	6	5
a_{13}	1	2	4	3	7	8	0	6	5
a_{14}	3	1	2	5	7	0	4	6	8
a_{15}	0	2	5	7	4	3	6	8	1
a_{16}	1	3	4	8	7	6	5	0	2
a_{17}	4	2	3	*	6	7	5	1	0
a_{18}	4	2	*	6	5	7	3	0	1
a_{19}	8	5	3	4	2	0	1	7	6

Fig. 2. The rank-based genes-attributes resulting from data in Fig. 1.

For every $u \in U$ and $a \in A$, we calculate the *expected rank* $\tilde{a}(u) \in [0, N-1]$ of $a(u)$. We assume a uniform random distribution of the missing value ranks. Simple combinatorial calculations result in the following formula:

$$\tilde{a}(u) = \begin{cases} a(u) + M(a)(a(u)+1)/(N - M(a) + 1) & \text{if } a(u) \neq * \\ (N-1)/2 & \text{if } a(u) = * \end{cases} \tag{4}$$

It is worth noticing that the sum of the expected ranks of observations is the same for every gene, equal to the sum of rankings in case of non-missing values, that is $N(N-1)/2$. For example, in Fig. 3 each row sums up to 36.

A	u_0	u_1	u_2	u_3	u_4	u_5	u_6	u_7	u_8
$\tilde{a}_0$	8.0	0.125	6.875	5.75	3.5	3.5	1.25	4.625	2.375
$\tilde{a}_4$	3.5	6.875	3.5	4.625	2.375	1.25	0.125	8.0	5.75
$\tilde{a}_5$	6.875	2.375	5.75	3.5	1.25	0.125	8.0	3.5	4.625
$\tilde{a}_6$	3.5	2.857	3.5	5.429	4.143	1.571	0.286	6.714	8.0
$\tilde{a}_9$	0.125	8.0	3.5	3.5	4.625	6.875	1.25	5.75	2.375
$\tilde{a}_{17}$	4.625	2.375	3.5	3.5	6.875	8.0	5.75	1.25	0.125
$\tilde{a}_{18}$	4.625	2.375	3.5	6.875	5.75	8.0	3.5	0.125	1.25

Fig. 3. The expected ranks for the data table from Fig. 1. The ranks of not displayed genes are the same as in Fig. 2 because they have no missing values.

6 Rough entropy for the missing values

The expected expression ranks can be compared using the standard Euclidean distance, referred to as to the *revised* Spearman distance in Section 7. The fact that the coordinates of the expected rank vectors always sum up to the same value assures a standardized comparison. However, if we want the directly and reversely correlated genes to be grouped together, we should consider a measure with properties analogous to the rough entropy distance.

To adapt $\varrho_H(a,b)$ for the missing values, we do not need to calculate the expected ranks. Instead, while discretizing the attributes $a \in A$ with respect to their values $a(u) \in \{0,\ldots,N-M(a)-1\} \cup \{*\}$ we should estimate the probability that the values of objects $e \in U$ are greater/lower than $a(u)$.

Let us focus, e.g., on the probability that the rank $a(e)$ is lower than $a(u)$, denoted by $P(a(e) < a(u))$. We have the following estimation cases:

1. $a(u) \neq * \wedge a(e) \neq *$ – the ranks are fully comparable
2. $a(u) \neq * \wedge a(e) = *$ – we estimate P using the $a(u)$'s rank
3. $a(u) = * \wedge a(e) \neq *$ – we estimate P using the $a(e)$'s rank
4. $a(u) = * \wedge a(e) = *$ – the ranks are totally incomparable

The proposed estimations are given in Fig. 4. In particular, $a(u)/(N-M(a))$ is the percentage of known ranks below $a(u) \neq *$ and can be applied to estimate a chance that $* < a(u)$. Further, $(N-M(a)-a(e))/(N-M(a)+1)$ scales the chances that $a(e) < *$, beginning from $1-1/(N-M(a)+1)$ for $a(e)=0$, down to $1/(N-M(a)+1)$ for the greatest $a(e) = N-M(a)-1$.

	$a(u) \neq *$	$a(u) = *$
$a(e) \neq *$	$1 \leftrightarrow a(e) < a(u)$ $0 \leftrightarrow a(e) \geq a(u)$	$\frac{N-M(a)-a(e)}{N-M(a)+1}$
$a(e) = *$	$\frac{a(u)}{N-M(a)}$	$1/2$

Fig. 4. Probability $P(a(e) < a(u))$ depending on the values of $a(u)$ and $a(e)$.

Given the above local estimates for objects $u, e \in U$, we derive the probabilities $P(a_u = v)$, $P(b_u = w)$, $P(a_u = v, b_u = w)$, $v = 0,1$, $w = 0,1$, and use them to calculate $H(a_u)$, $H(b_u)$, and $H(a_u, b_u)$. As an example, let us focus on $P(a_u = 0)$ and $P(a_u = 0, b_u = 0)$. We can define them as

$$\begin{array}{c} P(a_u = 0) = 1/N \cdot \sum_{e \in U} P(a(e) < a(u)) \\ P(a_u = 0, b_u = 0) = 1/N \cdot \sum_{e \in U} P(a(e) < a(u)) P(b(e) < b(u)) \end{array} \tag{5}$$

where

$$P(a_u = 0) = \begin{cases} a(u)/(N-M(a)) & \text{if } a(u) \neq * \\ 1/2 & \text{if } a(u) = * \end{cases} \tag{6}$$

7 The experimental framework

We tested the Spearman and rough entropy distances using already mentioned GenOM system [6], based entirely on the paradigm of self-organizing maps [9]. We simply adjusted GenOM to handle the new data formats.

In case of the Spearman-related technique we operate with the vectors of non-negative real values summing up to $N(N-1)/2$, where N is the number of experiments in the given data system $\mathbb{A} = (U, A)$. We cluster the expected rank vectors resulting from (4). The following *revised* Spearman distance is a straightforward modification enabling to deal with such vectors:

$$\widetilde{\varrho_S}(a,b) = \sqrt{\frac{\sum_{u\in U}\left(\tilde{a}(u) - \tilde{b}(u)\right)^2}{N(N-1)}} \tag{7}$$

To apply the rough entropy distance, we refer to the rank-based attributes $a : U \to \{0, \ldots, N - M(a) - 1\} \cup \{*\}$ and calculate the values of $\varrho_H(a,b)$, $a, b \in A$, as described in Section 6. In this case, we had to modify additionally some technical details related to strengthening similarities between the gene expression vectors, crucial for the self-organizing map learning process.

The partial results of the GenOM application are illustrated in Fig. 5, for both studied types of distances. The objective is to get clusters of genes with similar functionality, possibly coding the same proteins. The interesting sets of genes are well recognizable by their group names (like e.g. FLT1 for a_0 and a_6), as well as additional functional descriptions (like e.g. a relation to the *tyrosine kinase* protein in case of a_0, a_6, and a_5). Let us consider the two following examples:

1. Genes a_8, a_9 belong to the JUNB group. A high correlation of their expressions would support an idea of clustering genes based on the microarray data. The revised Spearman and rough entropy distances are 0.12 and 0.16, respectively. Although both these values are relatively low on the $[0, 1]$ scale, it is not enough to cluster them together using SOM. Still, a_8 and a_9 have relatively closer positions on the Spearman-related grid. It seems to correspond to a lower Spearman distance between them.
2. Genes a_0, a_6 belong to the FLT1 group. We can see that they are put into the same SOM clusters in case of both the Spearman and rough entropy distances. This is a strong evidence, although the distances are equal, respectively, to 0.56 and 0.27 – much more than in the previous example. While comparing a_0 and a_6 we have 3 out of 9 coordinates with the missing values involved. In spite of that, a_0 and a_6 are more similar to each other than to any other significant genes considered in Fig. 5. Out of 20, there are no other genes grouped together with a_0 and a_6.

The above short case studies show that there is still a lot to be done in interpreting the results provided by the proposed distances. However, as a starting point for further analysis, the observed performance is quite promising.

A	Gene Description	Entr.	Spea.
a_0	115575 FLT1 fms-related tyrosine kinase 1	0 x 1	0 x 0
a_1	101364 FGFR1 fibroblast growth factor receptor 1	5 x 2	9 x 6
a_2	119431 FGFR1 fibroblast growth factor receptor 1	9 x 8	0 x 2
a_3	102991 CSF1R colony stimulating factor 1 receptor	7 x 9	8 x 0
a_4	101986 CSF1R colony stimulating factor 1 receptor	0 x 7	3 x 1
a_5	99123 FLT3LG fms-related tyrosine kinase 3 ligand	1 x 0	7 x 9
a_6	107367 FLT1 fms-related tyrosine kinase 1	0 x 1	0 x 0
a_7	111813 JUN v-jun avian sarcoma virus 17 oncogene homolog	1 x 7	7 x 9
a_8	100044 JUNB jun B proto-oncogene	9 x 1	5 x 2
a_9	115464 JUNB jun B proto-oncogene	7 x 7	6 x 0
a_{10}	99028 KIT v-kit feline sarcoma viral 4 oncogene homolog	5 x 8	9 x 6
a_{11}	100337 MDM2, human homolog of p53-binding protein	9 x 9	0 x 1
a_{12}	119774 SHC1 SHC transforming protein 1	6 x 0	7 x 6
a_{13}	111828 SHC1 SHC transforming protein 1	5 x 8	7 x 6
a_{13}	116399 SHB SHB adaptor protein	5 x 1	7 x 5
a_{15}	121008 TP53 tumor protein p53 (Li-Fraumeni syndrome)	3 x 6	6 x 9
a_{16}	111192 TP53 tumor protein p53 (Li-Fraumeni syndrome)	1 x 6	2 x 6
a_{17}	113533 SRCL scavenger receptor with C-type lectin	5 x 6	3 x 1
a_{18}	109299 SRCL scavenger receptor with C-type lectin	9 x 3	3 x 1
a_{19}	118000 LYN v-yes Yamaguchi sarcoma viral 1 oncogene hom.	5 x 2	8 x 3

Fig. 5. Previously considered 20 genes taken from the *Synovial Sarcoma* data, with the functional descriptions included. Columns *Entr.* and *Spea.* provide the two-dimensional coordinates resulting from GenOM for the rough entropy distance calculated as in Section 6 and the revised Spearman distance (7), respectively. GenOM was applied to all 5520 available genes in both cases.

8 Conclusions

We developed a new approach to compare the genes basing on their possibly incomplete expression characteristics. It is entirely based on the ranks calculated from the original microarray data for particular genes-attributes. The missing values are not ranked – instead we show how to calculate the expected ranks and the expected results of the rank comparisons. It enables to apply the previously studied rank-based distances based on the Spearman correlation and rough entropy to the clustering of partially missing data.

The results reported in Section 7, obtained for the data set related to a selected type of soft tissue tumor, confirm intuitions behind the proposed method. However, the actual verification of the proposed approach is possible only via further cooperation with the domain experts.

Acknowledgements: Supported by the research grant from Natural Sciences and Engineering Research Council of Canada, as well as by the Research Centre of Polish-Japanese Institute of Information Technology.

References

1. Alizadeh AA et al (2000) Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. Nature 403:503-511.
2. Baldi P, Hatfield WG (2002) DNA Microarrays and Gene Expression: From Experiments to Data Analysis and Modeling. Cambridge University Press, Cambridge.
3. de Brevern AG, Hazout S, Malpertuy A (2004) Influence of microarrays experiments missing values on the stability of gene groups by hierarchical clustering. BMC Bioinformatics 5:114.
4. Dembele D, Kastner P (2003) Fuzzy C-means method for clustering microarray data. Bioinformatics 19:973-980.
5. Friedman JH, Hastie T, Tibshirani R (2001) The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer, Berlin Heidelberg New York.
6. Grużdź A, Ihnatowicz A, Ślęzak D (2005) Interactive SOM-Based Gene Grouping: An Approach To Gene Expression Data Analysis. In: Proc of ISMIS 2005, Springer, Berlin Heidelberg New York.
7. Kapur JN, Kesavan HK (1992) Entropy Optimization Principles with Applications. Academic Press, San Diego.
8. Khan AH, Ossadtchi A, Leahy RM, Smith DJ (2003) Error-correcting microarray design. Genomics 81:157-165.
9. Kohonen T (1982) Self-organized formation of topologically correct feature maps. Biological Cybern 43:59-69.
10. Liu JS, Zhang, JL, Palumbo MJ, Lawrence CE (2003) Bayesian Clustering with Variable and Transformation Selections. In: Bayesian Statistics 7. Oxford University Press, Oxford, pp 249-275.
11. Oba S et al (2003) A Bayesian missing value estimation method for gene expression profile data. Bioinformatics 19:2088-2096.
12. Pawlak Z (1991) Rough sets – Theoretical aspects of reasoning about data. Kluwer Academic Publishers, Dordrecht.
13. Rebhan M, Chalifa-Caspi V, Prilusky J, Lancet D (1997) GeneCards: encyclopedia for genes, proteins and diseases. Weizmann Institute of Science, Bioinformatiecs Unit and Genome Center.
14. Ross DT et al (2000) Systematic variation in gene expression patterns in human cancer cell lines. Nat Genet 24:227-235.
15. Safran M et al (2003) Human Gene-Centric Databases at the Weizmann Institute of Science: GeneCards, UDB, CroW 21 and HORDE. Nucleic Acids Res 31(1):142-146.
16. Ślęzak D (2005) Rough entropy – non-parametric approach to measuring dependencies in quantitative data. In preparation.
17. Spellman PT et al (1998) Comprehensive identification of cell cycle-regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization. Mol Biol Cell 9:3273-3297.
18. Tamayo P et al (1999) Interpreting patterns of gene expression with self-organizing maps: methods and application to hematopoietic differentiation. Proc Natl Acad Sci USA 96(6):2907-2912.
19. Troyanskaya O et al (2001) Missing value estimation methods for DNA microarrays. Bioinformatics 17:520-525.

Estimation the Rhythmic Salience of Sound with Association Rules and Neural Networks

Bożena Kostek[1], Jarosław Wójcik[2], and Piotr Holonowicz[2]

[1] Gdansk University of Technology, Narutowicza 11/12, Gdansk, Poland
[2] Wroclaw University of Technology, Wyb. Wyspianskiego 27, Wroclaw, Poland

Abstract. In this paper experiments done towards improving the performance of systems retrieving musical rhythm are described. Authors briefly review machine learning models used to estimate tendency of sounds to be located in accented positions. This is done on the basis of their physical attributes such as duration, frequency and amplitude. For this purpose Data Mining association rule model and neural networks with discrete output - LVQ networks are used. By means of evaluation method introduced by the authors it is possible to compare the results returned by both models. This work aims at retrieving multi-level rhythmic structure of a musical piece on the basis of its melody, which may result in systems retrieval systems for automatic music identification.

1 Introduction

Content-based music retrieval is one of the multimedia retrieval research areas. Musicologists claim that melody and rhythm are two main elements of a musical piece [1], that is why melodic and rhythmic contents are usually analyzed in music retrieval research. A number of research was done in the area of melody retrieval, i.e. [14,16]. Melody retrieval systems can now accept hummed queries and retrieve melodies even though users make musical mistakes in queries. Music information retrieval area concerning rhythm has not been well explored, contrarily to melody-based information retrieval. Scientists search for characteristic rhythmic pattern of the known length in a piece [2] or try to find length and onsets of rhythmic patterns to a given melody [3,11]. Other rhythm finding works are, among others, by Povel and Essens [10] or Parncutt [9]. Most of approaches are based on generative theory of tonal music [7]. Dixons and Rosenthals systems form and then rank the rhythmical hypotheses, basing mainly on musical salience of sounds. However salience functions proposed by Dixon and Rosenthal are based on human intuition only. In this paper we describe how we adopted artificial intelligence learning techniques to find salience functions on the basis of real-life musical files. Melody is built of sounds having physical attributes - duration, amplitude, frequency - appearing subsequently in a proper moment called the onset time. Some onset times are accented. They are equally spaced and they create rhythmic levels - pairs of onsets and periods. The two levels are related if a period of the first of them is a natural multiply of other period

levels, and if some of their onsets are common ones. Related rhythmic levels are collected into families and considered as a hypothetical rhythm of a piece. Such a family is called rhythmic hypothesis. Formal definitions and the algorithm of forming hypotheses are described in details by the authors [6,15]. One can find other features that are related to melody and rhythm of a musical piece. However, the larger the set of features, the more likely it contains irrelevant or redundant attributes. In such a case the first goal of experiments should be finding the most significant features. Swiniarski proposes that feature selection can be treated as the process of searching an optimum subset of features from the original set of pattern features [13]. This can be done by defining a criterion according to which this optimum subset is to be found. The optimum subset guarantees the accomplishment of a processing goal while minimizing a defined feature selection criterion and results in a diminished number of features needed in the process [13]. However, in this work features are assigned based on the experts' musicological knowledge, the number of attributes being quite low. The task of the decision systems, that are used in the experiments, is to test whether features selected are sufficient for accomplishing the automatic rhythm finding.

2 Estimating Musical Salience of Sounds with Neural Network Approach

Self-organizing networks can learn to detect regularities and correlation in their input and adapt their future responses to that input, accordingly. The neurons of competitive networks (layers) learn to recognize groups of similar input vectors. Learning Vector Quantization (LVQ) is a method for training competitive layers in a supervised manner. A competitive layer automatically learns to classify input vectors into subclasses, then a linear layer transforms them into target classes chosen by the user. The subclasses that are found by the competitive layer are dependent only on the distance between input vectors, thus if two input vectors are very similar, the competitive layer will probably put them into the same subclass. The LVQ network consists of two layers: the first is a competitive layer, the second one is a linear layer. The competitive layer learns to classify input vectors into subclasses. In the beginning, the negative distances between the input vector and input weight (IW) vectors are calculated. The distance vector consists of n_c elements, where n_c is the number of neurons in the competitive layer. The net input vector is the sum of the distance vector and the bias vector. The competitive transfer function finds an appropriate subclass by seeking in the network an input vector for the most positive or the least negative value, depending on the bias vector. If all elements of bias vector are zeros then the output subclass number is the position of the least negative value in the net input vector, otherwise the output subclass number is the position of the most positive value of that vector. The role of bias is balancing the activation of

the neurons. This causes dense regions of the input space to be classified in more subsections. Both the competitive and linear layers have one neuron per subclass or per target class. Thus, the competitive layer can learn up to n_c subclasses. These, in turn, are combined by the linear layer to form n_t target classes. Value n_c is always larger than n_t. For example, suppose neurons 1, 2, and 3 in the competitive layer, they all learn subclasses of the input space that belongs to the linear layer target class No. 2. Then competitive neurons 1, 2 and 3 will have weights of 1 to neuron n_2 in the linear layer, and weights of 0 to all other linear neurons. Thus, the linear neuron produces 1 if any of the three competitive neurons (1, 2 and 3) win the competition. This is the way the subclasses of the competitive layer are combined into target classes in the linear layer. LVQ networks are trained in a supervised manner. Therefore learning data consist of pairs $\{p,t\}$ where p and t are input and desired output vectors respectively. The output vector t consists of values 0 and single value 1 placed at the position corresponding to the number of the class a given element p belongs to. During the q-th epoch vector $p(q)$ is presented at the input then the output from the network a_2 is compared to t. Let i^* be a position in t where 1 occurs and j^* the position where 1 occurs in t. If $i^* = j^*$, this means that p is classified correctly, then:

$$_{i^*}IW^{1,1}(q) =_{i*} IW^{1,1}(q-1) + \alpha(p(q) -_{i^*} IW^{1,1}(q-1)) \tag{1}$$

otherwise (p classified incorrectly)

$$_{i^*}IW^{1,1}(q) =_{i*} IW^{1,1}(q-1) - \alpha(p(q) -_{i^*} IW^{1,1}(q-1)) \tag{2}$$

The i^*-th row of IW matrix is adjusted in such a way as to move this row closer to the input vector p if the assignment is correct, and to move the row away from p otherwise. Described corrections made to the i^*-th row of $IW^{(1,1)}$ can be made automatically without affecting other rows of $IW^{(1,1)}$ by backpropagating the output errors back to layer 1. Such corrections move the hidden neuron towards vectors that fall into the class for which it forms a subclass, and away from vectors that fall into other classes.

3 Data Mining Estimation of Musical Salience

We also used Data Mining technique, proposed by [8] in order to estimate musical salience. In [8] *support* and *confidence* formulae can also be found. This approach explores training data set and finds tendencies, which determine knowledge used to predict most probable associations between attributes in testing set. If the tendencies discovered are confirmed in tests, the knowledge obtained can be used for other melodies to rank rhythmical hypotheses. In the association rule model there is a set of attributes in learning table T, some of which are classifying attributes. The rows of the table are teaching objects. In our approach the row is a sound. Attributes can have Boolean values 0 or

1. A rule is a statement saying that appearance of values 1 in a certain set of attributes causes that classifying attribute usually also have value 1. An example of a rule can be "long sounds tend to be placed in accented positions of the musical piece". A statement $X \rightarrow Y$ can be acknowledged as a rule if its *confidence* in table T has higher values than other rules.

4 Neural Network Experiments

We conducted experiments using Artificial Neural Networks. Learning and testing sets were created from MIDI files of various styles. We divided the data in two databases:

- single-track melodies: 20 351 sounds divided into 10 learning and 10 testing sets
- multi-track musical pieces: 42 998 sounds divided into 21 learning and 21 testing sets

The size of testing sets was averagely 2.5 times larger than the learning ones. Music files were obtained from the Internet using a web robot. For the training purpose we found accented locations in each melody. Then, we fed the network with musical data. One of tested networks had three inputs - one for each physical attribute of a sound (duration, frequency and amplitude), the second group consisted of three networks having one input each for each physical attribute of a sound. A desired output of the network could adopt one of two values 1, if the sound was accented, or 0 if it was not. In the testing stage the LVQ network states whether a sound is accented or not according to the knowledge received in the learning stage. Outputs given by such a network are compared to real outputs. After the testing phase, the set of all sounds can be divided into four subsets (see the Tab. 1), because there exist four possible combinations of the desired and network outputs. The network accuracy is formulated as a ratio of number of accented sounds, which were accurately detected by the network (number of elements in subset 4) to the

Table 1. Possible combinations of real and network outputs

Description	Desired output	Network output
Sound not accented, accurately detected by network	0	0
Sound not accented but falsely detected as accented	0	1
Sound accented, unfortunately not detected	1	0
Sound accented, accurately detected	1	1

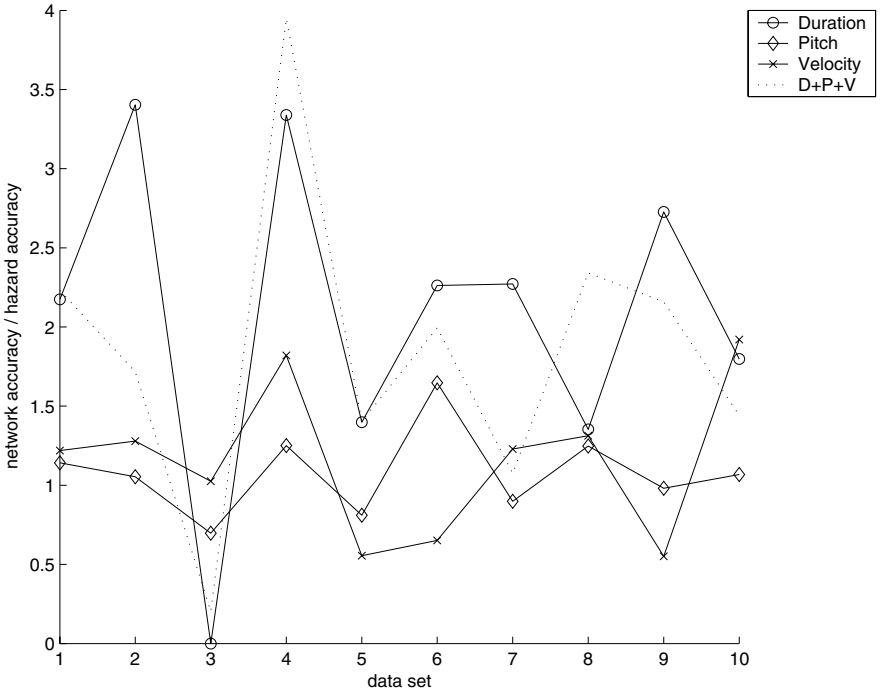


Fig. 1. Accuracy of four networks for single-track melodies (10 training sets and 10 testing sets)

number of accented sounds in a melody (number of elements in the sum of subsets 3 and 4).

network accuracy = number of accurately detected accented sounds / number of all accented sounds

Since the network accuracy depends on the number of accents given by the network, we introduce a new measure, a so-called *hazard accuracy*, which determines how accurately accented sounds could be found if they were hit in a randomize way. After dividing the network accuracy by the hazard accuracy it becomes clear, how many times the network recognizes accented sounds better than a blind choice. This approach also helps to determine how well network recognizes accented sounds if it takes into consideration single physical attributes or three of them simultaneously. The hazard accuracy depends on the number of accents given by the network (number of elements in the sum of subsets 2 and 4) and the number of all sounds in a set (number of elements in the sum of all four subsets).

hazard accuracy = number of accented sounds detected by the network / number of all sounds

We used single-track melodies as learning/testing set. Fig. 1 presents how accurately four networks found the accented sounds. There are three lines

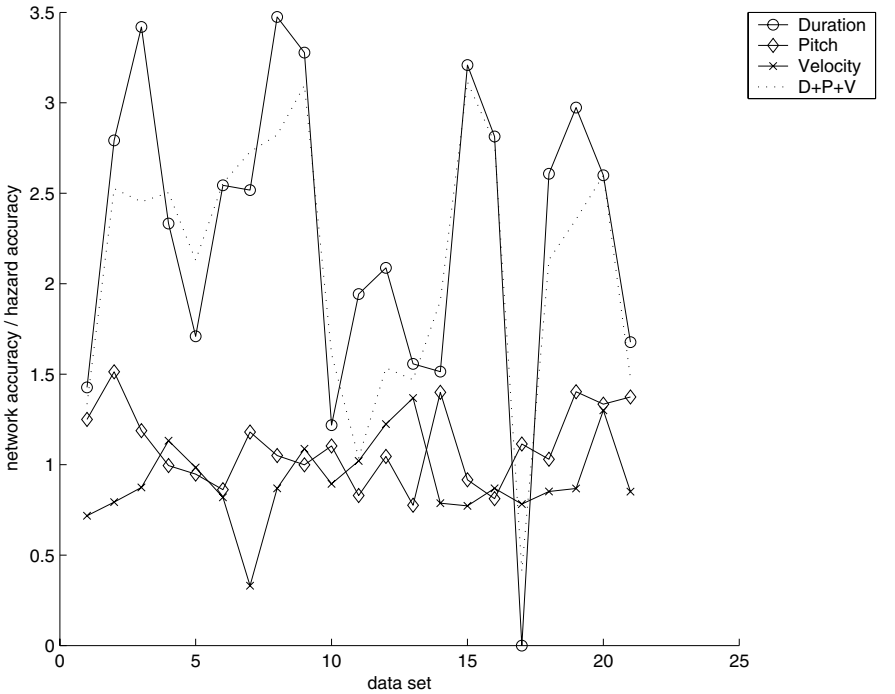


Fig. 2. Accuracy of four networks for multi-track musical pieces (21 training sets and 21 testing sets)

Table 2. Possible combinations of real and network outputs

		Network 1 Duration	Network 2 Frequency	Network 3 Amplitude	Network 4 D+F+A
10 sets	Mean (average) value	2.07	1.08	1.16	1.85
	Standard deviation	1.01	0.27	0.48	0.98
	Std. dev. / Mean value	0.49	0.24	0.42	0.53
21 sets	Mean (average) value	2.27	1.10	0.91	2.12
	Standard deviation	0.86	0.21	0.22	0.72
	Std dev. / Mean value	0.38	0.20	0.24	0.34

presenting the results of networks fed only with one attribute and one line representing the network fed with all three physical attributes. We tested network accuracy on single-track melodies. We conducted the analogical experiment on multi-track musical pieces – see results in Fig. 2. We averaged the above results in order to get comparable, single numbers assigned for each of the networks. We also counted standard deviation (see Tab. 2) and divided average values by standard deviations. This fraction helps to compare stability of the results for all networks. The lower value of the fraction, the more stable results.

5 Data Mining Association Rule Experiments

We also conducted experiments using the Data Mining association rule model to estimate salience of sounds in a melody. The testing set contained 36 966 sounds from 191 melodies. Since in the association rule model attributes contained in learning table can adopt only 1 or 0 values, thus they had to be preprocessed. Discretization was performed for all three attribute values. We used equal subrange quantization method. Other discretization methods can be also applied to this task ([4,5], and [13]). For each physical attribute we found its minimum and maximum values in a piece and divided subranges by thresholds placed according to the formula: $MinValue + (MaxValue MinValue) \cdot j \ / \ m$ for $j = 0, 1, 2...m$, where *m-1* is the number of subranges. We conducted experiments for 10 various numbers of subranges belonging to a range from 3 to 100. Using formulae introduced and presented in the former paragraph of this paper we received 10 tables with information about all rules, their supports and confidences in table columns. Rows of those tables are rules, number of rows of each table is 3m+1. For each physical attribute it is possible to find a rule or rules with the highest confidence. We used rules along with their confidences to propose ranking hypothesis functions. We summed all maximum confidences. The received sum is called *SMC* - sum of maximum confidences. The value of each physical attribute of a sound, for which we want to calculate a salience function value, is first quantized using the same number of subranges as in the learning stage. Let the subranges where the value falls be i_{dur} , i_{frq} and i_{amp}. After reading the values of confidences C from the table of all rules, we receive $C(i_{dur})$, $C(i_{frq})$ and $C(i_{amp})$. We propose the following ranking functions basing on Data Mining knowledge:

$$RANK1 = \frac{C(i_{dur}) + C(i_{frq}) + C(i_{amp})}{SMC} \tag{3a}$$

$$RANK2 = \frac{\frac{C(i_{dur})}{MaxC_{dur}} + \frac{C(i_{frq})}{MaxC_{frq}} + \frac{C(i_{amp})}{MaxC_{amp}}}{3} \tag{3b}$$

$$RANK3 = \frac{C(i_{dur})}{MaxC_{dur}} \tag{3c}$$

Formulae counting $RANK4$ and $RANK5$ are analogical to $RANK3$, but they concern frequency and amplitude respectively. The first two functions take into account all physical attributes simultaneously, the remaining ones are consistent with $RANK1$ and $RANK2$, but they consider attributes separately. The values of all above formulae are normalized, they fall within the interval $< 0, 1 >$. In Data Mining experiments we compared our formulae with the ones proposed in research related. In [3] two combinations of physical attributes values of sounds are proposed - a linear combination (additive function) and the multiplicative function. We used *precision/recall* method to validate accuracy of each of the above given functions.
Precision = N / number of documents in answer
Recall = N / number of relevant documents in database
N denotes a number of relevant documents in answer. In our evaluation proposal a single sound plays a role of a document, an accented sound is a relevant document. We sort sounds descending according to the value of each ranking function. In the answer we place highly ranked sounds. Number of sounds placed in the answer equals a number of the relevant documents (sounds placed in the accented positions). This results in equality of precision and recall giving a single measure, allowing for an easy comparison of ranking approaches. Fig. 3 presents the accuracy of all seven described ranking functions. In experiments we counted precisions/recalls for one piece, first, then

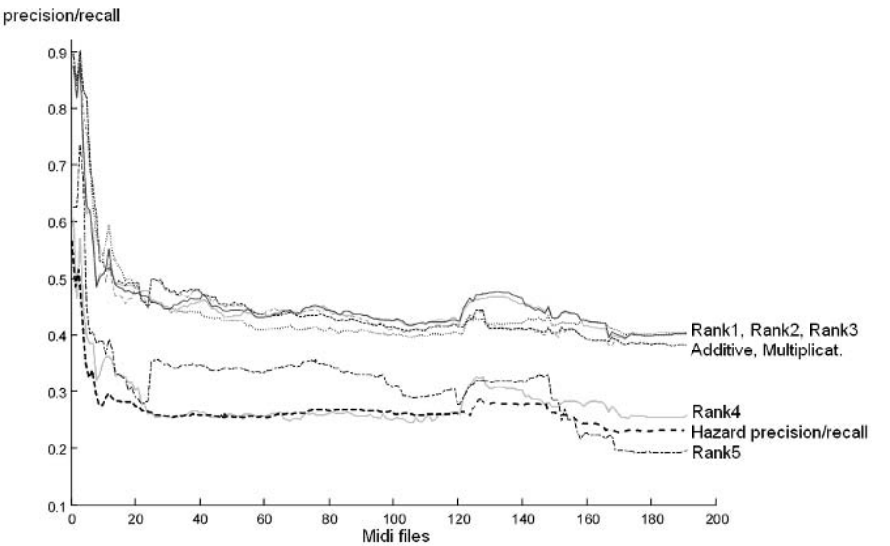


Fig. 3. Precision/recall of retrieval ranking functions approach

we added to the learning table sounds from one more piece, and counted precisions/recalls again. We repeated this action as many times as many pieces

we had. We used this approach in order to check, whether the value of precision/recall stabilizes.

6 Conclusions and Future Directions

We used real-life musical files in learning and testing processes and we found dependencies between physical attributes of sounds and their rhythmical salience. We describe our conclusions below.

1. According to our observations, duration is the only attribute, which should be considered in the ranking hypotheses phase. Neural network accuracy based on frequency and amplitude oscillates around the hazard accuracy (see Fig. 1 and Fig. 2), those two attributes are too much dependent on learning/testing data set choice. The same conclusions can be done on the basis of Fig. 3 - ranking functions awarding duration retrieve accented sounds much better than RANK4 and RANK5 functions, which take into consideration frequency and amplitude, respectively. What is more, in the association rule model it is possible to conclude that long sounds tend to be accented.
2. Precision/recall of RANK3 stabilizes after adding about 30 pieces to a testing/training Data Mining set, thus the duration-based salience of a sound can be considered as certain. RANK4 and RANK5 loose stability even after 120th piece - those two attributes are very dependent on training/testing data. Consequently, RANK1, RANK2 and both Dixons precision/recall formulae are less stable in those locations. This is why we recommend using duration only in calculating the sound rhythmic salience.
3. Results of experiments for single track melodies and multi-track musical pieces are consistent. In real music retrieval systems it seems to be reasonable to take sound duration only into account either for melodies or polyphonic, multi-instrumental pieces.
4. Rhythmic salience depends on physical attributes in a rather simple way. Neural Networks fed with combination of attributes performed even slightly worse than the ones fed with single attribute (duration).
5. In association rules experiment performance of salience functions appeared to depend on the number of discretization subranges. We observed that while conducting experiments for larger numbers of subranges the system performance still grew up from 3 up to about 30 subranges. For learning/testing datasets preprocessed with more than 30 subranges we did not observe further growth of performance. The relations between performances of functions awarding duration, frequency and amplitude remained unchanged, however.

We conducted some preliminary experiments to rank rhythmic hypotheses, using knowledge described in this paper. The best hypotheses in almost all

tested pieces were ranked higher than their positions in the randomize ranking process, some of them were even ranked first. However, periodicity of musical phrases should also be concerned in order to improve accuracy of hypotheses ranking. The next aim of the research conducted is to test these data with the rough set-based system, which will allow for checking our findings at this stage. This especially concerns feature selection, optimum choice of the number of subranges and the quality of rules derived.

References

1. Byrd, D. and Crawford T. (2002) Problems of music information retrieval in the real world, Information Processing and Management, vol. 38, Issue 2, 249–272
2. Chin, F. and Wu, S. (1992) An Efficient Algorithm for Rhythm-finding, Computer Music Journal, MIT, vol. 16, No. 2, 35-44
3. Dixon, S. (2001) Automatic Extraction of Tempo and Beat from Expressive Performances, Journal of New Music Research, vol. 30, No. 1, Swets & Zeitlinger (March), 39–58
4. Kostek, B. (1999) Soft Computing in Acoustics, Applications of Neural Networks, Fuzzy Logic and Rough Sets to Musical Acoustics, Studies in Fuzziness and Soft Computing, Physica Verlag, Heidelberg, New York
5. Kostek, B. (1998) Computer Based Recognition of Musical Phrases Using The Rough Set Approach, J. Information Sciences, vol. 104, 15–30
6. Kostek B., Wojcik J. (2004) Forming and Ranking Musical Rhythm Hypotheses. Proceedings of Knowledge-Based Intelligent Information & Engineering Systems, Wellington, New Zealand (Lecture Notes in Artificial Intelligence)
7. Lerdahl, F. and Jackendoff, R. (1983) A generative theory of tonal music, MIT Press, Cambridge, MA
8. Mannila, H. (1996) Data mining: machine learning, statistics, and databases. 8th Intern. Conference on Scientific and Statistical Database Management, 2–9
9. Parncutt, R.A (1994) A perceptual model of pulse salience and metrical accent in musical rhythms, Music Perception, vol. 11(4), 409–464
10. Povel, D.J. and Essens, P. (1985), Perception of temporal patterns, Music Perception, vol. 2(4), 411–440
11. Rosenthal, D.F. (1992) Emulation of human rhythm perception, Computer Music Journal, vol. 16, No.1 (Spring), 64–76
12. Rosenthal, D.F. (1992) Machine Rhythm: Computer Emulation of Human Rhythm Perception, PhD thesis, MIT Media Lab, Cambridge, MASS
13. Swiniarski, R. (2001) Rough sets methods in feature reduction and classification, Int. J. Applied Math. Comp. Sci., vol. 11 (3), 565–582
14. Tseng, Y.H. (1999) Content-based retrieval for music collections, 22nd Intern. Conf. on Research and Development in Information Retrieval, 176–82
15. Wojcik J., Kostek B. (2004) Intelligent Methods for Musical Rhythm Finding Systems, chapter in Intelligent Techn. for Inconsistent Proc., vol. 10, 187-202
16. Wu, S. and Manber, U. (1992) Fast text searching allowing errors, Communications of the ACM, vol. 35, No. 10, 83–91

A Neuro-Fuzzy Classifier Based on Rough Sets

Huanglin Zeng[1] and Roman W. Swiniarski[2,3]

[1] Sichuan University of Sciences and Engineering, 643033, P. R. China
[2] San Diego State University Department of Computer Science San Diego, California U.S.A.
[3] IPI PAN, ul. Ordona 21, Warsaw Poland

Abstract. In this paper, we use the concept of rough sets to define equivalence classes encoding input data, and to eliminate redundant or insignificant attributes in data sets which leads to reduction of the complexity of designed systems. In order to deal with ill-defined or real experimental data, we represent input object as fuzzy variables by fuzzy membership function. Furthermore we incorporate the significance factor of the input feature, corresponding to output pattern classification, in order to constitute a fuzzy inference which enhances classification considered as a nonlinear mapping. A new kind of rough fuzzy neural classifier and a learning algorithm with LSE are proposed in this paper. The neuro-fuzzy classifier proposed here can realize a nonlinear mapping from the input feature vector space (that may have the overlapping characteristic) into the output classification vector space.

1 Introduction

In recent years we have witnessed a surge of interest in soft computing, which combines fuzzy logic, neural networks, rough sets, and other techniques [1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13]. Fuzzy set theory provides an approximate but effective and flexible way of representing, manipulating and utilizing vaguely defined data and information, as well as describing the behavior of systems that are too complex or ill-defined to admit of precise mathematical analysis by classical methods. Although the fuzzy inference tries to model the human thought process in a decision-making system, it does not discriminate attribute characteristics among various pattern classes, nor does it take into account the adaptive learning in a data-changing environment. Neural networks provide crude emulation of the human brain, that can store, learn and process information more like human beings, but they cannot make the dimensionality reduction in a system space, and sometimes require computationally intensive processing.

An attribute-oriented rough sets technique reduces the computational complexity of learning processes and eliminates the unimportant or irrelevant attributes so that the knowledge discovery in database or in experimental data sets can be efficiently learned. It therefore seems that a integration of the merits of these several technologies can ensure more efficient intelligent systems by evolving synergism between them, to handle real life ambiguous and changing situations, and to achieve tractability, robustness,

and low-cost design solutions [4], [5], [8], [9], [10], The main purpose of this article is providing a paradigm of hybridization in soft computing on pattern classification. At first, we try to use the concept of rough sets to define equivalence class encoding input data, and to eliminate redundant or insignificant attributes in data sets, and incorporate the significance factor of the input feature corresponding to output pattern classification in order to enhance fuzzy inference. Then the fuzzy approach is used to deal with either incomplete or imprecise or even ill-defined database, and to constitute a fuzzy decision table with reduced attributes. In section 4, a neuro-fuzzy classifier is proposed, and a supervised algorithm with the least squared error (LSE) criterion is suggested to train the system. Finally, the efficiency of applications of neuro-fuzzy classifier based on rough sets in real life pattern classification is briefly reviewed in the paper.

2 Knowledge encoding and attribute reduction using rough sets

In a general setting of the process of pattern classification, the first step relates to raw data acquisition and raw data preprocessing. The second step deals with the extraction and selection of features from input objects (subjects of classification). The main objective of feature selection is to choose minimal subset of features that retain the optimum salient characteristics necessary for the classification process and to reduce the dimensionality of the measurement space so that effective and easily computable algorithms can be devised for efficient classification. Let $S =< U, A >$ be a universe of a training set. Divide S into Q tables (U_l, A_l), $l = 1, 2, \cdots, Q$, corresponding to the decision classes. Here we have $U = U_1 \bigcup U_2, \cdots, U_Q$ and attributes $A_l = C \bigcup \{d_l\}$ and C, d_i are the antecedent and decision attributes respectively. Suppose that there is n_k objects of U_l that occur in S_l, $l = 1, 2, \cdots, Q$ and $\sum_{l=1}^{Q} n_{lk} = N$. An input vector $X_j = (x_{1j}, x_{2,j}, \cdots, x_{nj})$ in $n - dimensional$ input space is constituted with real-valued features taking values from the interval (x_{min}, x_{max}). When an input feature is numerical, it can be normalized by partitioning the value of input features into L intervals. An equivalence class of an input feature x_i is defined as:

$$[(x_i)]_R = \{x_i : \frac{(x_{max} - x_{min})k - 1}{L} \leq x_{ik} \leq \frac{(x_{max} - x_{min})k}{L}\}, k = 1, 2, \cdots, L \tag{1}$$

A significance factor of the input feature x_i $(i = 1, 2, \cdots, n)$ corresponding to output pattern classification W is defined by

$$\alpha_{xi}(W) = \frac{card[POS_X(W) - POS_{X-xi}(W)]}{card[U]}, \ i = 1, 2, \cdots, n \tag{2}$$

Here, U is the domain of discourse in training set, and $card[.]$ denotes the cardinality of a set. The term $POS_X(W) - POS_{X-xi}(W)$ denotes a change

of positive region of input vector with respect to output pattern classification when an input feature x_i is reduced. A significance factor $\alpha_{xi}(W)$ denotes dependency relation of the output pattern classification with respect to the input feature x_i $(i = 1, 2, \cdots, n)$. It can be taken into account for enhancing a classification ability of the decision algorithm since the larger $\alpha_{xi}(W)$ is, the more significant of input attribute x_i is with respect to the classification of output patterns. An input attribute x_i can be reduced when $\alpha_{xi}(W) = 0$. Sometime the input feature reduction means that the number of antecedent attributes is reduced with the help of a threshold value of significance factor $\alpha_{xi}(W)$. Then one can consider only those attributes that have numerical values greater than some threshold Th (for example: $0.20 \leq Th \leq 1$).

Let us consider the information system shown in Table 1. According to the objects given in the decision Table 1, (a, b, c) denotes antecedent attributes and $\{d, e\}$ denotes decision attributes.

Table 1. A decision table of classification information system

U	a	b	c	d	e
1-5	2	2	1	0	0
6-8	1	1	3.2	0	1
9-11	3.1	1	2	1	1
12-16	2	2	2	1	0
17-20	2	0.9	1	1	2
21-24	3	3.1	2	1	1
25-28	3	2	3	0	1
29-31	0.9	3	3	1	2

Based on formula (1), an input feature is normalized by partitioning the value of input features into 3 intervals. If we label the training sets shown in Table 1 as $1, ..., 8$, an equivalence class of an input vector constituted with features (a, b, c) is expressed by

$$U|(a, b, c) = \{\{1\}, \{5\}, \{2\}, \{8\}, \{3\}, \{4\}, \{6\}, \{7\}\} \tag{3}$$

An equivalence class of an output vector constituted with features (d, e) is expressed by

$$U|(d, e) = \{\{1\}, \{2, 7\}, \{3, 6\}, \{4\}, \{5, 8\}\} \tag{4}$$

If we will reduce the input feature a, then the equivalence class of the input vector constituted with features (b, c) is expressed by

$$U|(b, c) = \{\{1\}, \{5\}, \{2\}, \{8\}, \{7\}, \{3\}, \{4\}, \{6\}\} \tag{5}$$

Reduction of the input feature b gives the following equivalence class of the input vector (a, c)

$$U|(a, b) = \{\{1, 5\}, \{2, 8\}, \{3, 6\}, \{4\}, \{7\}\} \tag{6}$$

Reduction of the input feature c gives the following equivalence class of the input vector (a, b)

$$U|(a,b) = \{\{1,4\},\{2\},\{8\},\{3\},\{5\},\{6\},\{7\}\} \tag{7}$$

Based on rough sets [6], [8], [2], a positive region of input vector with respect to output pattern classification is

$$POS_P(W) = \{1,2,3,4,5,6,7,8\} \tag{8}$$

$$POS_{P-a}(W) = \{1,2,3,4,5,6,7,8\}, \tag{9}$$

$$POS_{P-b}(W) = \{3,4,6,7\},\ POS_{P-c}(W) = \{2,8,3,5,6,7\}$$

Based on formula (2), a significance factors of the input features corresponding to output pattern classification are expressed by

$$\alpha_a(W) = \frac{8}{8} - \frac{8}{8} = 0;\ \alpha_b(W) = \frac{8}{8} - \frac{4}{8} = 0.5;\ \alpha_c(W) = \frac{8}{8} - \frac{6}{8} = 0.125 \tag{10}$$

It can be seen that some of input attributes are more significant than others since their significance factors are larger. In the considered information system, the input feature a can be reduced since $\alpha_a(W) = 0$.

3 Fuzzy Representation and Input Pattern Reduction

In pattern classification of real-life data systems, input features can be imprecise or incomplete. The impreciseness or ambiguity of the input data may arise from various reasons. Based on the fuzzy sets concepts [2], [5], in this case it may become convenient to use linguistic variables and hedges to augment or even replace numerical input features. Each input feature F_j can be expressed in terms of membership values of fuzzy membership function. A fuzzy membership function is chosen by Gaussian -like function as follows

$$\mu_A(x) = exp(-\frac{1}{2}(\frac{x - c_i}{\sigma_i})^2) \tag{11}$$

where c_i denotes the center of a membership function, and σ_i denotes distribution of a membership function. The distribution of membership function σ_i is generally chosen this way that membership functions is fully overlapped, and c_i is defined by

$$c_i = \frac{x_{max} - x_{min}}{C - 1}(i-1) + x_{min},\ i = 1,2,\cdots,C \tag{12}$$

The term C denotes the number of linguistic variables used. It is assumed that input feature x_i takes real number from the interval (x_{min}, x_{max}). Let us consider an example where we let $C = 3$, and we choose three linguistic variables denoted as: *small, medium, large* with the overlapping partition.

Suppose, that an input vector is reduced to $s - dimensional$ vector with the help of $\alpha_a(W)$, a fuzzy membership function is used to express linguistic variables and the input vector in $s - dimensional$ input space is expressed by

$$\mu(X_j) = (\mu_l(x_{lj}), \mu_m(x_{lj}), \mu_s(x_{lj}), \cdots, \mu_l(x_{sj}), \mu_m(x_{sj}), \mu_s(x_{sj})) \quad (13)$$

in $3 \times s - dimensional$ fuzzy vectors space. Where $\mu_l(x_{ij}), \mu_m(x_{ij}), \mu_s(x_{ij})$ denote the membership functions of linguistic variables $large_l(x_i)$, $medium_m(x_i)$, $small_s(x_i)$ for the input feature x_i respectively. Similarly,

$$\mu(X_j) = (\mu_l(x_{lj}), \mu_m(x_{lj}), \mu_s(x_{lj}), \cdots, \cdots, \mu_l(x_{nj}), \mu_m(x_{nj}), \mu_s(x_{nj})) \quad (14)$$

is normalized by partitioning the value of input fuzzy features into L intervals. An equivalence class of a linguistic variable is defined as:

$$[_s(x_i)]_R = \{_s(x_i) : \frac{k-1}{L} \leq \mu_s(x_{ik}) \leq \frac{k}{L}\},\ k = 1, 2, \cdots, L \quad (15)$$

$$[_m(x_i)]_R = \{_m(x_i) : \frac{k-1)}{L} \leq \mu_m(x_{ik}) \leq \frac{k}{L}\},\ k = 1, 2, \cdots, L \quad (16)$$

$$[_l(x_i)]_R = \{_l(x_i) : \frac{k-1}{L} \leq \mu_l(x_{ik}) \leq \frac{k}{L}\},\ k = 1, 2, \cdots, L \quad (17)$$

Based on the definition of an equivalence class of a linguistic variable, we make the decision table in which

$$\mu(X_j) = (\mu_l(x_{lj}), \mu_m(x_{lj}), \mu_s(x_{lj}), \cdots, \mu_l(x_{sj}), \mu_m(x_{sj}), \mu_s(x_{sj}))$$

is normalized as L value partitions. For example, let $L = 5$, and $\mu_l((x_{ij}), \mu_m(x_{ij},$ and $\mu_s(x_{ij})$ be $1/5, 2/5, 3/5, 4/5, 1$ respectively. Input pattern reduction means that the size of each S_l $(i = 1, 2, \cdots, Q)$ is reduced with the help of a threshold value of membership function. Then, one can consider only those attributes that have a numerical value greater than some threshold Th (for example $0.2 \leq Th \leq 1$). Considering an information system with reduction of input feature a, a fuzzy membership function is chosen by Gaussian-like function

$$\mu_A(x) = exp(-0.5(\frac{x - c_i)}{\sigma_i})^2) \quad (18)$$

in order to consider a membership of three linguistic variables that are denoted as *small, medium*, and *large* which is expressed by

$$\mu(X_j) = (\mu_l(x_{2j}), \mu_m(x_{2j}), \mu_s(x_{2j}), \mu_l(x_{3j}), \mu_m(x_{3j}), \mu_s(x_{3j}))$$

in $2 \times 3 - dimensional$ fuzzy vectors for 2 input features. A decision table in which $\mu(X_j) = (\mu_l(x_{2j}), \mu_m(x_{2j}), \mu_s(x_{2j}), \mu_l(x_{3j}), \mu_m(x_{3j}), \mu_s(x_{3j}))$ is clamped as $L = 5$ value partitions, that is $\mu_l(x_{ij})$, $\mu_m(x_{ij})$, and $\mu_s(x_{ij})$ be

$0, 1/5, 2/5, 3/5, 4/5, 1$ respectively. Then, input patterns are reduced with the help of a threshold value $Th = 0.2$ of the membership function as shown in Table 2. In this table we label the training sets as $1, ..., 8$, where x_1, x_2, x_3 denote a, b, c,and $W_l,\ l = 1, 2, \cdots, 5$ denotes classification decision.

Table 2. A fuzzy membership function decision table of an information system

U	$\mu_l(x_2)$	$\mu_m(x_2)$	$\mu s(x_2)$	$\mu_l(x_3)$	$\mu_m(x_3)$	$\mu s(x_3)$	W
1	0.1	0.95	0.1	0	0.1	0.9	W_1
2	0	0.1	0.9	0.98	0.05	0	W_2
3	0	0.1	0.9	0.1	0.95	0.1	W_3
4	0.1	0.95	0.1	0.1	0.95	0.1	W_4
5	0	0.1	0.95	0	0.1	0.9	W_5
6	0.98	0.05	0	0.1	0.95	0.1	W_3
7	0.1	0.95	0.1	0.95	0.05	0	W_2
8	0.95	0.05	0	0.95	0.05	0	W_5

4 A Neuro-fuzzy Classifier Based on Rough Sets

If we assume that some attribute x_i eliminated since it is not significant for the output pattern classification W with $\alpha_{xi}(W) \approx 0$, then the rest of attributes from training data sets consists of pattern (input vectors) $X_j = (x_{1j}, x_{2j}, \cdots, x_{sj})^T \in \mathbf{R}^s$, where s is the number of feature of an input object. A neuro-fuzzy classifier based on rough sets can be constituted with three parts. The first part of the classifier performs the input fuzzy representation. The task of the fuzzy representation is mapping of input pattern (an input vector) $X_j = (x_{1j}, x_{2j}, \cdots, x_{sj})^T \in \mathbf{R}^s$ to a linguistic variable $\mu(X_j) = (\mu_l(x_{lj}), \mu_m(x_{lj}), \mu_s(x_{lj}), \cdots \mu_l(x_{sj}), \mu_m(x_{sj}), \mu_s(x_{sj}))^T$, denoted by $\mu(X_j) = \mu_p(x_{ij}),\ p = 1, 2, 3;\ i = 1, 2, \cdots, s$. A fuzzy membership function is expressed by

$$\mu_p(x) = exp(-0.5(\frac{x - c_i}{\sigma_i})^2),\ \ p = 1, 2, 3 \tag{19}$$

where $\mu_p(x)$ denotes $\mu_l(x_i), \mu_m(x_i), \mu_s(x_i),\ i = 1, 2, \cdots, s$.

The second part is the fuzzy inference in which we incorporate $\alpha_{xi}(W)$ (the significance factor of the input feature corresponding to the output pattern classification) in order to enhance fuzzy inference. It uses a sum operator of weighted product in inference network to represent fuzzy logic as follows

$$\nu_p = f(\sum_{i=1}^{s} \alpha_{xi}(W)\mu_p(x_i)),\ p = 1, 2, 3 \tag{20}$$

$$u_p = f(\nu_p r_{pp}(\sum_{k=1}^{3} \nu_k r_{pk})^{-1}),\ p = 1, 2, 3 \tag{21}$$

where $f(.)$ is the nonlinear function denoted by $f(x) = (1 + e^{-ax})^{-1}$, and a is a constant. It can be noted that in formula $u_p = f(\nu_p r_{pp}(\sum_{k=1}^{3} \nu_k r_{pk})^{-1})$ only r_{pk}, $p = 1, 2, 3$ need to be determined by learning processes.

In the third part, that is a multi-layer perceptron, the input of real object and fuzzy inference consequent are incorporated to yield an output classification decision, in which fuzzy variables are transformed into real processes outputs, in terms of the neural network computation, by

$$C_p = f(\sum_{i=1}^{s} x_i T_{ip}),\ p = 1, 2, 3 \tag{22}$$

It can be noted that if a multi-layer perceptron has the most simple constitution, then only T_{ip}, $i = 1, 2, 3$ need to be determined by learning processes. For the object represented by X_j, an output classification decision d_j is expressed by

$$d_j = A\, f(\sum_{p=1}^{3} C_p u_p) \tag{23}$$

Here, A is a normalized factor that is chosen based on the form of output classification decision. In order to obtain learning algorithm of a neuro-fuzzy classifier which based on an error backpropagation (BP), we define an error e_{lj} as $e_{lj} = W_l - d_j$ for an object $X_j \subset X$ being classified as $W_l\, l = 1, 2, \cdots, Q$. For a universe of the training set, a square error function defined by

$$E = \frac{1}{2} \sum_{l=1}^{Q} \sum_{j=1}^{N} e_{lj}^2 = \frac{1}{2} \sum_{l=1}^{Q} \sum_{j=1}^{N} (W_l - d_j)^2 \tag{24}$$

is considered as a criterion of minimization process. The standard gradient descent approach can be obtained by minimizing the square error function. For r_{pk}, $p = 1, 2, 3$; $k = 1, 2, 3$ using the chain rule for Eq. (24) we can write

$$\frac{\partial E}{\partial r_{pk}} = \frac{\partial E}{\partial e_{lj}} \frac{\partial e_{lj}}{\partial d_j} \frac{\partial d_j}{\partial u_p} \frac{\partial u_p}{\partial r_{pk}} = -e_{lj} f' C_p f_3' u_{rpk},\ \ p = 1, 2, 3;\ k = 1, 2, 3 \tag{25}$$

where

$$f_1' = \frac{df(y)}{dy} | y = \sum_{p=1}^{3} C_p u_p;\ f_3' = \frac{df(y)}{dy} | y = \nu_p r_{pp} (\sum_{k=1}^{3} \nu r_{pk})^{-1}, \tag{26}$$

and

$$u_{rpk} = \begin{cases} \frac{-\nu_p r_{pp} \nu_k}{(\sum_{k=1}^{3} \nu_k r_{pk})^2} & p \neq k \\ \frac{\sum_{k=1}^{3} \nu_k r_{pk} - \nu_p r_{pp} \nu_k}{(\sum_{k=1}^{3} \nu_k r_{pk})^2} & p = k \end{cases} \tag{27}$$

A general formula for updating the weights $r_{pk}, p = 1,2,3,\ k = 1,2,3$ is expressed as

$$\Delta r_{pk} = -\eta(t)\frac{\partial E}{\partial r_{pk}} = \eta(t) e_{lj} f_1' C_p f_3' u_{rpk},\ \ p = 1,2,3,\ \ k = 1,2,3 \tag{28}$$

For T_{ip}, $i = 1,\cdots,s$, $p = 1,2,3$, using the chain rule for Eq. (24) we can write

$$\frac{\partial E}{\partial T_{ip}} = \frac{\partial E}{\partial e_{lj}}\frac{\partial e_{lj}}{\partial d_j}\frac{\partial d_j}{\partial C_p}\frac{\partial C_p}{\partial T_{ip}} = -e_{lj} f_1' u_p f_2' x_i,\ \ i = 1,\cdots,s,\ \ p = 1,2,3 \tag{29}$$

where

$$f_1' = \frac{df(y)}{dy}|y = \sum_{p=1}^{3} C_p u_p;\ f_2' = \frac{df(y)}{dy}|y = \sum_{i=1}^{s} x_i T_{ip} \tag{30}$$

A general formula for updating the weights $T_{ip},\ i = 1,\cdots,s,\ \ p = 1,2,3$ is expressed as

$$\Delta T_{ip} = -\eta(t)\frac{\partial E}{\partial r_{pk}} = \eta(t) e_{lj} f_1' u_p f_2' x_i,\ i = 1,\cdots,s,\ p = 1,2,3 \tag{31}$$

where $\eta(t) = \frac{1}{|t|}$ denotes a learning factor in formula (4.9) and (4.11).

The presented method of classifier design has been used in numerical experiments in mammogram classification.

5 Summary

In this paper, we propose a method of rough neuro-fuzzy hybridization in soft co mputing in pattern classification. The merits of the neuro-fuzzy classifier prop osed here is that using the concepts of rough sets to define equivalence class e ncoding input data, eliminating redundant or insignificant attributes in data se ts so that reduces the computational complexity of learning processes, and incor porating the significance factor of the input feature corresponding to output pa ttern classification to constitute a fuzzy inference so that experimental data s ets can be efficiently learned. A integration of the merits of fuzzy and neural network technologies not only accommodates the overlapping classification, and therefore increases the performance of nonlinear mapping classification), but also ensures more efficient handling of real life ambiguous and changing situations and achieving tractability, robustness, and low-cost solutions.

References

1. Cichocki, A. (1993) Neural Networks for Optimization and Signal Processing, John Wiley & Sons Ltd. & B.G. Teubner, New York

2. Cios, K., Pedrycz, W., Swiniarski, R. (1998) Data mining methods for knowledge discovery, Kluwer academic publishers, Dordrecht
3. Dumitrescu, D. (2000) Fuzzy sets and their application to clustering and training, CRC Press, New York
4. Mitra, S. Pal, S. K. (2000) Rough-fuzzy knowledge-based networks, neuro-fuzzy pattern recognition methods in soft computing, John Wiley & Sons, Inc., 339-358
5. Pal, S., Mitra, S. (1999) Neuro-fuzzy pattern recognition methods in soft computing, A Wiley-interscience publication, John Wiley & Sons, Inc., New York
6. Pawlak, Z. (1999) Decision Rules, Bayes' rule and rough sets, N. Zhong, A. Skowron (eds): New directions in rough sets, data mining, and granular-soft computing, LNAI 1711, Springer-Verlag, 1-9
7. Ripley, B. D. (1996) Pattern recognition and neural networks, Cambridge University Press, New York
8. Skowron, A., Swiniarski, R. (2003) Information Granulation and Pattern Recognition. In S. K. Pal, L. Polkowski, A. Skowron (eds.) Rough-Neuro Computing: Techniques for Computing with Words Artificial Intelligence Series, Springer -Verlag, Berlin, Heidelberg, New York, 599-637.
9. Swiniarski, R., Skowron, A.(2004) Rough Sets Methods in Feature Selection and Recognition. Pattern Recognition Letters, 24(6), pp. 833-849.
10. Swiniarski, R., Hargis, L. (2001) Rough sets as a front end of neural-networks texture classifiers, International Journal, Neurocomputing. **36**, 85-102
11. Szczuka, M. (2000) Rough Sets and artificial neural networks, L. Polkowski, A Skowron (eds): Rough sets in knowledge discovery (2), Physica-Verlag, 449-469
12. Yao, Y. Y. (1998) A comparative study of fuzzy sets and rough sets, Information sciences, **109**, 227-242
13. Zhong, N. Skowron, A. (eds): (1999) New directions in rough sets, data mining, and granular-soft computing, LNAI 1711, Springer-Verlag, Berlin

Part IX

Invited Session: Knowledge Base Systems

Evolutionary Multi-Agent Model for Knowledge Acquisition

Wojciech Froelich

Institute of Computer Science, Silesian University,
ul. Bedzinska 39, Sosnowiec, Poland

Abstract. In this paper the conception of evolutionary multi-agent model for knowledge acquisition has been introduced. The basic idea of the proposed solution is to use the multi-agent paradigm in order to enable the integration and co-operation of different knowledge acquisition and representation methods. At the single-agent level the reinforcement learning process is realized, while the obtained knowledge is represented as the set of simple decision rules. One of the conditions of effective agent learning is the optimization of the set of it's features (parameters) that are represented by the genotype's vector. The evolutionary optimization runs at the level of population of agents.

1 Introduction

The researches being conducted in a domain of artificial intelligence has led to the formation of many methods that permit to solve effectively different and complex tasks. In case of difficult tasks in the field of knowledge acquisition, and particularly in case of dynamically changing environment or incomplete and uncertain data, it may be necessary to adjust every method to specific version of the task. Such necessity leads in consequence to changes in composition and parameters of algorithm being used (optimization through experimental researches). One of possible solutions is the use of hybrid methods, hierarchical composition of algorithms and multi-starting methods (incl. evolutionary). Such an approach is inspired by the observation of the natural world, where knowledge acquisition processes are realized on different levels of abstraction and by different representation structures(neural networks, evolution processes). The basic idea of the proposed conception is to apply multi-agent system [5] in order to enable the integration and co-operation of various methods of knowledge acquisition and representation. In the proposed model, the autonomous agent is understood as the system located within an environment, which constantly observes this environment and carries out in it the activities in order to execute the given task. The environment represents the problem being solved and at the same time enables the mutual agents' observation. In the environment there can be distinguished objects that are the source of information for the learning multi-agent system. These objects can undergo some interactions, and then the knowledge acquisition process is related to constant mutual interaction between the environment and system.

The task that the learning system has to perform can be applied to identify the features of objects, relations between them or can consist in specifying the effective strategy of objects' control. The issues being considered in this paper apply to the case, when the environment model, from which the knowledge is to be derived, is not entirely known. In such case, the knowledge acquisition takes place by the realization of the sequence of agent's decisions, that by intention are to lead to reaching the goal, while their intermediate effects (environment state changes) give information about its characteristic. So, even if the goal was not reached, agent's activities would give information resulting from the environment response. In the proposed model, the environment can be observed by agents through the set of changing signals. The environment signals can be interpreted as a-priori unknown, but measurable quantities represented by phenomena occurred in it. The set of environment signals is the source of information for all agents. Every environmental signal can change, in a discreet or constant way. Among the particular signals the time and spatial relations can occur. Agents placed within the environment create the set, which is called the multi-agent population. The initial agents' population is generated on the strength of the expert knowledge (the expert can be either the user or certain superior program) and is equipped with random set of features, including their spacing in the environment. In the environment, every agent undertakes the activities, as a consequence of which the changes of observed signals can appear. Agent's activities can mean the realization of single actions or the sequence of them. The reactive method of agents' activity is assumed, which means, that agents do not perform mathematical calculations and algorithmically controlled analysis of the history of their observations and actions. Planning in the meaning of building and estimation of behavioral patterns is not realized. Each of the agents has knowledge that can be classified according to its nature and the way of it's obtaining (given in brackets) as the following:

- model knowledge of the system handed by the designer (a-priori knowledge),
- evolutionary operators (a-priori knowledge),
- ontogenetic knowledge (rule knowledge base obtained through reinforcement learning),
- philogenetic knowledge (genotype obtained through multi-agent evolution process).

2 Multi-agent system

Multi-agent system is composed of the set of agents working in a common environment. The operation of this system will be considered in consecutive moments of time. We can say about dualistic nature of such defined system, in which on one hand we have to do with the environment that surrounds

the agents, and on the other hand with the population of agents building the knowledge representation, which applies to this environment.

In a given moment of time, the multi-agent system state is represented by the 2-tuple:

$$MAS \equiv < ES, AG >, \tag{1}$$

where: ES - environment state, AG - set of agent states.

The set of signals available in th environment is the source of information for every agent. Those signals are received by sensors, the agent is equipped with. The environment state is represented by the set of signals ($es \in ES$). The agent's internal structure and the way of co-operation between particular functional blocks has been defined a-priori at the stage of model design. The figure 1 illustrates agent's structure, which composes of below described elements.

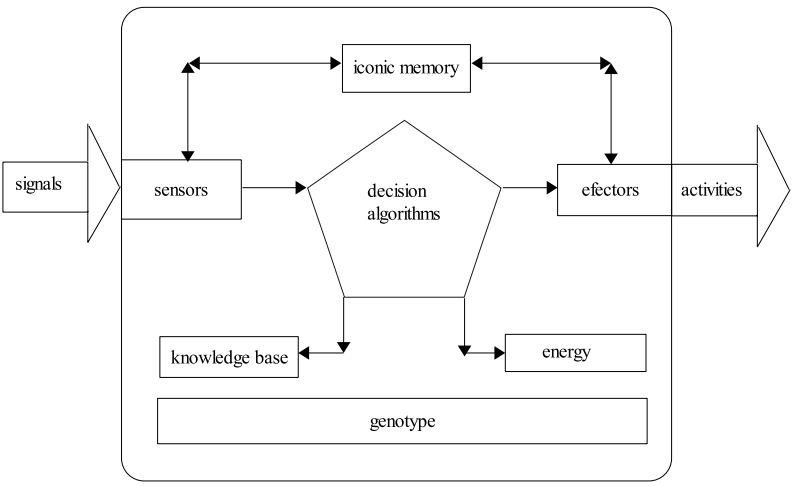


Fig. 1. Agent's structure chart.

Sensors – by means of sensors, agent observes the environment signals. Sensors are characterized among others by sensitivity that describes the minimal change of the environment signals that are being recorded.

Iconic memory – lets to store the successive observations and agent decisions. In case of incomplete environment observation, agent can be found in situation of lack of sufficient information necessary to make an appropriate decision, from the realized task's point of view. In order to make agent easier to take a decision in such situations, it is proposed to make use of the iconic memory in the decision process.

Effectors - with the help of effectors, the agent puts into practice the activities in the environment. Agent activities can be either single reactions

(actions) to received stimulus or sequences of actions leading to particular goals.

Knowledge base - consists of the set of decision rules, on the basis of which the agent realizes activities in the environment. Agent's knowledge base is composed of rules, every of which contains the observation pattern and assigned agent's decision.

Genotype - describes selected elements of the agent's structure, its sensory and morfological features. Examples of the agent's genotype features can be: sensors sensitivity or environment observation range.

Agent's energy – consumable and renewable resource that describe agent's usability in realization of the task.

Decision algorithms - make the agent's structure kernel, and make possible to take decisions and implementation of residual functions connected with agent's activity.

Agent's state in a given moment is represented by the 6-tuple:

$$ag \equiv < s, d, m, e, g, R > \tag{2}$$

where: s - observation vector, d – decision vector, m – iconic memory matrix, e - agent's energy, g - genotype vector, R - knowledge base.

Agent should provide correlation of observation vector with an adequate decision leading to accomplishing the task. Due to the limited volume of memory, the agent does not have the possibility to remember all values of observation vector, and store in the knowledge base only the patterns that represent the subsets of the observation space. Agent makes classification (fitting) of observation vector in a set of observation patterns. As a result the related agent's decision is determined, on the basis of which agent realizes actions in the environment.

Agent's decision process is divided into particular decisions that the agent undertakes in order to achieve the far-reaching target. In fact, agent's aim in a particular moment of time is to undertake the optimal decision from the rewarding function's point of view. The value of the rewarding function is usually known after finishing the total decision sequence in defined time horizon. In the moment of time (t) the agent observes the environment and its state defined by function s(t), then it selects action a(t). The environment responds in a way that after finishing the actions by all agents of the population(learning episode), it informs every agent about the reward granted. Learning consists in finding the optimal strategy $S \rightarrow A$ where S – set of the environment conditions, A – set of the agent's activities. In case we give consideration to the sequence of undertaking actions, the representation shall results as $S \times A \rightarrow A$. In the proposed model the description of agent's strategies is realized by the set of simple decision rules stored in it's knowledge base (R). Agent's knowledge acquisition is realized by using reinforcement learning paradigms and multi-agent evolution. Reinforcement learning leads to

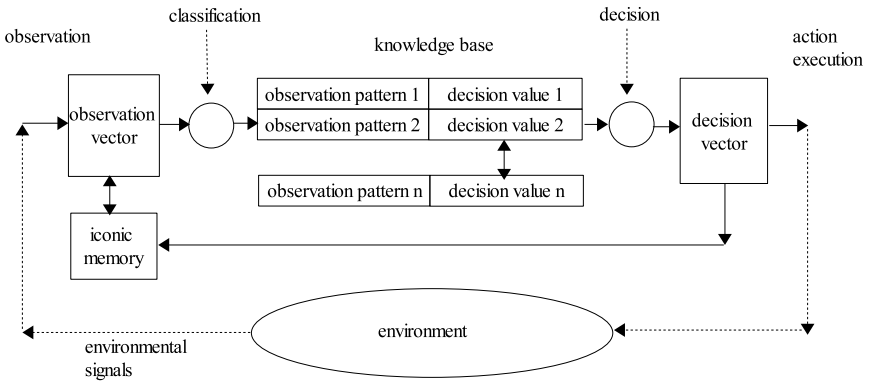


Fig. 2. Agent's activity chart.

creation and selection of decision rules that allow reaching the agent's goal. The task of multi-agent evolution is to improve the effectivnes of learning process through optimization of individual agent's features, which can apply among others to observation function (e.g. sensors sensitivity on the environment signals change) and the parameters of the decision process.

3 Knowledge acquisition process

During creation of a new agent, its ontogenetic knowledge does not exist (knowledge base is empty), agent starts obtaining experience in the environment. As a result of environment observation agent gets the observation vector. Agent's observation vector is also recorded temporarily in the iconic memory. Further, agent makes decision at random, which applies to the way of activity (exploration or exploatation).

1. In case it choses the exploration, the new rule in the knowledge base is generated. Its conditional part is the observation pattern (obtained on the basis of the observation vector). Decisional part is generated at random. New rule must be unique in the knowledge base. It is also the basis to make decision by the agent.
2. Second possibility is the choice of knowledge exploration. In this case, the obtained observation vector is the subject to matching in the knowledge base through making comparison with patterns included in conditional parts of already existing rules in the knowledge base.
   - If there is a lack of matching pattern, the complement of the knowledge base is undertaken. New rule is added. Its conditional part complies with present observation and its decisional part is generated at random. Similarly to the case of exploration, the rule being added must by unique in the knowledge base.

- In case of correct matching of more than one pattern in the knowledge base, decision is undertaken at random (the roulette wheel method has been used), the probability of choosing the particular rule depends on its performance weight(the adequate auxiliary)attribute.

3. The final effect of the decision process is the choice of the rule (activation) in the knowledge base and designation on its basis the decision vector of the agent. The selected rule is then marked by modification of the value of the usability indicator(auxiliary attribute). The attribute specifying the number of the rule's activation is also updated.
4. The agent executes actions as a result of the decision process. The execution of an action by the agent involves spending the energy resource. The decision vector can be stored in the iconic memory.
5. At the moment the agent achieves the success and gains the energetic reward, it modifies the performance weight for all used rules (that led to the success). In order to optimize the parameters of the knowledge base(size, searching speed), the rules with the low performance weight and rarely used (the values of related auxiliary attributes below certain threshold) are removed from the knowledge base.

The acquisition of phylogentetic knowledge is realized by the multi-agent evolution [3]. Genotype optimization takes place at the population level and takes advantages of the multi-agent evolution process. In the evolution model proposed in this paper we resign from applying selection method on the basis of global fitness function, that lead to the introduction of the order into the phenotypes space, which is not always adequate with the space structure of the given problem's solutions. Concerning the above, the energetic selection method observed in biology [4] has been used in our model. Every agent of the initial population is provided with a random allocation of the recource, called the energy. Having undertaken actions, the agents spend the energy, thus lose some part of energy possessed(energetic cost). The agents, whose energy diminishes below a certain energetic level, are to be eliminated. The agents who achieve successes during the realization of a task get the energetic reward, which allows them surviving in the environment. The agents who reaches certain energetic level undergo the reproduction. The agent's genotype is the subject of mutation. On its basis the new agent is being created. In this way, it is possible to gradually improve the efficiency of agents in the population.

4 The model realization

The co-operation of different knowledge acquisition methods within the multi-agent model can be used in solving of difficult decision problems. The realization of the model, which had been suggested in the paper, applied to the evasive maneuvers problem (pursue and evasion game) [2] in a two-dimensional space for a single pursuer (rocket) and a single evasive object (plane). In this case, towards the flying plane, the rocket is launched. It's goal is to hit the

plane. The rocket is led automatically (in deterministic way) at the target. The plane should be capable of evading the racket by changing the parameters of flight, i.e. the direction and speed. The plane is equipped with measuring devices that inform about the parameters of flight of the approaching rocket: the distance, mutual flight angle, rocket's speed. On the basis of the radar readings the agent controlling the plane should be capable of learning how to change the direction and speed of a plane to avoid the hit. Agent learning task has been divided into learning episodes. One of the initial conditions of each episode was the mutual location of the objects: the rocket and the plane. The episode ends when the plane was hit or managed to escape. The degree of difficulty has been established (by changing dynamical parameters of the objects) in such a way that the trivial strategies, consisting in continuous turning of the plane in one direction, did not allow to escape. The reference point for executed experiments was the case of using by the agent the random strategy of controlling the plane's escape (consisting in random changes of flight direction and speed). In this case, the number of escapes achieved, did not exceed 40 % of all learning episodes. Further, the calculation tests had been executed, during which, the agent's learning based on the suggested model had been performed. Obtained results have shown the effectiveness of the learning method being used. The learning effectiveness for the best population agent, let to achieve over 90 % ratio of the plane escapes from among all learning episodes. The optimization of the agent's knowledge base size also has been achieved (reduction of the number of decision rules).

Knowledge acquisition from the graphic pictures has been adopted as the second area of realization of the model proposed in this paper. The fingerprint lines recognition is often based on searching for characteristic features (called as "minutiae" points) or local irregularities in the course of the fingerprint lines [1]. It can be the lines' endings, the connections or other rarely occurring features at the picture of the typical points' class. The agent's task is to identify and locate the characteristic features within the picture. Agent is always assigned to the particular pixel on the picture and has the turn, in accordance with which it can move. On the basis of observations of surrounding pixels the agent calculates the vector of the picture's local features, performs its classification and undertakes decision about the movement or the change of its turn. In order to fulfill the realization of the detection task and features location, the agent should possess the knowledge about the picture that is represented by:

- Knowledge concerning the observed class of image, which make possible to identify the local characteristic features that are searched (for instance "minutiae" points),
- The strategy of relocating around the picture that allows localization of the features.

During the learning stage, agents were located on the example picture at random, on which the characteristic points "minutiae" were marked as well.

Agents had to learn to find the way from the place of actual location to the marked point. During the picture recognition stage, agents had to find the characteristic points minutiae within an unknown picture. Agents had been located again at random on an unknown picture of the fingerprints lines. They were moving on the picture using the knowledge (the "minutiae" features description and the strategy of searching) obtained during the previous learning stage. The task of finding the characteristic points within the picture of the fingerprint lines has been fully accomplished. The number of agents who achieved this success has exceeded 98 % of the population.

5 Summary

The main idea of the suggested solution was based on making use of the multiagent approach in order to fulfill the hybrid method of knowledge representation and acquisition. The reinforcement learning process of a single agent is being optimized using evolutionary method on the level of the entire population. Having based on the proposed idea, the model of the system has been elaborated, making easier to analyze the learning processes that occurs on different levels of abstraction. On the grounds of performed experiments, we can say that the suggested idea fulfills requirements applying to effectiveness of the obtained solutions and is characterized by great universality within the choosen range of applications. We forecast further improvements of the suggested knowledge acquisition model and experiments concerning other applications.

References

1. Bazen A.M., Gerez S.H., van Otterlo M., Poel M. (2001) A Reinforcement Learning Agent for Minutiae Extraction from Fingerprints, Proceedings of 13-th Belgian-Dutch Conference on Artificial Intelligence, Amsterdam.
2. Grefenstette J.J., C.Ramsey, A.Schultz (1990) Learning sequential decision rules using simulation models and competition, Machine Learning, Vol.5, Nr.4.
3. Kisiel-Dorohinicki M. (2000) Zastosowanie procesow ewolucyjnych w systemach wieloagentowych, PhD Dissertation, University of Mining and Metallurgy, Krakow, Poland.
4. Wierzchon S.T (2001) Sztuczne systemy immunologiczne. Teoria i zastosowania, Akademicka Oficyna Wydawnicza EXIT, Warszawa, Poland.
5. Wooldridge M. (2002) An Introduction to Multiagent Systems, John Wiley and Sons.

Restricted Linear Information Systems

Mikhail Ju. Moshkov

Institute of Computer Science, University of Silesia
39, Będzińska St., Sosnowiec, 41-200, Poland

Abstract. In the paper a class of infinite information systems is described. For decision tables over each such information system there exist low upper bounds on minimal complexity of decision trees and polynomial algorithms of decision tree optimization for various complexity measures.

1 Problems, Decision Tables and Decision Trees

An *information system* [11] is a pair (A, F) where A is a set and F is a set of functions from A to $\{0, 1\}$. Functions from F are called *attributes*.

The notion of a *problem* over this information system is defined as follows. Let G be a finite subset of F. Attributes from G divide the set A into regions in which values of these attributes are constant. These regions are enumerated such that different regions can have the same number. For a given element from the set A it is required to recognize the number of region to which this element belongs. The cardinality of the set G is called the *dimension* of this problem. Various problems of pattern recognition, discrete optimization and machine learning can be transformed into such form.

The considered problem can be represented in the form of a *decision table*. Columns of this table are labeled by attributes from G. Rows of the table correspond to regions. Each row is the tuple of values of attributes from G on elements from corresponding region. The considered row is labeled by the number of this region, which can be interpreted as the value of the decision attribute.

As algorithms for problem solving we will use decision trees with attributes from the set G. A *decision tree* is a tree with the root in which each terminal node is labeled by a number of region, each non-terminal node is labeled by an attribute from G, two edges start in each non-terminal node, and these edges are labeled by numbers 0 and 1 respectively.

Depth, *average depth* and *number of nodes* are considering usually as decision tree complexity measures. The depth of a decision tree is the maximal length of a path from the root to a terminal node. The average depth is defined in the natural way if for each region we know the probability of belonging of elements from A to this region.

2 Restricted Information Systems

We will study so-called restricted information systems. An information system (A, F) is called *restricted* if there exists natural r such that for any compatible on A system of equations

$$\{f_1(x) = b_1, \ldots, f_m(x) = b_m\} \tag{1}$$

where $f_1, \ldots, f_m \in F$ and $b_1, \ldots, b_m \in \{0, 1\}$ there exists an equivalent subsystem of this system with at most r equations. The system (1) is called *compatible* on A if it has a solution from A. Two systems of equations are called *equivalent* if they have the same set of solutions.

Evidently, each information system with finite set F is restricted. The interest to study of restricted information systems with infinite set F is determined by the following facts:

1. For a restricted information system in the worst case the minimal depth of decision tree solving a problem grows as logarithm on problem dimension. For an information system which is not restricted in the worst case the minimal depth of decision tree solving a problem grows as linear function on problem dimension [8].
2. For restricted and only for restricted information system minimal average depth of decision tree solving a problem is bounded from above by a function depending only on the entropy of probability distribution [1]. For restricted information system in the capacity of such function it is possible to take a linear function [9].
3. If a subset of the set A is fixed we obtain a *subproblem* of the considered problem. This subproblem is called *proper* if the considered subset is the set of solutions of an equation system of the kind (1) where $f_1, \ldots, f_m \in G$ and $b_1, \ldots, b_m \in \{0, 1\}$. For restricted and only for restricted information system the number of proper subproblems is bounded from above by a polynomial on problem dimension [10]. This fact allows to construct for restricted information systems polynomial algorithms for minimization of number of nodes [3], depth [10] and average depth [2] of decision trees for problems represented in the form of decision tables.

3 Linear Information Systems

Let P be the plane and l be a straight line (line in short) in the plane. This line divides the plane into two open half-planes H_1 and H_2 and the line l. Two attributes *correspond* to the line l. The first attribute takes value 0 on points from H_1, and value 1 on points from H_2 and l. The second one takes value 0 on points from H_2, and value 1 on points from H_1 and l. Denote by L the set of all attributes corresponding to lines in the plane. Information systems of the kind (P, F), where $F \subseteq L$, will be called *linear* information

systems. The aim of the paper is to describe all restricted linear information systems.

Let l be a line in the plane. Denote by $L(l)$ the set of all attributes corresponding to lines which are parallel to l. Let p be a point in the plane. Denote by $L(p)$ the set of all attributes corresponding to lines which pass through p. A set C of attributes from L will be called *clone* if $C \subseteq L(l)$ for some line l or $C \subseteq L(p)$ for some point p.

Theorem 1. *A linear information system (P, F) is restricted if and only if F is a union of finite number of clones.*

The proof of this theorem is based on Helly's theorem [5] and Ramsey's theorem [6]. It is too long to be considered in this paper.

4 Example

Let (P, F) be a linear information system where F is the set of attributes corresponding to straight lines each of which is parallel to a coordinate axis. From Theorem 1 it follows that (P, F) is a restricted linear information system.

Let W be a finite set of points from P which are colored into two colors. For a given point from W it is required to find the color of this point using attributes from F. It is not difficult to reformulate this problem as a subproblem of some problem over (P, F).

From results of [4] it follows that the problem of construction of a subset of F with minimal cardinality, attributes from which separate each pair of points from W with different colors, is NP-hard.

From results of [10] it follows that there exists a polynomial algorithm which constructs a decision tree with minimal depth using attributes from F only and solving the problem of color recognition for points from W.

5 Conclusion

In the paper the class of restricted linear information systems is described. Systems from this class are very interesting from the point of view of decision tree theory and can be used in various applications of rough set theory [7,11].

References

1. Chikalov, I.V. (1998) Bounds on average weighted depth of decision trees depending only on entropy. Proceedings of the Seventh International Conference on Information Processing and Management of Uncertainty in Knowledge-based Systems, Vol. 2. Paris, France, 1190–1194

2. Chikalov, I.V. (2000) Algorithm for constructing of decision trees with minimal average depth. Proceedings of the Eighth International Conference on Information Processing and Management of Uncertainty in Knowledge-based Systems, Vol. 1. Madrid, Spain, 376–379
3. Chikalov, I.V. (2000) Algorithm for constructing of decision trees with minimal number of nodes. Proceedings of the Second International Conference on Rough Sets and Current Trends in Computing. Banff, Canada 107–111
4. Chlebus, B.S., Nguyen, S.H. (1998) On finding optimal discretization for two attributes. Proceedings of the First International Conference on Rough Sets and Current Trends in Computing. Warsaw, Poland. Lecture Notes in Artificial Intelligence **1424**, Springer-Verlag, 537–544
5. Danzer, L., Grunbaum, B., Klee, V. (1963) Helly's theorem and its relatives. Proceedings of Symposia in Pure Mathematics, Vol. 7. American Mathematical Society, Providence, Rhode Island, 101–180
6. Hall, M. (1967) Combinatorial Theory. Blaisdell Publishing Company, Waltham (Massachusetts)
7. Komorowski, J., Pawlak, Z., Polkowski, L., Skowron, A. (1999) Rough sets. A tutorial. Rough-Fuzzy Hybridization: A New Trend in Decision-Making. Edited by S.K. Pal and A. Skowron, Springer-Verlag, Singapore, 3–98
8. Moshkov, M.Ju. (1994) Decision Trees. Theory and Applications. Nizhny Novgorod University Publishers, Nizhny Novgorod (in Russian)
9. Moshkov, M.Ju., Chikalov, I.V. (1997) Bounds on average depth of decision trees. Proceedings of the Fifth European Congress on Intelligent Techniques and Soft Computing. Aachen, Germany 226–230
10. Moshkov, M.Ju., Chikalov, I.V. (2000) On algorithm for constructing of decision trees with minimal depth. Fundamenta Informaticae **41**(3), 295–299
11. Pawlak, Z. (1991) Rough Sets – Theoretical Aspects of Reasoning about Data. Kluwer Academic Publishers, Dordrecht Boston London

The Concept of the Hierarchical Clustering Algorithms for Rules Based Systems

Agnieszka Nowak and Alicja Wakulicz-Deja

Institute of Computer Science, Silesian University, ul. Bedzinska 39, Sosnowiec, Poland

Abstract. This paper presents a conception of fast and useful inference process in knowledge based systems. The main known weakness is long and not smart process of looking for rules during the inference process. Basic inference algorithm, which is used by the rule interpreter, tries to fit the facts to rules in knowledge base. So it takes each rule and tries to execute it. As a result we receive the set of new facts, but it often contains redundant information unexpected for user. The main goal of our works is to discover the methods of inference process controlling, which allow us to obtain only necessary decision information. The main idea of them is to create rules partitions, which can drive inference process. That is why we try to use the hierarchical clustering to agglomerate the rules.

1 Introduction

Discovering groups in data is an important problem in many application areas. Research in the field of clustering has been extensive, and many different methods for grouping data have been developed. The main goals of every clustering method are to find, for the given set of objects, a set of clusters where objects within each cluster are similar and objects in different clusters are very dissimilar to each other. Each of the clustering methods, however, describes the data from one point of view. One group of widely used clustering techniques is the hierarchical clustering methods. These methods find partitions of objects such that each cluster contains at least one object and each object belongs to exactly one cluster, i.e. clusters are disjoint[3].
Let $O = \{x_1, x_2, ..., x_n\}$ be a set of objects. A clustering C of the object set O is a partition $\{c_1, c_2, ..., c_k\}$ where each cluster $c_i$is a subset of O so that $\bigcup_{i=1}^{k} c_i = O$ and $c_i \cap c_j = 0$ for $i \neq j$. The size of clustering C, i.e., the number of clusters is the clustering, is denoted as $\mid C \mid$. A cluster c_i that contains only one object is called a singleton cluster[1]. Instead of one single partition of the given objects, the hierarchical clustering (HC) methods construct a sequence of clusterings. Such a sequence of clusterings is often given as a clustering tree, also called a dendrogram. In such a tree, leaves represents the individual objects and internal nodes the clusters. There are two kind of HC techniques: agglomerative and divisive. The difference between these techniques is the direction in which they construct the clusterings. An agglomerative HC (AHC) algorithm starts from the situation where each

objects forms a cluster, i.e. we have n disjoint clusters. Then in each step the algorithm merges the two most similar clusters until there is only one cluster left. A divisive HC algorithm, starts with one big cluster containing all the objects. In each step it divides the most distinctive cluster into two smaller clusters and proceeds until there are n clusters, each of which contains just one object.
Let c_i and c_j be two clusters in a clustering C. An inter-cluster distance $d(c_i, c_j)$ between two singleton clusters $c_i = \{x_i\}$ and $c_j = \{x_j\}$ is defined as the distance between objects x_i and x_j, i.e. $d(c_i, c_j) = d_f(x_i, x_j)$, where d_f is a distance measure defined for the particular type of objects x_i and x_j. If at least one of the clusters c_i, c_j consists of two or more objects, the inter-cluster distance between two clusters c_i and c_j is a function F of the pairwise distances between objects when one of them is in the cluster c_i and the other in the cluster c_j, i.e., $d(c_i, c_j) = F\{(d_f(x_k, x_l) \mid x_k \in c_i \; and \; x_l \in c_j)\}$. The function F defining the inter-cluster distance can be chosen in different ways. There are: single linkage, complete linkage, average linkage method, unweighted pair-group method, weighted pair-group method, unweighted pair-group method with the centroid average, weighted pair-group method using the centroid average and sometimes there is used Ward's method[2,5]. Agglomerative algorithms start with each object being a separate cluster itself, and successively merge groups according to a distance measure. The clustering may stop when all objects are in a single group or at any other point the user wants. That is why we propose to stop clustering rules when we get a satisfactory groups. These will be the groups, where the similarity/distance measure is higher or equal to some threshold. We will consider this subject[1] in next section.

2 Comparison of the non-hierarchical and hierarchical method

We can consider a lot of the methods for grouping objects. But the most often used are: k-means method, k-medoids method and hierarchical clustering method. We will try to compare they, to choose the best to our task and we put our ideas in Table 1.

After this short comparison process, we can tell that the best chose is the AHC, but it requires another stop condition as we want to. The classical version tells that the algorithm ends when we get one group. But our goal is to get some groups, we don't know how many, but these groups need to come up to some threshold and they have to be as similar as it possible. The advantage of HC methods is that they allow the end user to choose from either many clusters or only a few. The non-hierarchical techniques in general create output information faster but require that the user make some decision about the number of clusters. Non-hierarchical techniques usually run multiple times start with some arbitrary or even random clustering and

Table 1. The comparison of the clustering methods (here t is # of iterations, k is # of clusters, n is # of objects, normally $k, t \ll n$).

	k-means	k-medoids	AHC
a	$O(tkn) \rightarrow O(kn)$	$O(k(k-n)^2)$	at least $O(n^2)$
b	often terminates at a local optimum, need to specify k, sensitive to noise and outliers	often terminates at a local optimum and need to specify k	needs a termination condition
c	simple structure, and smaller time complexity than other methods	better than k-means cause it tries to find the medoids, not so sensitive to outliers	doesn't require the number of clusters as an input, very effective, and very popular

a- time complexity, b- weakness, c- advantages.

then iteratively improving the clustering. Initial choices of which starting clusters can make an influence of created groups[6].

3 Conception of the algorithm

The analysis introduced in this paper was carried out on testing set of 24 rules. But our goal is to get 568 rules with different length of PC-Shell knowledge base file format [8]. We will then try to clustering them with AHC modified algorithm. We assume that it helps us with easy classification new cases and the selection of the final decision of attribute value. The groups of combined objects will be created and the retrieval process will be carried out only on special groups (the most popular groups). This process consists of 2 stages: (*i*) new analyzed object is compared with the representative vector of each group, and then (*ii*) the object is compared with all elements in groups with large degree of similarity/ smallest distance. We propose to change the common algorithm for AHC. [2,7] It is given as follows:

Algorithm 1.(Agglomerative hierarchical clustering for grouping rules)

Input: A set O of n objects and a matrix of pairwise distances between the objects.
Output: Clusterings $C_0, C_1, ..., C_{n-1}$ of the input set O
Method:
C_0= the trivial clustering of n objects in the input set O;
while ($d(ci, cj) >$ STOP[1]) do

[1]the stopping condition

find $c_i, c_j \in C_{k-1}$ so that the distance $d(c_i, c_j)$ is shortest;
$c_k = (c_{k-1}\{c_i, c_j\}) \cup \{c_i \cup c_j\}$;
compute the distance $d(c_i, c_j)\ \forall\ c_i, c_j \in C_k$;
output: $C_0, C_1, ..., C_{n-1}$;

Algorithm 1. gets as input a finite set O of n objects and a matrix of pairwaise distances between these objects. This means that executing the clustering algorithm is completely independent of how the distances between the objects were computed. The algorithm starts with a trivial clustering c_0 with n singleton clusters. At each iteration phase the algorithm searches those two clusters c_i , c_j that have the shortest distance in c_{k-1} and merges them. A new clustering C_k is formed by removing the two clusters and adding the new merged cluster, i.e. C_k is C_{k-1} with clusters c_i *and* c_j merged. The merging of clusters is continued until there is only one cluster left. The output of the algorithm is the sequence of clusterings $C_0, C_1, ..., C_{n-1}$. At each step the algorithm creates a centroid as the representative of created group-cluster. It is calculated as the average distance all objects till the given cluster.

3.1 Problems with the goal's achievement

There are always problems with various types of data in cluster analysis. We can represent N objects with p variables (attributes, measures) as continuous variables, binary variables, nominal and ordinal variables and variables of mixed types. In cluster analysis method we then may use the dissimilarity matrix $d(i, j)$, where on each position we remember the dissimilarity between objects i and j. Of course, this matrix is nonnegative, and the value close to zero means that two objects i and j are similar. The Euclidean distance is probably the most commonly chosen type of distance, it simply is the geometric distance in the multidimensional space $d(x, y) = \sqrt{\sum_{k=1}^{p}(x_{ki} - x_{kj})^2}$ where x_i, x_j are measured objects and p is the dimension of the space, which we consider. Sometimes, the application objective dictates that some attributes are more important to the definition of similarity than others. In these situations, we can modify some of the formulas for distance or similarity by weighting the contribution of each attribute: $d(x, y) = \sqrt{w_1(x_{i1} - x_{ji})^2 + ... + w_p(x_{ip} - x_{jp})^2}$. The second, very important problem, is the stopping condition. Normally, the agglomerative method stops when only one group is left, so there is nothing left to cluster. But, as this method, turns out to be quite slow, it is sometimes suggested that it should be stopped when either specified number of clusters remains or similarity value between two closest clusters in iteration falls below certain threshold. This approach often fails, as these two threshold have to be arbitrary, and it is hard to find a value that would suit different cases on input data. This will be our the most important considered task[4].

3.2 The guidelines

First, we normalize our objects. In this situation, the values of each attribute are numbered from 1 to n. We had to create the function to replace the textual structure with vector space model. After that, we got, 568 vectors. On each position of this vector there is the value of given attribute, if it is in conditional parts of given rule. If the rule doesn't use given attribute there is *zero* value on this position. As the result we get the data set containing the same structure, which can be easy to analyse. We have to create some structure to recording the results of clustering. It has to be the hierarchical structure. Our proposition is to create the tree of clustering results. It can be the simple ArrayList, where we can split the groups to smaller pieces, and for each node we have to create its representatives. It will be the centroid vector with the same length as the rules. Each groups can be also recorded as an array of rules, which belonging to the group. The data structure for saving a rule will be quite complicated, cause we have to remember the conclusion for each given rule and also the set of rule's conditions, for each group - the centroid vector. Now it will be easily to classify new cases. The rules interpreter implemented in inference engine will try to measure the similarity facts (new case) to each nodes in such tree. Only those, which will give the high similarity will be consider as important for inference process. It mean, that only allowed objects (rules) will be checked, and in take future advantages, it will helps to decreased the time of making decision by our system[7].

4 An example of rules clustering

For simple analysis of the AHC work we use Statistica 5.5. We take a set of 24 rules with 8 variables (conditional attributes), which are the first stage of classification. Each rules was replaced with 8 dimensional solid length vector. On each position of this vector there is the value of given attribute, if it is in conditional parts of given rule. If given rule doesnt use given attribute there is zero value on this position. In result we get the data set contains the same structured, which will be easy to analyse. We choose the single linkage method to create the new clusters, and we use the Euclidean distance measure[5].

The results of AHC is presented in Fig. 1 as the dendrogram.

The point is that Statistica can only find the groups, presents them at the plot and save the history of clustering in the file. It doesn't allow you to use the created tree to future tasks. There is no access to structure of founded clusters. That is the main reason for us to create the system, which will build the structure of clusters in the format possible to use in main part of our work - inference process. Besides the Statistica doesn't have a step-by-step work, that is why the way of clustering is presented only at the simple table or at the dendrogram. We can't even see how representatives of created groups

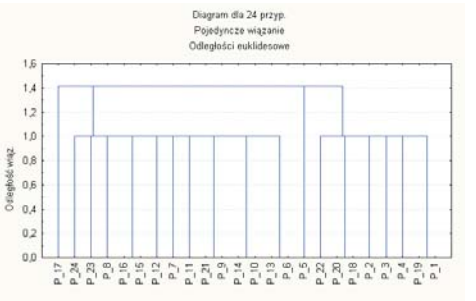


Fig. 1. The dendrogram created by AHC algorithm.

look like. So, it is needed to create the system, which will allow the user watching at the step by step work, tells the system the stopping condition and take a decision by user about the technique of form the centroids.

5 Conclusions

This paper presents a concept of some agglomerative rules clustering system, which we want to build. Now we can only present the common ideas of its work. Clustering is a common technique for the analysis and makes possible to analyse simply large data sets. We want to present in this paper a new approach to hierarchical clustering of those very large data sets. Our goal is to build a system which will clustering rules by the conditional part of them. We need to form the project of data structure to record a multidimensional grid data. The patterns are grouped into blocks and clustered with respect to the blocks by a topological neighbor search algorithm. We want to make the inference process in knowledge based systems faster and more useful[2].

References

1. Anderberg M.R. (1973) Cluster analysis for applications, New York, Academic Press
2. Everitt B.S. (1993) Cluster Analysis (3rd edition), Edward Arnold / Halsted Press, London
3. Kaufman L., Rousseeuw P.J. (1990) Finding Groups in Data: An Introduction to Cluster Analysis, John Wiley Sons, New York
4. Gatnar E. (1998) Symboliczne metody klasyfikacji danych,PWN, W-wa, [polish]
5. (1997) Statistica PL dla Windows (Tom III). Statystyki II, StatSoft, Kraków
6. http://www.thearling.com/text/dmtechniques/dmtechniques.htm
7. Nowak A., Wakulicz-Deja A. (2004) Effectiveness Comparison of Classification Rules Based on k-means Clustering and Salton's Method, Procederings of the International Workshop on Monitoring, Security and Rescue Techniques in Multiagent Systems, Płock, Poland
8. http://www.aitech.com.pl/pcshell.htm

Petri Net and Matrix Representation of Rule Knowledge Base for Verification Task

Roman Siminski

University of Silesia, Institute of Computer Science Poland, 41-200 Sosnowiec, Bedzinska 39, Phone (+48 32) 2 918 381 ext. 768, Fax (+48 32) 2 918 283, siminski@us.edu.pl

Abstract. The problem of verification of rule knowledge base covers the verification of dynamic properties, which reflect the processes occurring during inference. Process of detection of these anomalies requires modelling of dynamics of these processes. Suggested in previous papers the decision unit conception did not guarantee such a possibility. This paper gives attention to the analysis of possible use of Petri nets and incidence matrix as the way of representation of knowledge base. The paper presents the relation between Petri nets and decision units' nets and simultaneously points at the possible use of Petri nets to develop the properties of decision units.

1 Introduction

Rule base is still one of the most prevailing methods of knowledge base representation. Rules are intuitively simply way of knowledge modelling. The creation of rule knowledge bases only apparently seems to be an easy task. It is easy to create the prototype of knowledge base. Much more difficult is to develop the system capable of acting in the real world.

Adding new rules to knowledge base can result in formation of the anomalies, which in certain situations can make system malfunction [7]. Anomaly does not always mean error, but it always is the signal indicating that error can potentially occur. Several methods of anomalies detection in rule knowledge bases [7,8] have been developed. One of them is the method developed by the author of this paper — it is based on decision units' conceptions, which are the groups of rules linked by common literals in the conclusion part of the rules [9]. More specific, when presenting literals as attribute–value pairs, rules belonging to decision unit contain the same attribute in conclusion part of the rule.

This simple criterion of rules division into subsets let to naturally describe the problem of rule knowledge base verification. The division of anomalies into global and local has been made. For each of these categories the methods of their detection have been proposed. Thus, decision units are the medium of modularisation of rule knowledge base. Within given unit it is possible to implement the local verification, which does not have to be limited to just one method. Different approaches to anomalies detection [12] can be used interchangeably.

At the global level the dynamic anomalies, occurring during inference, are mainly detected. Because the decision unit can be considered as a *glass box*, a decision unit net is a global model of dependences occurring in knowledge base. It allows not only to carry the verification actions but to retrieve the model, hidden in, potentially numerous, set of rules. Decision units' conception has been used in `kbBuilder` system — the application that assists building and verification of rule knowledge bases [11].

The methods of local and global verification that were suggested during papers so far permit detection of basic anomalies. Nevertheless, decision units' conception lets to use different methods of verification. Very interesting verification method of rule knowledge bases is to make use of graph knowledge representation and incidence matrixes. The aim of this paper is the analysis of possible use of representation in the form of Petri nets and incidence matrixes describing the nets.

2 Rules and decision units versus Petri nets

Petri nets, which are the abstract tool of formal description of information flows, have been utilized in verification issues of knowledge bases. The net obtains the dynamic properties via the possibility of setting in motion the appropriate transitions, which is accompanied by tokens transfer. It is possible to present knowledge base as Petri nets [1–3]. Usually, the point is that the rules represent transitions and literals, occurring in rules, represent the places [5,6]. As you can see in the example below [4].

$r_1 : a \wedge b \rightarrow c$
$r_2 : c \wedge d \rightarrow e$
$r_3 : e \rightarrow f$

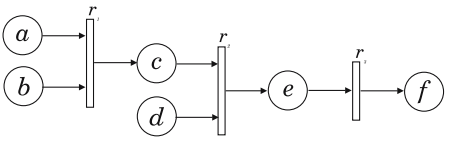


Fig. 1. First example of knowledge base – representation in the Petri net form

Such net can be represented in the form of incidence matrix, in which the rows represent the transitions and columns represent the places. For this example, the incidence matrix can assume the following shape:

$$\begin{vmatrix} -1 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 \end{vmatrix}$$

Table 1. The rules as labeled incidence matrix

rules	*a*	*b*	*c*	*d*	*e*	*f*
r_1	-1	-1	1	0	0	0
r_2	0	0	-1	-1	1	0
r_3	0	0	0	0	-1	1

or labeled form presented in Tab. 1.

Shown at the Fig. 1 representation of hypothetical rules in the form of Petri nets is identical to representation in the form of decision units net. Fig. 2, presents the hypothetical rules in the form of graphical convention established for decision units. This is the view of the *black box* .

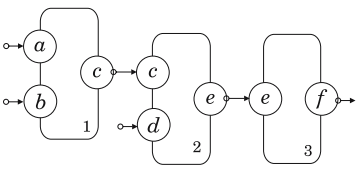


Fig. 2. First example of knowledge base — representation in the decision units net form

Between the two representations occurs the analogy. Transitions representing the rules correspond to decision units; places representing the literals correspond to input and output entries of decision units. However, it is only the particular case, because — in our example — each rule has different literal in conclusion and such literal has only one rule. In general case, decision units net in the view of *black box* shall was generalized representation of rules recorded in base and what follows, corresponding Petri net. It is presented by the example below. Let us consider the following set of rules:

$r_1 : a \wedge b \rightarrow c$
$r_2 : a \wedge g \rightarrow c$
$r_3 : c \wedge d \rightarrow e$
$r_4 : c \wedge g \rightarrow e$
$r_5 : e \rightarrow f$
$r_6 : g \rightarrow f$

It can be presented as Petri net and decision units net as shown at Fig. 3 and Fig. 4. Decision units can be also presented in the view of the glass box. Relationships occurring between input and output entries of units coincide with Petri net considering at one level of hierarchy. It is shown at Fig. 4.

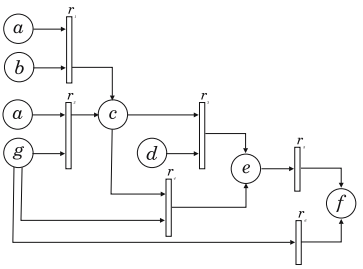


Fig. 3. Second example of knowledge base — representation in the Petri net form

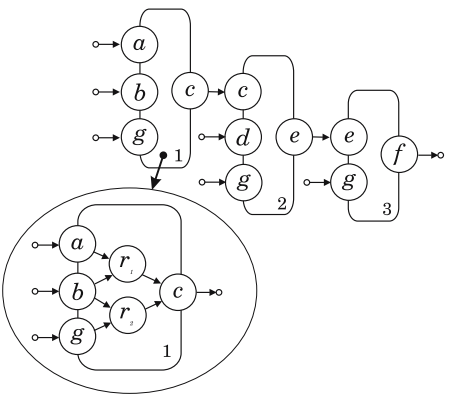


Fig. 4. Second example of knowledge base — representation in the decision units net form

3 Petri nets as extension of decision units properties

Thus, there exists the pertinence between knowledge base in the form of Petri net and representation in the form of decision units net. It is obviously nothing strange, since both representations make use of the same conception of connections representation between conditional literals and conclusion in the form of edges of directed graph, which nodes represent rules and their literals.

Such pertinence seems to be very interesting matter of further studies of knowledge base dynamical properties verification. In Petri nets the acknowledged methods of dynamic processes modelling are used. These methods can be also used during examination of decision units net, and more specific, during detection of anomalies occurring during inference.

Petri nets can be represented in the form of incidence matrix. Also, detection of the anomalies in rule knowledge base can take place taking advantage of the matrix transformation. Thus, matrix representation can be applied to decision units' nets. The anomalies detection can be also realized by making operations on incidence matrix. Because present paper is limited in its size, examples of verification realisation using Petri nets and corresponding

to them incidence matrix shall be omitted. They can be easily found in other papers [4,13].

One can have the impression that represented pertinence between decision units net and Petri net is nothing but smart manipulation and truly, the matter is of the same representation based at directed graph. And indeed, in a formal sense it is really so. Nevertheless both representations, we are discussing about, were created independently, and their primary uses and origins were different. A decision unit set is the representation that allows running the hierarchical verification of rule knowledge bases. At the global level — in view of the whole net and at the local level — considering every decision unit separately. The possibility of applying the verification techniques based on Petri nets is a very interesting supplement to methods of global verification suggested for decision units.

This is not the first case of analogy between decision units' conception and other methods. Global verification that let to reduce the inference graph, which was formed on the basis of decision units, seemed to be convergent with the conception of anomalies detection using ATMS [10] and the algorithms of local verification rely upon conceptions derived from decision tables.

The significant difference between classical, non-hierarchical Petri nets and decision units is just the hierarchy of the later. The use of Petri nets verification techniques for units considered as *glass box* directly corresponds to the concept of global verification. Petri net can be also a very interesting tool of presentation of verification course and making the knowledge engineer aware of dangers connected with given anomalies. Transition to the *glass box* view allows running further the verification using methods based on Petri nets. However, a detailed representation of the whole base is not necessary, but only a fragment chosen for verification.

4 Summary

The aim of this paper was to analyse the possibility of use of the Petri nets and incidence matrix for verification of rule knowledge bases represented in the form of decision units. What we managed to do was to show that there exists the pertinence between knowledge base in the form of Petri net and the same base in the form of decision units net. In particular cases, these are the same representations which differ in conception and notation.

In more general case the decision units net is the generalized representation of rule base, thus it is some kind of Petri net's abstraction. This allows further and consistent use of global and local verification conception described at the decision units' level. Transition from one change to the other does not require any change of the organization but only an extension of the scheme hidden inside decision units. What is even more interesting is that the visualisation of verification can take place using Petri net, which structurally does

not differ from decision units net which allows keeping cohesion of verification conception and presentation of its course and outcomes.

It's good to anticipate that the use of Petri nets together with decision units will allow to improve significantly the results of the verification process and to improve suggestively and efficiently their presentation in the next version of the `kbBuilder` system.

References

1. Agarwal R., Tanniru M., A Petri–net approach for verifying the integrity of production systems, International Journal of Man–Machine Studies, 36 (3), 1992.
2. Coenen F., An Advanced Binary Encoded Matrix Representation For Rulebase Verification, Journal of Knowledge Based Systems, vol. 8, no 4, 1995.
3. Coenen F., Rulebase Checking Using a Spatial Representation, 1998, available at: frans@uk.ac.liverpool.compsci
4. Coenen F. Bench–Capon T., Maintenance of Knowledge–Based Systems, Academic Press Inc., San Diego,1993.
5. Nazareth D.L., Investigating the applicability of petri nets for rule–based system verification, IEEE Transactions on Knowledge and Data Engineering, 4 (3), 1992.
6. Nazareth D.L., Kennedy M.H., Verification of Rule–Based Knowledge using Directed Graphs, Knowledge Acquisition , 3, 1991.
7. Preece A.D., Foundation and Application of Knowledge Base Verification, International, Journal of Intelligent Systems, 9, 1994.
8. Preece A.D., Methods for Verifying Expert System Knowledge Base, 1991, apreece@csd.abdn.ac.uk .
9. Siminski R., Wakulicz–Deja A., Verification of Rule Knowledge Bases Using Decision Units, Advances in Soft Computing, Intelligent Information Systems, Physica-Verlag, Springer Verlag Company, 2000.
10. Siminski R., O formalnym uzasadnieniu pewnego intuicyjnego algorytmu, [in polish:] Materiały Konferencji Systemy Wspomagania Decyzji, Zakopane, 3 5.12.2002.
11. Siminski R., Wakulicz–Deja A., kbBuilder — system wspomagania tworzenia i weryfikacji regulowych baz wiedzy, [in polish:] Materialy V-tej Konferencji Naukowej Inzynieria Wiedzy i Systemy Ekspertowe, Wroclaw, 13-15.06.2003.
12. Siminski R., Wakulicz–Deja A., Application of Decision Units in Knowledge Engineering, Lecture Notes in Artificial Intelligence, Springer–Verlag, Springer Verlag Company, 2004.
13. Siminski R., Wykrywanie anomalii w regulowych bazach wiedzy z wykorzystaniem macierzy incydencji,[in polish:] Statystyka i Informatyka w nauce i zarzadzaniu, Wydawnictwo WSZIM, Sosnowiec, 2004.
14. Szpyrka M, Ligeza A., Szmuc T., Badanie (weryfikacja) wlasnosci baz wiedzy z wykorzystaniem kolorowanych sieci Petriego, [in polish] Materialy IV Konferencji Inzynieria Wiedzy i Systemy Ekspertowe, Wroclaw 2000.

Artificial Neural Networks in Incomplete Data Sets Processing

Magdalena Tkacz

University of Silesia, Institute of Computer Science, Będzińska 39,
41-200 Sosnowiec, Poland

Abstract. This paper presents some results obtained in experiments with artificial neural networks trained with different learning algorithms in case of lack of some data in training and testing sets.

1 Introduction

Artificial neural networks are a well-known tool for different tasks, for example for prediction, classification or pattern recognition. Their topology and used learning algorithms are the subject of wide investigations and research works both in theory as in practice and also as a helpful tool in applications [1], [2], [3], [9]. However, in all kinds of experimental works there are serious problems with data: they should be completed for all samples for which an experiment was carried out. In case of loosing some parameters — regardless of a reason (human factor, loss of power supply, meter failure) — such data were considered insufficient, and with because of that — as lost, useless at all. In papers,there is an advice to eliminate such kinds of data from datasets or — if it is possible — to complete the data by performing additional experiments and/or measurements. Unfortunately, in many kinds of investigations samples are destroyed during the investigation or it is later impossible to reconstruct identical conditions as they were in time of experiment. Sometimes, incomplete set of data is a result of compiling data from different sources. The idea presented in that paper makes possible to use all data instead of excluding some of them. This paper shows how some neural network work (in dependency of different topologies and used type of learning algorithm) in processing an incomplete dataset.

2 Preparation of dataset

First of all, there was a necessity to prepare appropriate set of data [5], [6]. The complete set of data was obtained from information brochures about heat pumps that were provided by producers. Different producers give different heat pump parameters. So, after consolidation of data from brochures the data set looks like presented in Table 1. The entire dataset consisted of 31 objects with 7 parameters (a1 up to a7), respective.

Table 1. Data assembled from brochures (part of dataset — sample) [4]

	a1	a2	a3	a4	a5	a6	a7
Object 1		2		50		16,2	65
Object 2	21000		7800		4,5	1,9	5,5
Object 3				16	6	1,5	4,4

Such form of data presentation practically disqualified them from directly processing with computers. A researcher should complete the data — or eliminate objects described by incomplete set of parameters. It is possible when such kind of data make a small part of the entire dataset. It is worse when respectable — or all set (as presented in Table 1) of the entire dataset is composed from such incomplete data. In practice different methods for completing dataset are used: instead of empty places different kinds of certain numbers are fulfilled, e.g. some type of average is calculated from the rest of known parameters. It is not quite good method: average can widely differ from hypothetic parameter — more suitable in this condition and in context to other parameters describing certain object.

3 Proposed idea

When we want to analyze noised or incomplete image — in any way it have to be prepared in suitable form for computer processing. We can treat incomplete image as incomplete or noised dataset. Such situation can be thought to fit to dataset shown in Table 1. It is also considered as a special kind of "image" of certain situation — it is an ordered set of parameters describing the object of examination. Such set of parameters allow you to have a specific opinion about what object is, and what is its behavior in different situations, what are its features. Similarly, when you see an image as something visible (e.g a photo) you see its width and length, you see if it is color or black and white or sepia — there are the attributes. You also see what kind of objects it presents. There are questions: "Are artificial neural networks suitable tool for processing such kind of "image"? What happen when they will be used for processing such type of "image"?".

4 Linear neural network trained on incomplete dataset — results

First of all, blank places in dataset were fulfilled with one, small, nearby zero constant number. Next, the entire dataset was divided into three subsets: learning set, validation set and testing set. All subsets were separable. Data

in each set have been normalized. Parameters from one to six were the inputs of neural network, parameter seven was an output (predicted value). Only training and validation sets were presented to the network during training.

One of the examined networks [4] was a linear network with six inputs and one output. At first this network was trained with the backpropagation algorithm. Experiments were made with different values of both learning coefficient and momentum, but effects were similar to each other — no special influence of changing these parameters was noticed. Results (comparison between real data from testing set and network answer) obtained during verification with data from testing dataset was as presented in Figure 1.

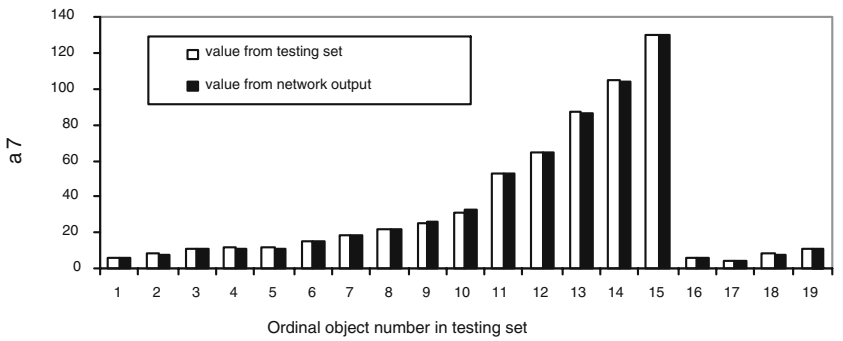


Fig. 1. Comparision of data in linear network trained with backpropagation algorithm [6]

In the next experiment [5] the same network was trained with conjugate gradient algorithm. Learning process was faster than the previous one, and results were similar to the network trained with backpropagation algorithm. Quite different results were obtained when the Quasi-Newton algorithm was used. Learning process was slower than in case of using the conjugate gradient algorithm, but faster than in case of backpropagation algorithm. Despite of relatively good learning process and relatively low error during training on validating set network did not pass final test — on testing set. Their answers were different, without any regularity or without correlation with original data. Results are shown in Figure 2.

The next examined network was the multilayer perceptron. The effects were similar to the previous once, and the best results were obtained when backpropagation algorithm was used [7].

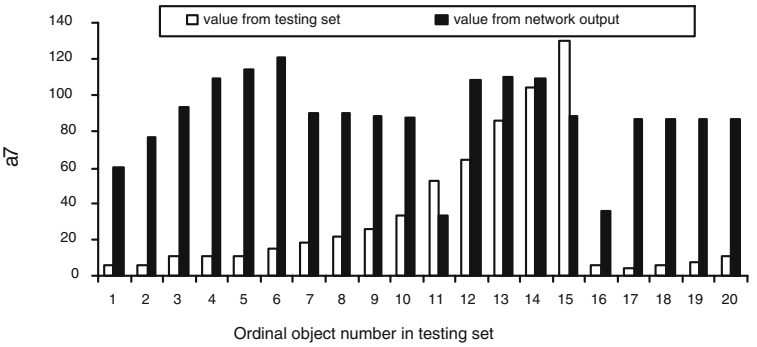


Fig. 2. Comparision of data in linear network trained with Quasi-Newton algorithm [6]

5 Multilayer perceptron trained on incomplete dataset — results

Previous examination was carried out on relatively simple and small dataset. Next experiment was made with more complicated and more numerous dataset — it was a set of parameters describing an geoenvironmental conditions of certain objects (materials from which the house was made, type and thickness of isolation, ground type, ground humidity, windiness, solarity etc). The entire dataset in this case consists of 113 objects with 34 parameters from which 29 was an input and up to 5 was an output (a few of them are showing there and marked as parameters b1, b2, b3). This dataset, like previous one presented in this paper, was incomplete. Usually it was caused by unknown certain parameters (e.g. thermal conductivity or certain type of ground) or with regard to impossibility of realization of some measurements. The task for neural network was training was to predict the power of heat pump and boiler for central heating.

At first, blank places in dataset were fulfilled with pseudorandom numbers. All pseudorandom numbers were from range [min_parameter_value; max_ parameter_value] (e.g. for parameter four (see Table 1) it was range [16; 50]). The influence of different ranges from which numbers can be generated was discussed in [6]. Next, the multilayer perceptron was trained with different learning algorithms. Coefficients within learning algorithms have been changing during experiments, in different tours of learning network. Finally, it seems that best solution is the multilayer perceptron trained with backpropagation algorithm. When used — there were relatively regular run of training process. Special influence of arbitrary chosen learning parameters (hints known from usual work of neural net remains the same) were not observed in examined cases. Because a number of possible combinations of these coefficients is very large, dependency cannot be definitely stated. Examplary

effects of answers of neural net trained on artificially completed, incomplete at the beginning datasets, were shown in Figures 3 to 5. Deatiled discussion can be found in [7], [8].

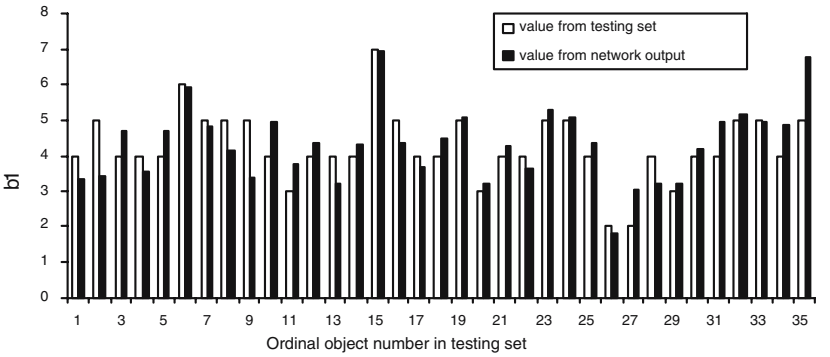


Fig. 3. Comparison of b1 parameters prediction by multilayer perceptron [8]

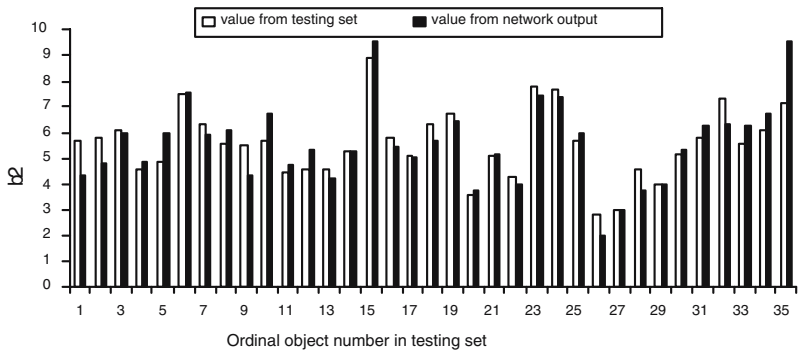


Fig. 4. Comparison of b2 parameters prediction by multilayer perceptron [8]

All results obtained with this method (insertion of pseudorandom values) are loaded with an error. On the base of my investigations [4], [5], [6], [7], [8] it can be concluded that such an error is from a few to a dozen percent in range. Such errors are acceptable, especially in modeling of complex and dynamic structures. At first, trends or possibly behavior of the entire structure than exact value of any certain parameter have to be known. Deatiled discussion can be found in [8].

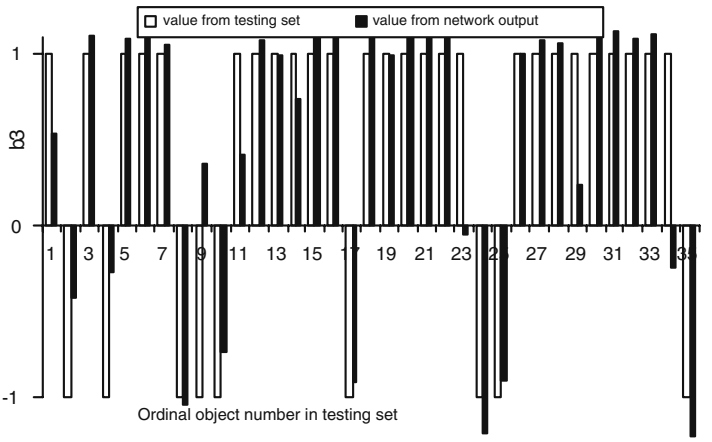


Fig. 5. Comparison of b3 parameters prediction by multilayer perceptron [8]

6 Summary

Basing on presented above and described [4], [5], [6], [7], [8] in experiments it can be stated that artificial neural networks can be used as a tool for processing the datasets with incomplete data included in them, without any special preparation or completing or filtering data. The only thing it should be done is to fulfill blanks in the dataset with some numbers. These numbers can be both a constant number as a pseudorandom number [5], [6], [8]. The only restriction is that it has to be generated from bounded range — not below minimal and not above maximum value of known parameter. In that way we can assume that definition of image as an ordered set of parameters describing the object of examination can be used for processing data with artificial neural networks. The effectiveness of this proposed idea and method needs to be verified by practitioners — as an application in different fields of sciences. If effectiveness of this method will be confirmed, then it can be a widely used method in most experimental researches. Some experiment then will be cheaper and less time consuming.

References

1. Osowski Stanisław (1996): Sieci neuronowe w ujęciu algorytmicznym. WNT, Warszawa 1996
2. Korbicz J., Obuchowicz A., Uciński D. (1994): Sztuczne sieci neuronowe. Podstawy i zastosowanie. Akademicka Oficyna Wydawnicza PLJ, Warszawa 1994.
3. Tadeusiewicz Ryszard (1993): Sieci neuronowe.Akademicka Oficyna Wydawnicza RM, Warszawa 1993
4. Tkacz Magdalena (1998): Przetwarzanie danych niepelnych. Red. G. Szpor. Katowice, SILGIS Center 1998

5. Tkacz Magdalena (1999): Processing an incomplete data using artificial neural networks. International Workshop Control and Information Technology, IWCIT'01. Ostrava, Technical University1999.
6. Tkacz Magdalena (2001): Processing an incomplete and random data using artificial neural networks. International Workshop Control and Information Technology, IWCIT'01. Ostrava, Technical University 2001
7. Tkacz Magdalena (2003): Dane niepełne i ich przetwarzanie przy pomocy sztucznych sieci neuronowych. Systemy wspomagania decyzji. Red. A. Wakulicz-Deja. Katowice, Uniwersytet Śląski 2003
8. Tkacz Magdalena (2004): Modelowanie warunków geośrodowiskowych przy użyciu metod sztucznej inteligencji na przykładzie hybrydowego systemu płytkiej geotermiki. Praca doktorska. Sosnowiec, Uniwersytet Śląski 2004. (Doctorate disertation: Geoenvironmental modelling with artificial intelligence methods in case of hybrid geothermal system.)
9. Żurada J., Barski M., Jędruch W. (1996) Sztuczne sieci neuronowe. PWN, Warszawa 1996

Intelligent Data Processing in Distributed Internet Applications

Beata Zielosko and Alicja Wakulicz-Deja

Institute of Computer Science, University of Silesia,
ul. Będzińska 39, Sosnowiec, Poland

Abstract. We discuses usage of elements of .Net platform — Web Service and XML to create distributed internet applications as data processing system. The main aim is to create retrieval system of hidden relations between data. The application uses elements of rough sets theory in order to get data, which could be used for further exploration. Data are sent to particular Web Services as XML format and could represent various domains e.g. medicine, pharmacy. The implementation intelligent techniques of data processing based on Web Services and XML standard, illustrates new possibilities for data processing in distributed environment. It gives scaling and possibility to adapt it in many specialized solutions.

1 Introduction

Along with modern technologies appears a lot of information. When we want to transform them very often traditional methods are not good enough. They might be too general and they need additional processing and transformation. Therefore we use advanced methods of data processing such as rough sets, genetic algorithms, neural nets or fuzzy sets. Rough sets have application e.g. in problems of excessing data, problems with correct classification or problems with retrieval hidden relations between data. Placing these methods in environment of distributed applications, based on .Net platform and XML Web Services, introduce new possibilities in usage of artificial intelligence methods in data processing systems.

2 Web Services and XML technology components

Together with development of internet W3C (World Wide Web Consortium), created universal and readable format of data - standard XML. It introduced the separation of data layer and presentation layer [1]. Therefore exchange of data is easy to do between applications and an independent platform, both on software or hardware. Not only XML is a markup language, defining data file, but also it defines methods of its transformation, presentation, queries, schemas and rules. Thus XML standard includes components of XML language [6]. The following table (Table 1) illustrates the most important components of XML language.

Table 1. The most important components of XML language

XML Language Extensibility	**Notification**	**Used in application**
CSS (Casscade Style Sheat)	Simple mechanism for adding style (e.g. fonts, colors, spacing) to Web documents	No
DTD (Document Type Declaration)	Contains or points to markup declarations that provide a grammar for a class of documents	No
XML Schema (Extensible Markup Language Schema)	Provide a means for defining the structure, content and semantics of XML documents	Yes
Xpath (Extensible Path)	An expression language used by XSLT to access or refer to parts of an XML document	Yes
XSLT (Extensible Stylesheet Language Transformation)	Transformation language provides elements that define rules for how one XML document is transformed into another XML document	Yes
DOM (Document Object Model)	Platform- and language-neutral interface that will allow programs and scripts to dynamically access and update the content, structure and style of documents. The document can be further processed and the results of that processing can be incorporated back into the presented page	Yes
SAX (Simple API for XML)	Protocol that most servlets and network-oriented programs will want to use to transmit and receive XML documents	Yes
SOAP (Simple Object Access Protocol)	Provides the framework by which application-specific information may be conveyed in an extensible manner	Yes
XQuery (XML Query Language)	XQuery operates on the abstract, logical structure of an XML document, rather than its surface syntax. This logical structure, known as the data model, is defined in the [XQuery 1.0 and XPath 2.0 Data Model] document	No
XQL(Extensible Query Language)	It is a general purpose query language, providing a single syntax that can be used for queries, addressing, and patterns. XQL is concise, simple, and powerful	No
WSDL (Web Service Description Language)	Is an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information. The operations and messages are described abstractly, and then bound to a concrete network protocol and message format to define an endpoint	Yes

Due to XML properties, I have chosen this language as a standard of data and standard of communication in the application placed on .Net platform. It gives such adventages as: hierarchical structure of documents, possibility to transmit data through any nets based on HTTP protocol and legibility of documents. This enables these documents to be apllied as sets of input data for another applications. XML and .Net are approved industrial standards [4].

3 Application

The main aim is to create retrieval system of hidden relations between data. The application in current form includes Web Service (see Fig.1) - which is one of Web Services in multi modular container and thin-client as Web browser. The application uses elements of rough sets theory in order to get data, which could be used for further exploration. Data are sent to particular Web Services as XML format and could represent various domains e.g. medicine, bilology, economics or pharmacy. It is possible to use these Web Services as independent components, by other applications. Obviously it must be compatible with WSDL [6]. Therefore these Web Services could be used for developing the existing application or to create a new application. HTTP and SOAP are standards protocols applied to exchange data between Web Services or between Web Services and clients [5].

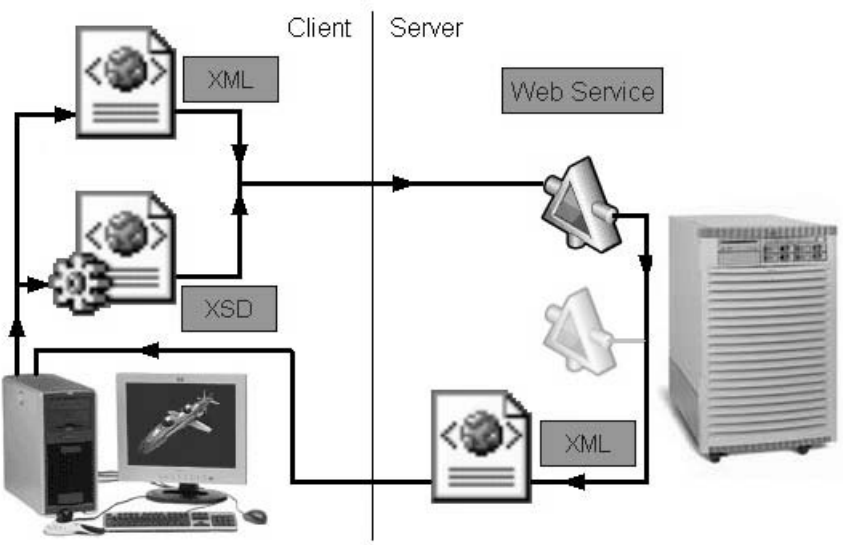


Fig. 1. Scheme of the Web Service

Data are send to the Web Service as XML file (see Fig.2). This file is a decision table:

$$S = (U, A \cup d), \tag{1}$$

where S - decision table (decision system), A - set of attributes called conditional attributes (conditions), $d \notin A$ -Ű decision attribute.

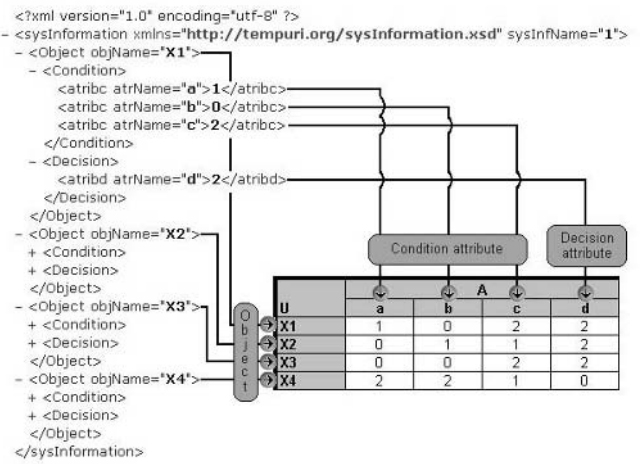


Fig. 2. Decision table in XML format

This file XML must be compatible with scheme defined in XSD file. This scheme assures correctness of XML file according to rules included in this file. XPathDocument class creates XPath queries, which have the access to particular elements of decision table. The size of decision table is dependent on number of objects and number of attributes. The same or indiscernible objects may be represented several times, or some of attributes may be superfluous. The Web Service first checks if decision table is consistent and if decision table has redundant objects. Next it computes equivalence classes:

$$IND(B) = (x, x') \in U^2 \mid \bigvee a \in B, a(x) = a(x'), \tag{2}$$

where IND(B) is called B - indiscernibility relation, A - set of attributes, B Ű- subset of attributes, $B \in A$. If $(x, x') \in IND(B)$ then objects x and x' are indiscernible for each other by attributes from B.

An equivalence relation induces a partitioning of the universe. These partitions can be used to build new subsets of the universe [2].

The Web Service uses XPath queries to fix subsets of attributes. They are sent as parameters to function, which generates equivalence classes. File XML is the result. It includes equivalent classes such as XPath queries. This XML file is a "log" of operations made on decision table (see Fig. 3).

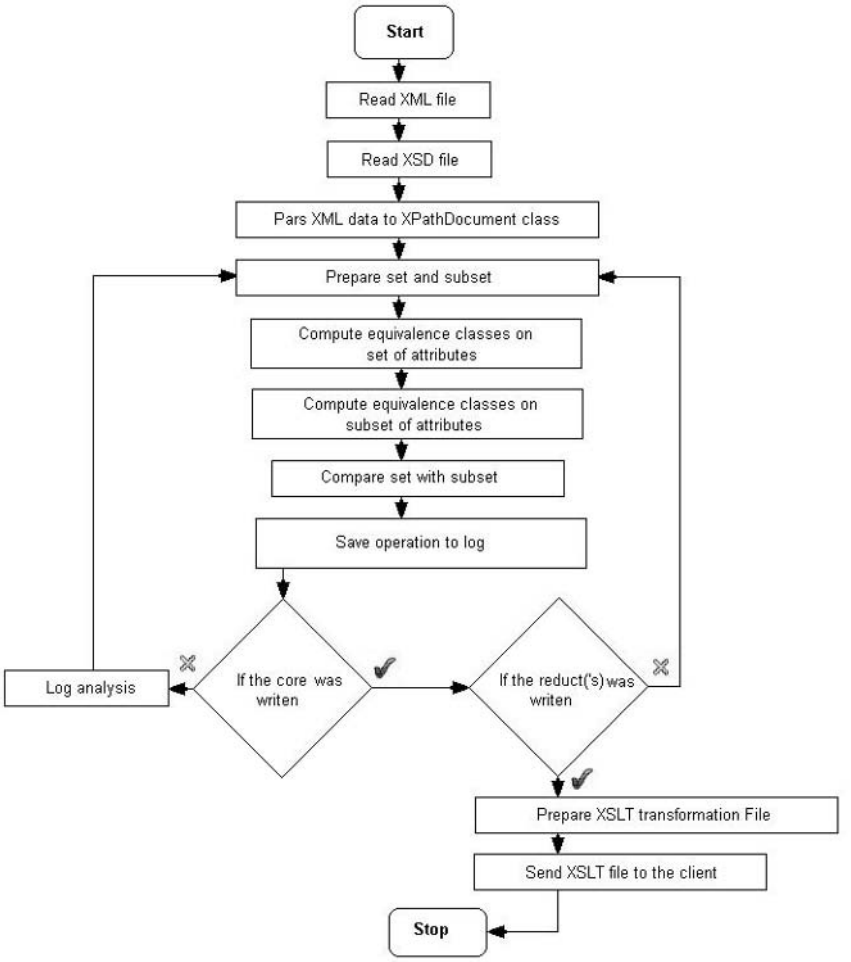


Fig. 3. Scheme of algorithm implemented in Web Service

To remove redundant attributes Web Service has implement algorithm to compute core and reducts. Attributes are redundant since their removal does not worsen the classification. Core is a set of indispensable attributes. Reduct is a minimal set of attributes that perseves the partitioning of the universe, and hence the ability to perform classifications as the whole attribute set A does. In other words, reduct of S is a minimal set of attributes $B \subseteq A$ such that IND(B) = IND(A), where S=(U, A), S - information system, U - universe, A - set of attributes, B - subset of attributes [3]. The usage of the concept of the core is twofold. First, it can be used as a basis for computation of all reducts, for the core included in every reduct, and its computation is straightforward. Secondly, the core can be interpreted as a set of the most

characteristic part of knowledge, which can not be eliminated when reducing the knowledge. Relation between core and reducts is [3]:

$$Core(B) = \cap Red(A), \tag{3}$$

where Red(A) - family of all reducts of A.

As the result of this algorithm is XSLT file. This file transform XML file and we get core and reducts as XML file (see Fig.3).

Web Service was implemented in PC with parameters:

- CPU Intel Celeron 1,2 GHz,
- 384 MB RAM,
- MS Windows 2000 Profesional system with IIS 6.0 and .NET Framework 1.1,
- application was developed in Visual Studio 2002, client was developed as ASP.NET.

Basic purposes was achievement. Algorithm operates correctly on data set with 500 objects and 10 attributes.
The following table (see Fig. 4) illustrates the average time needed for computing equivalence classes.

number of objects	time of computing equivalence classes [sec]
5	<1
10	<1
15	<1
20	<1
25	<1
30	<1
40	<1
50	1
60	1
70	1
80	1
90	1
100	2
150	5
200	10
250	14
300	26
350	37
400	55
450	83
504	103

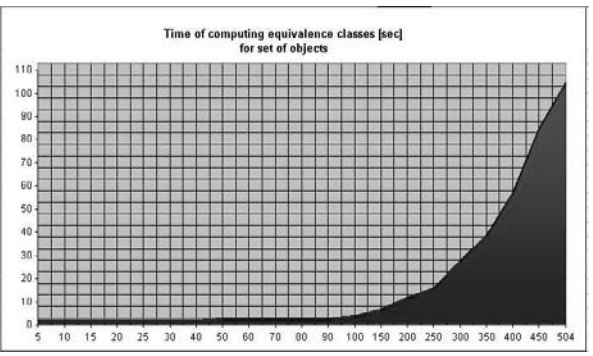


Fig. 4. Time of computing equivalence classes [sec] for set of objects

DOM (Document Object Model) as method to access to XML data file, was rejected, because it ocupies too much memory. DOM has good results for data set with maximum 150 objects and 10 atributes. If we have more objects, performance dicreases. Testing was done on fictional data set.
Next tasks are:

- to test Web Service, which compute core and reducts, with real data above 500 objects and 10 attributes,
- to create Web Service for set approximation,
- to imply SSL (Secure Sockets Layer) protocol as one of security aspects,
- to test performance Web Services with more clients,
- to opitmizate the code.

4 Summary

Web Service introduced in this article is an elementary module to prepare data for next analisys. It uses elements of rough sets theory and will develop. The implementation intelligent techniques of data processing based on Web Services from .Net platform, illustrates new possibilities for data processing in distributed environment. As clients we can imply each application, also on devices as PalmTop, PocketPC, or phone with GSM and WAP. Web Services could be implements in one server in LAN or WAN, or in cluster of servers. It gives scaling and possibility to adapt it in many specialized solutions.

References

1. Esposito D. (2002) Building Web Solutions with ASP .NET and ADO .NET, MS Press, Redmond.
2. Komorowski J, Pawlak Z, Polkowski L, Skowron A. Rough Sets: A Tutorial.
3. Pawlak Z. (1991) Rough Sets: Theoretical Aspects of Reasoning about Data, Kluwer, Dordrecht.
4. Panowicz Ł. (2004) Software 2.0: Darmowa platforma .Net, 1: 18-26.
5. Short Scott (2002) Building XML Web Services for the Microsoft .Net Platform, MS Press, Redmond.
6. http://www.w3.org

Part X

Invited Session: KDD and Facial Recognition

Skyline with Presorting: Theory and Optimizations

Jan Chomicki[1], Parke Godfrey[2], Jarek Gryz[2], and Dongming Liang[2]

[1] University at Buffalo, USA
[2] York University, Canada

Abstract. There has been interest recently in skyline queries, also called Pareto queries, on relational databases. Relational query languages do not support search for "best" tuples, beyond the `order by` statement. The proposed skyline operator allows one to query for best tuples with respect to any number of attributes as preferences. In this work, we explore what the skyline means, and why skyline queries are useful, particularly for expressing preference. We describe the theoretical aspects and possible optimizations of an efficiant algorithm for computing skyline queries presented in [6].

1 Introduction and Motivation

Often one would like to query a relational database in search of a "best" match, or tuples that best match one's preferences. Relational query languages provide only limited support for this: the `min` and `max` aggregation operators, which act over a single column; and the ability to *order* tuples with respect to their attribute values. In **SQL**, this is done with the `order by` clause. This is sufficient when one's preference is synonymous with the values of one of the attributes, but is far from sufficient when one's preferences are more complex, involving more of the attributes.

Consider a table of restaurant guide information, as in Figure 1. Column **S** stands for *service*, **F** for *food*, and **D** for *decor*. Each is scored from 1 to 30, with 30 as the best. We are interested in choosing a restaurant from the guide, and we are looking for a best choice, or best choices from which to choose. Ideally, we would like the choice to be the best for service, food, *and* decor, *and* be the lowest priced. However, there is no restaurant that is better than all others on every criterion individually, as is usually the case in real life, and in real data. No one restaurant "trumps" all others. For instance, Summer Moon is best on food, but Zakopane is best on service.

While there is no one best restaurant with respect to our criteria, we want at least to eliminate from consideration those restaurants which are worse on all criteria than some other. Thus, the Briar Patch BBQ should be eliminated because the Fenton & Pickle is better on all our criteria and is thus a better choice. The Brearton Grill is in turn eliminated because Zakopane is better than it on all criteria. Meanwhile the Fenton & Pickle is worse on every criterion than every other (remaining) restaurant, except on price, where it

restaurant	S	F	D	price
Summer Moon	21	25	19	47.50
Zakopane	24	20	21	56.00
Brearton Grill	15	18	20	62.00
Yamanote	22	22	17	51.50
Fenton & Pickle	16	14	10	17.50
Briar Patch BBQ	14	13	3	22.50

Fig. 1. Example restaurant guide table, GoodEats.

is the best. So it stays in consideration. This would result in the choices in Figure 2.

restaurant	S	F	D	price
Summer Moon	21	25	19	47.50
Zakopane	24	20	21	56.00
Yamanote	22	22	17	51.50
Fenton & Pickle	16	14	10	17.50

Fig. 2. Restaurants in the skyline.

In [3], a new relational operator is proposed which they name the *skyline operator*. They propose an extension to **SQL** with a `skyline of` clause as counterpart to this operator that would allow the easy expression of the restaurant query we imagined above. In [9] and elsewhere, this is called the *Pareto operator*. Indeed, the notion of Pareto optimality with respect to multiple parameters is equivalent to that of choosing the non-dominated tuples, designated as the skyline.

```
select ... from ... where ...
    group by ... having ...
    skyline of a[1] [min | max | diff], ..., a[n] [min | max | diff]
```

Fig. 3. A proposed skyline operator for SQL.

The `skyline of` clause is shown in Figure 3. Syntactically, it is similar to an `order by` clause. The columns $a_1, \ldots, a_n$ are the attributes that our preferences range over. They must be of domains that have a natural total ordering, as integers, floats, and dates. The directives `min` and `max` specify whether we prefer low or high values, respectively. The directive `diff` says that we are interested in retaining best choices with respect to every distinct value of that attribute. Let `max` be the default directive if none is stated. The

skyline query in Figure 4 over the table `GoodEats` in Figure 1 expresses what we had in mind above for choosing "best" restaurants, and would result in the answer set in Figure 2.

```
select * from GoodEats
    skyline of S max, F max, D max, price min
```

Fig. 4. Skyline query to choose restaurants.

Skyline queries are not outside the expressive power of current **SQL**. The query in Figure 5 shows how we can write an arbitrary skyline query in present **SQL**. The c_i's are attributes of `OurTable` that we are interested to retain in our query, but are not skyline criteria. The s_i are the attributes that are our skyline criteria to be maximized, and would appear in `skyline of` as s_i `max`. (Without loss of generality, let us only consider `max` and not `min`.) The d_i are the attributes that are the skyline criteria to *differ*, and would appear in `skyline of` as d_i `diff`.

```
select c_1, ..., c_k, s_1, ..., s_m, d_1, ..., d_n
      from OurTable
except
select D.c_1, ..., D.c_k, D.s_1, ..., D.s_m, D.d_1, ..., D.d_n
    from OurTable T, OurTable D
    where D.s_1 ≤ T.s_1 and ... D.s_m ≤ T.s_m and
        (D.s_1 < T.s_1 or ... D.s_m < T.s_m)and
        D.d_1 = T.d_1 and ... D.d_n = T.d_n
```

Fig. 5. SQL for generating the skyline set.

Certainly it would be cumbersome to need to write skyline-like queries in this way. The skyline clause is a useful syntactic addition. More important than ease of expression, however, is the expense of evaluation. The query in Figure 5 can be quite expensive. It involves a self-join over a table, and this join is a θ-join, not an equality-join. The self-join effectively computes the tuples that are trumped—or *dominated*—by other tuples. The tuples that remain, that were never trumped, are then the skyline tuples. It is known that the size of the skyline tends to be small, with certain provisos, with respect to the size of the table [7]. Thus, the intermediate result-set before the `except` can be enormous.

No current query optimizer would be able to do much with the query in Figure 5 to improve performance. If we want to support skyline queries, it is necessary to develop an efficient algorithm for computing skyline. And if we want the skyline operator as part of **SQL**, this algorithm must be

easy to integrate in relational query plans, be well-behaved in a relational context, work in all cases (without special provisions in place), and be easily accommodated by the query optimizer.

Recent years have brought new interest in expressing preference queries in the context of relational databases and the World Wide Web. Two competing approaches have emerged so far. In the first approach [1,8], preferences are expressed by means of preference (utility) functions. The second approach uses logical formulas [4,9] and, in particular, the `skyline` operator [3,12], described in the previous section. Skyline computation is similar to the *maximal vector problem* studied in [2,10,11]. These consider algorithmic solutions to the problem and address the issue of skyline size. None of these works addresses the problem in a database context, however. In [7], we address the question of skyline query cardinality more concretely.

In this paper, we explore what the skyline means, and why skyline queries are useful, particularly for expressing preference. We describe a well-behaved, efficient algorithm for computing skyline queries. Our algorithm improves on exisiting approaches in efficiency, pipelinabilty of output (of the skyline tuples), stability of run-time performance, and being applicable in any context.

2 Skyline versus Ranking

The skyline of a relation in essence represents the best tuples of the relation, the Pareto optimal "solutions", with respect to the skyline criteria. Another way to find "best" tuples is to score each tuple with respect to one's preferences, and then choose those tuples with the best score (ranking). The latter could be done efficiently in a relational setting. In one table scan, one can score the tuples *and* collect the best scoring tuples.

How is skyline related to ranking then? It is known that the skyline represents the closure over the maximum scoring tuples of the relation with respect to all *monotone scoring functions*. For example, in choosing a restaurant as in the example in Section 1, say that one values service quality twice as much as food quality, and food quality twice as much as decor, those restaurants that are best with respect to this "weighting" will appear in the skyline. Furthermore, the skyline is the least-upper-bound closure over the maximums of the monotone scoring functions [3].

This means that the skyline can be used instead of ranking, or it can be used in conjunction with ranking. First, since the best tuples with respect to any (monotone) scoring are in the skyline, one only needs effectively to query the skyline with one's preference queries, and not the original table itself. The skyline is (usually) significantly smaller than the table itself [7], so this would be much more efficient if one had many preference queries to try over the same dataset. Second, as defining one's preferences in a preference query can be quite difficult, while expressing a skyline query is relatively easy, users may find skyline queries beneficial. The skyline over-answers with respect to

the users' intent in a way, since it includes the best tuples with respect to *any* preferences. So there will be some choices (tuples) among the skyline that are not of interest to the user. However, every best choice with respect to the user's implicit preferences shows up too.

While in [3], they observe this relation of skyline with monotone scoring functions, they did not offer proof nor did they discuss *linear scoring functions*, to which much work restricts focus. Let us investigate this more closely, and more formally, then, for the following reasons:

- to relate skyline to preference queries, and to illustrate that expressing preferences by scoring is more difficult than one might initially expect;
- to rectify some common misconceptions regarding scoring for the purposes of preference queries, and regarding the claim for skyline; and
- to demonstrate a useful property of monotone scoring that we can exploit for an efficient algorithm to compute the skyline.

Let attributes $a_1, \ldots, a_k$ of schema R be the skyline criteria, without loss of generality, with respect to "max". Let the domains of the a_i's be real, without loss of generality. Let $\mathbf{R}$ be a relation of schema R, and so represents a given instance.

Definition 1. Define a *monotone scoring function* S with respect to R as a function that takes as its input domain tuples of R, and maps them onto the range of reals. S is composed of k monotone increasing functions, $f_1, \ldots, f_k$. For any tuple $t \in \mathbf{R}$, $S(t) = \sum_{i=1}^{k} f_i(t[a_i])$.

Lemma 1. *Any tuple that has the best score over* $\mathbf{R}$ *with respect to any monotone scoring function* S *with respect to* R *must be in the skyline.*[1]

It is more difficult to show that every tuple of the skyline is the best score of some monotone scoring. Most restrict attention to linear weightings when considering scoring, though, so let us consider this first.

Definition 2. Define a *positive, linear scoring function*, W, as any function over a table $\mathbf{R}$'s tuples of the form $W(t) = \sum_{i=1}^{k} w_i t[a_i]$, in which the w_i's are positive, real constants.

As we insist that the w_i's are positive, the class of the positive, linear scoring functions is a proper sub-class of the monotone scoring functions. Commonly in preference query work, as in [8], the focus is restricted to linear scoring. It is not true, however, that every skyline tuple is the best with respect to some positive, linear scoring.

Theorem 1. *It is possible for a skyline tuple to exist on* $\mathbf{R}$ *such that, for every positive, linear scoring function, the tuple does not have the maximum score with respect to the function over table* $\mathbf{R}$.

[1] Proofs of all lemmas and theorems can be found in [5].

Consider $\mathbf{R} = ((4,1),(2,2),(1,4))$. All three tuples are in the skyline (`skyline` of a_1, a_2). Linear scorings that choose (4,1) and (1,4) are obvious, but there is no positive, linear scoring that scores (2,2) best. Note that (2,2) is an interesting choice. Tuples (4,1) and (1,4) represent in a way outliers. They make the skyline because each has an attribute with an extrema value. Whereas (2,2) represents a balance between the attributes (and hence, preferences). For example, if we are conducting a house hunt, a_1 may represent the number of bathrooms, and a_2, the number of bedrooms. Neither a house with four bathrooms and one bedroom, nor one with one bathroom and four bedrooms, seem very appealing, whereas a 2bth/2bdrm house might.

Theorem 2. *The skyline contains all, and only, tuples yielding maximum values of monotone scoring functions.*

While there exists a monotone scoring function that chooses—assigns the highest score to—any given skyline tuple, it does not mean anyone would ever find this function. In particular, this is because, in many cases, any such function is a contrivance based upon that skyline's values. The user is searching for "best" tuples and has not seen them yet. Thus, it is unlikely anyone would discover a tuple like (2,2) above with any preference query. Yet, the 2bth/2bdrm house might be exactly what we wanted.

For the algorithm for skyline computation we are to develop, we can exploit our observations on the monotone scoring functions. Let us define the *dominance relation*, "$\preceq$", as follows: for tuples any $r, t \in \mathbf{R}$, $r \preceq t$ *iff* $r[a_i] \leq t[a_i]$, for all $i \in 1, \ldots, k$. Further define that $r \prec t$ *iff* $r \preceq t$ *and* $r[a_i] < t[a_i]$, for some $i \in 1, \ldots, k$.

Theorem 3. *Any total order of the tuples of* $\mathbf{R}$ *with respect to* any *monotone scoring function (ordered from highest to lowest score) is a topological sort with respect to the skyline dominance partial relation ("$\preceq$").*

Consider the total ordering on $\mathbf{R}$ provided by the basic **SQL** `order by` as in the query in Figure 6. This total order is a topological sort with respect to dominance.

```
select * from R
       order by a_1 desc, ..., a_k desc;
```

Fig. 6. An `order by` query that produces a total monotone order.

The following proposition is fairly obvious and it is used to build a better skyline algorithm.

Theorem 4. *Any nested sort of* $\mathbf{R}$ *over the skyline attributes (sorting in descending order on each), as in the query in Figure 6, is a topological sort with respect to the dominance partial order.*

As we read the tuples output by the query in Figure 6 one by one, it is only possible that the current tuple is dominated by one of the tuples that came before it (if, in fact, it is dominated). It is impossible that the current tuple is dominated by any tuple to follow it in the stream. Thus, the very first tuple must belong to the skyline; no tuple precedes it. The second tuple might be dominated, but only by first tuple, if at all. And so forth.

The last observation provides us the basis for an algorithm to compute skyline (the details of the algorithm, called **SFS**, are presented in [6]). First, we sort our table as with the query in Figure 6. In a relational engine, an external sort routine can be called for this. Buffer pool space is then allocated as a *window* in which skyline tuples are to be placed as found. A cursor pass over the sorted tuples is then commenced. The current tuple is checked against the tuples cached in the window. If the current tuple is dominated by any of the window tuples, it is safe to discard it. It cannot be a skyline tuple. (We have established that the current tuple cannot dominate any of the tuples in the window.) Otherwise, the current tuple is incomparable with each of the window tuples. Thus, it is a skyline tuple itself. Note that it was sufficient that we compared the current tuple with just the window tuples, and not all tuples that preceded it. This is because if any preceding tuples were discarded, it can only be because another tuple already in the window dominated it. Since dominance is transitive, then comparing against the window tuples is sufficient. In the case that the current tuple was not dominated, if there is space left in the window, it is added to the window. Note that we can also place the tuple on the output stream simultaneously, as we know that it is skyline. The algorithm fetches the next tuple from the stream and repeats.

3 Optimizations

Reduction Factor

A key to efficiency for any skyline algorithm is to eliminate tuples that are not skyline as quickly as possible. In the ideal, every eliminated tuple would only be involved in a single comparison, which shows it to be dominated. In the worst case, a tuple that is eventually eliminated is compared against every other tuple with which it is incomparable (with respect to dominance) before it is finally compared against a tuple that dominates it. In cases that **SFS** is destined to make multiple passes, how large the run of the second pass will be depends on how efficient the window was during the first pass at eliminating tuples.

For **SFS** only skyline tuples are kept in the window. One might think on first glance that any skyline tuple ought to be good at eliminating other tuples, that it will likely dominate many others in the table. This is not necessarily true, however. Recall the definition of a skyline tuple: a tuple that is *not* dominated by any other. So while some skyline tuples are great dominators, there are possibly others that dominate *no* other tuples.

Let us formalize this some, for sake of discussion. Define a function over the domain of tuples in **R** with respect to **R** called the *dominance number*, *dn*. This function maps a tuple to the number of tuples in **R** that it properly dominates ("$\prec$"). So, given that **R** has n tuples, $dn(t)$ can range from 0 to $n-1$. If tuple t is in the window for the complete first pass, at least $dn(t)$ tuples will be eliminated. Of course, *dn*'s are not additive: window tuples will dominate some of the same tuples in common. However, this provides us with a good heuristic: We want to maximize the cumulative *dn* of the tuples in the window. This will tend to maximize the algorithm's reduction factor.

Once the window is filled on a pass for **SFS**, the cumulative *dn* is fixed for the rest of the pass. Our only available strategy is to fill the window initially with tuples with high *dn*'s. This is completely dependent upon the sort order of the tuples established before we commence the filtering passes. Let us analyze what happens currently. We employ a sort as with the query in Figure 6, a nested sort over the skyline attributes. The very first tuple t_1 (which must be skyline) has the maximum a_1 value with respect to **R**. Say that $t_1[a_1] = 100$. Then $t_1[a_2]$ is the maximum with respect to all tuples in **R** that have $a_1 = 100$. This is probably high. And so forth for $a_3, ..., a_k$. Thus, t_1 with high probability has a high *dn*. Now consider t_i such that $t_i[a_1] = 100$, but $t_{i+1}[a_1] < 100$. So t_i is the last of the "$a_1 = 100$" group. Its a_2 value is the *lowest* of the "$a_1 = 100$" group, and so forth. With high probability, t_i's *dn* is low. However, if t_i is skyline (and it well could be), it is added to the window.

So **SFS** using a nested sort for its input tends to flood the window with skyline tuples with low *dn*'s, on average, which is the opposite of what we want. In Section 2, we observed that we can use *any* monotone scoring function for sorting as input to **SFS**. It might be tempting, if we could know tuples' *dn*'s, to sort on *dn*. The *dn* function is, of course, monotone with respect to dominance. However, it would be prohibitively expensive to calculate tuples' *dn*'s. Next best then would be to approximate the *dn*'s, which we can do.

Instead of a tuple's *dn*, we can estimate the probability that a given tuple dominates an arbitrary tuple. For this, we need a model of our data. Let us make the following assumptions. First, each skyline attribute's domain is over the reals between 0 and 1, non-inclusive. Second, the values of an attribute in **R** are uniformally distributed. Lastly, the values of the skyline attributes over the tuples of **R** are pair-wise independent. So given a tuple t and a randomly chosen $r \in \mathbf{R}$, what is the probability that $t[a_i] > r[a_i]$? It is the value $t[a_i]$ itself, due to our uniform distribution assumption (and due to that $t[a_i]$ is normalized between 0 and 1). Then the probability that $r \prec t$, given t is $\prod_{i=1}^{k} t[a_i]$ by our independence assumption. We can compute this for each tuple just from the tuple itself. Is this probability a monotone scoring function? It is easy to show that it is monotone. However, it is not formally a monotone scoring function as we defined this in Section 2; the definition only allowed

addition of the monotone functions applied over the skyline attributes. Define the monotone scoring function E then as $E(t) = \sum_{i=1}^{k} \ln(t[a_i] + 1)$. This clearly results in the same order as ordering by the probability. Interestingly, this is an *entropy* measure, so let us call this monotone scoring function E *entropy scoring*.

Our first assumption can always be met by normalizing the data. Relational systems usually keep statistics on tables, so it should be possible to do this without accessing the data. The second assumption of uniform distribution of values is often wrong. However, we are not interested in the actual dominance probability of tuples, but in a relative ordering with respect to that probability. Other distributions would not effect this relative ordering much, so E would remain a good ordering heuristic in these cases. The last assumption of independence too is likely to be wrong. Even in cases where independence is badly violated, E should remain a good heuristic, as again, the relative ordering would not be greatly effected. Regardless of the assumptions, E is always a monotone scoring function over **R**, and we can always safely use it with **SFS**.

Projection

For **SFS**, a tuple is added to the window only if it is in the skyline. Therefore, the tuple at the same time can be pushed to the output. So it is not necessary to keep the actual tuple in the window. All we need is that we can check subsequent tuples for whether they are dominated by this tuple. For this, we only need the tuple's skyline attributes. Real data will have many attributes in addition to the attributes we are using as skyline criteria. Also, attributes suitable as skyline conditions are comparables, as integer, float, and date. These tend to be small, storage-wise. A tuple's other attributes will likely include character data and be relatively large, storage-wise. So projecting out the non-skyline attributes of tuples when we add them to the window can be a great benefit. Significantly more skyline tuples will fit into the same size window. Likewise, there is no need to ever keep duplicate (projected) tuples in the window. So we can do duplicate elimination, which also makes better use of the window space.

Dimensional Reduction

Another optimization available to **SFS** is due again to the fact that we first sort the table. Recall the nested sort that results from the query in Figure 6. Now consider the table that results from the query in Figure 7. It has precisely the same skyline as table **R**. We choose the maximum a_k for each "$a_1, \ldots, a_{k-1}$" group. Clearly, any tuple in the group but with a non-maximum a_k cannot belong to the skyline. Of course, we can only apply this reduction once. (Implemented internally, other attributes of **R** besides the a_i's could be preserved during the "`group by`" computation.)

This optimization is useful in cases when the number of distinct values for each of the attributes $a_1, \ldots, a_{k-1}$ appearing in **R** is small, so that the number of groups is much smaller than the number of tuples. If one attribute

```
select a_1, ..., a_{k-1}, max(a_k) as a_k from R
    group by a_1, ..., a_{k-1};
    order by a_1 desc, ..., a_{k-1} desc;
```

Fig. 7. An order by query that produces a total monotone order.

has many distinct values, we can make this one our "a_k". In such a case, we are applying **SFS** to the result of the query in Figure 7, which can be a much smaller input table.

4 Conclusions

We believe that the skyline operator offers a good start to providing the functionality of preference queries in relational databases, and would be easy for users to employ. We believe that our **SFS** algorithm for skyline offers a good start to incorporating the skyline operator into relational engines, and hence, into the relational repertoire, effectively and efficiently.

References

1. R. Agrawal and E. L. Wimmers. A framework for expressing and combining preferences. In *Proceedings of SIGMOD*, pages 297–306, 2000.
2. J. L. Bentley, H. T. Kung, M. Schkolnick, and C. D. Thompson. On the average number of maxima in a set of vectors and applications. *JACM*, 25(4):536–543, 1978.
3. S. Börzsönyi, D. Kossmann, and K. Stocker. The skyline operator. In *Proceedings of ICDE*, pages 421–430, 2001.
4. J. Chomicki. Querying with intrinsic preferences. In *Proceedings of EDBT*, 2002.
5. J. Chomicki, P. Godfrey, J. Gryz, and D. Liang. Skyline with presorting. Technical Report CS-2002-04, Computer Science, York University, Toronto, Ontario, Canada, Oct. 2002.
6. J. Chomicki, P. Godfrey, J. Gryz, and D. Liang. Skyline with presorting. In *Proceedings ICDE*, pages 717–719, 2003.
7. P. Godfrey. Cardinality estimation of skyline queries: Harmonics in data. In *Proceedings of FoIKS Conference*, pages 78–97, 2002.
8. V. Hristidis, N. Koudas, and Y. Papakonstantinou. PREFER: A system for the efficient execution of multi-parametric ranked queries. In *Proceedings of SIGMOD*, pages 259–270, 2001.
9. W. Kiessling. Foundations of preferences in database systems. In *Proceedings of the 28*th *VLDB*, Aug. 2002.
10. H. T. Kung, F. Luccio, and F. P. Preparata. On finding the maxima of a set of vectors. *JACM*, 22(4):469–476, 1975.
11. F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction.* Springer-Verlag, 1985.
12. K.-L. Tan, P.-K. Eng, and B. C. Ooi. Efficient progressive skyline computation. In *Proceedings of VLDB*, pages 301–310, 2001.

Faster Clustering with DBSCAN

Marzena Kryszkiewicz and Łukasz Skonieczny

Institute of Computer Science, Warsaw University of Technology, Nowowiejska 15/19, 00-665 Warsaw, Poland

Abstract. Grouping data into meaningful clusters belongs to important tasks in the area of artificial intelligence and data mining. DBSCAN is recognized as a high quality scalable algorithm for clustering data. It enables determination of clusters of any shape and identification of noise data. In this paper, we propose a method improving the performance of DBSCAN. The usefulness of the method is verified experimentally both for indexed and non-indexed data.

1 Introduction

Grouping data into meaningful clusters belongs to important tasks in the area of artificial intelligence and data mining. Clustering can be perceived as unsupervised classification of data. A number of clustering algorithms were offered in the literature. Usually, different clustering algorithms group data differently. Some of the algorithms are capable to discover proper clustering of data only when the number of the clusters is known. Other algorithms are capable to discover clusters only of particular shapes. There are algorithms that are unable to identify noise data. The DBSCAN algorithm (Density-Based Spatial Clustering of Applications with Noise) [5] is recognized as a high quality scalable algorithm for clustering, which is free of these limitations. It belongs to the class of density-based algorithms. Other distinct representative algorithms of this class are: Denclue [8], DBCLASD [10], Optics [1] and O-Cluster [9]. The characteristic feature of such algorithms is that they perceive the elements of the data set not as particular objects, but a substance that fills data space. We say that the area is of high density if it contains a large number of elements; otherwise, the area is of low density. Under this understanding of space, a cluster is an area of density exceeding the required threshold value or greater than the density of the surrounding space. The areas that do not constitute clusters are treated as noise.

A distinct feature of DBSCAN is its simplicity. Although simple, DBSCAN is very efficient and finds clusters of high quality, as well as determines noise correctly. In this paper, we propose a method improving the performance of this algorithm. Our solution is based on the observation that the fewer points are in a data set, the shorter is a clustering process. In accordance with this observation, we propose a method that deletes some points from the data set as soon as possible without changing the clustering results. The usefulness of the method is verified experimentally both for indexed and non-indexed data.

The paper has the following layout. Section 2 recalls the notion of a cluster, which was introduced in [5]. Section 3 presents the DBSCAN algorithm. In Section 4, we offer an optimized version of DBSCAN. Section 5 reports the performance of the DBSCAN algorithm and its optimized version on different indexed and non-indexed data sets. The results are concluded in Section 6.

2 Basic Notions

The authors of the DBSCAN algorithm [5] understand a cluster as an area of high density. Low density areas constitute noise. A point in space is considered a member of a cluster if there are a sufficient number of points within a given distance. The distance between two points p and q will be denoted by dist(p,q). The authors of DBSCAN do not impose the usage of any specific distance metric[1]. Depending on an application, one metric may be more suitable than the other. In particular, if Euclidean distance is used, a neighborhood of a point has a spherical shape; when Manhattan distance is used the shape is rectangular. For simplicity of the presentation, the examples we provide in this paper refer to Euclidean distance. Below we recall formal definitions of a density based cluster and related notions.

Definition 1. (Eps-neighborhood of a point)
Eps-neighborhood of a point p ($N_{Eps}(p)$) is a set of points q in data set D that are distant from p by no more than Eps; that is, $N_{Eps}(p) = \{q \in D \mid dist(p, q) \leq Eps\}$.

Definition 2. (a core point)
p is a *core point* if its *Eps-neighborhood* contains at least MinPts points; that is, if $| N_{Eps}(p) | \geq$ MinPts.

Definition 3. (directly density-reachable points)
A point p is *directly density-reachable* from a point q w.r.t. Eps and MinPts if the following two conditions are satisfied:
1) $p \in N_{Eps}(q)$,
2) q is a core point.

Suppose MinPts = 6. Point q in Figure 1a has 6 neighbors (including itself) in its neighborhood $N_{Eps}(q)$, so it is a core point. Point p has 2 neighbors in $N_{Eps}(p)$, hence it is not a core point. Point p, however, is directly density-reachable from q, while q is not directly density-reachable from p.

Definition 4. (density-reachable points)
A point p is *density-reachable* from a point q w.r.t. Eps and MinPts if there is a sequence of points $p_1, \ldots, p_n$ such that $p_1 = q$, $p_n = p$ and p_{i+1} is directly density-reachable from p_i, i = 1..n-1.

[1] Overview of metrics used for clustering purposes can be found e.g. in [7].

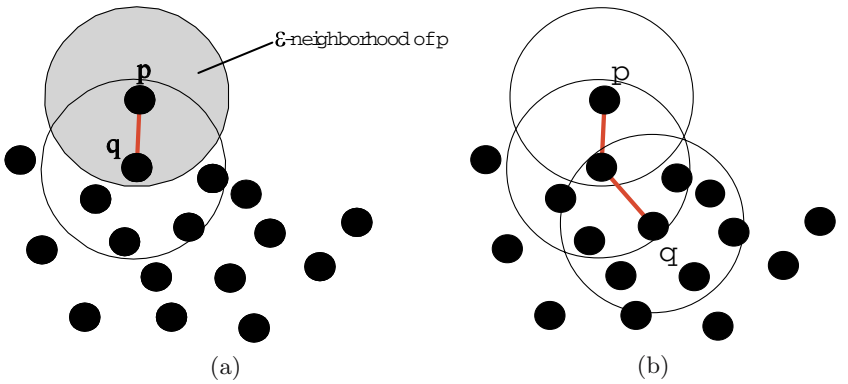


Fig. 1. (**a**) p is directly density-reachable from q (MinPts = 6). (**b**) p is density-reachable from q (MinPts = 6)

Point p in Figure 1b is density-reachable from point q, since there is a point such that p is directly density-reachable from it and it is directly density-reachable from q. Please note that p, which is density-reachable from core point q, is not a core point.

Definition 5. (a border point)
p is a *border point* if it is not a core point, but is density-reachable from some core point.

Hence, a point is a border one if it is not a core point, but belongs to the Eps-neighborhood of some core point.

Let C(p) determine all points in D that are density-reachable from point p. Clearly, if p is not a core point, then C(p) is empty.

Definition 6. (cluster)[2]
A cluster is a non-empty set of all points in D that are density-reachable from a core point.

Hence, each C(p) is a cluster provided p is a core point. Interestingly, if p and q are core points belonging to the same cluster, then C(p) = C(q); that is, both points determine the same cluster [5]. Thus, a core point p belongs to exactly one cluster, namely to C(p). However, a border point may belong to more than one cluster.

Definition 7. (noise)
The *noise* is the set of all points in D that do not belong to any cluster; that is, the set of all points in D that are not density-reachable from any core point.

[2] This definition differs from the original one provided in [5]. However, the definition of a cluster presented here is equivalent to the original one by Lemma 1 in [5], and is more suitable for our presentation.

Hence, points that are neither core points, nor border ones, are noise.

3 Clustering with DBSCAN

In this section, we recall the DBSCAN algorithm. Its input parameters are: the set of points D, the maximal radius of the neighborhood Eps and the required minimal number of points MinPts within Eps-neighborhood (please, see [1,5] for the method of determining appropriate values of parameters Eps and MinPts). There is associated ClusterId field with each point in D. Initially, each point in D is assumed to be unclassified (ClusterId = UNCLASSIFIED).

The algorithm starts with generating a label for first cluster to be found. Then, it analyzes point after point in D. Clearly, the value of ClusterId field of a first checked point equals UNCLASSIFIED. As will follow from the logic of the program, the labels of some points may be changed before they are analyzed. Such a situation occurs when a point is density-reachable from a core point analyzed earlier. In this case, the point will be assigned ahead to the cluster of the core point. Such pre-classified points will be skipped from analysis. If, however, a currently analyzed point p is still unclassified, then the ExpandCluster function (described later) is called for it. If p is a core point, then the function assigns the current cluster's label to all points in C(p), and DBSCAN generates a label for a new cluster to be built. Otherwise, the function assigns label NOISE to point p.

After all points in D are analyzed, ClusterId of each point is assigned either a label of a respective cluster or label NOISE.

Algorithm DBSCAN(set of points D, Eps, MinPts);

```
ClusterId = label of first cluster;
for each point p in set D
    if (p.ClusterId = UNCLASSIFIED) then
        if ExpandCluster(D, p, ClusterId, Eps, MinPts) then
            ClusterId = NextId(ClusterId)
        endif
    endif
endfor
```

The ExpandCluster function starts with calculating Eps-neighborhood of point p passed as a parameter. If its Eps-neighborhood does not contain sufficient number of points, then it is not a core point. In such a case, it is temporary labeled as a noise point and the function reports failure of expanding a cluster. Otherwise, the examined point is a core point and all points that are density-reachable from it will constitute a cluster. First, all points in the neighborhood of the examined point are assigned a label (ClId) of the

currently created cluster. All points in $N_{Eps}(p)$, except for p, are stored in the seeds collection and are subject to determination of their Eps-neighborhood. Each seed point, which is a core point, further extends the seeds collection with the points in its Eps-neighborhood that are still unclassified. Clearly, the points in the seeds collection that were classified earlier as noise are border points of the current cluster, and hence are assigned the current cluster label[3]. After processing a seed point, it is deleted from the seeds collection. The function ends when all points found as cluster seeds are processed.

---

function ExpandCluster(D, point p, cluster label ClId, Eps, MinPts)

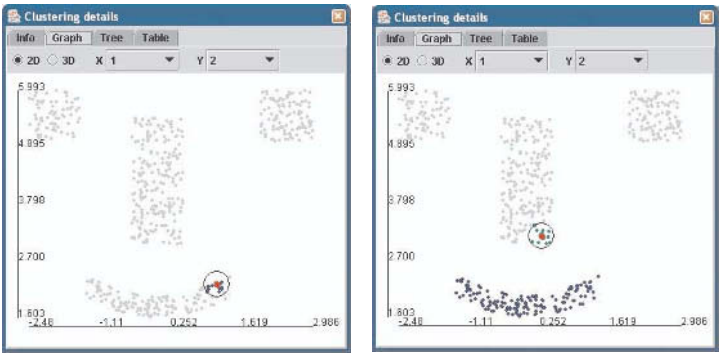
```
seeds = Neighborhood(D, p, Eps);
if |seeds| < MinPts then
    p.ClusterId = NOISE;
    return FALSE
else
    for each point q in seeds do                    // including point p
        q.ClusterId = ClId;
    endfor
    delete p from seeds;
    while |seeds| > 0 do
        curPoint = first point in seeds;
        curSeeds = Neighborhood(D, curPoint, Eps);
        if |curSeeds| >= MinPts then
            for each point q in curSeeds do
                if q.ClusterId = UNCLASSIFIED then
                /* N_Eps(q) has not been evaluated yet, so q is added to seeds */
                    q.ClusterId = ClId;
                    append q to seeds;
                elseif q.ClusterId = NOISE then
                /* N_Eps(q) has been evaluated already, so q is not added to seeds */
                    q.ClusterId = ClId;
                endif
            endfor
        endif
        delete curPoint from seeds;
    endwhile
    return TRUE
endif
```

Please note that Eps-neighborhood of point p passed as parameter to the

[3] Although a border point may belong to many clusters, DBSCAN assigns it only to one of them. It is straightforward to modify the algorithm so that the border points are assigned to all including clusters.

ExpandCluster function may contain points classified earlier as NOISE. For such points, the function redundantly calculates their neighborhoods.

Figure 2 illustrates the initial part of sample execution of the DBSCAN algorithm.



(a) (b)

Fig. 2. (**a**) Neighborhood of the first (core) point assigned to a cluster. (**b**) Subsequent assignment of density-reachable points forms first cluster; the initial seeds are determined for the second cluster

4 Optimizing DBSCAN by Early Removal of Core Points

The solution we propose consists in removing a point from set D as soon as it is found to be a core point by the ExpandCluster function.

Clearly, if a point p is deleted, then for each point q in its Eps-neigborhood, $| N_{Eps}(q) |$ determined in the reduced set D is smaller than $| N_{Eps}(q) |$ determined in the original set D. In order to calculate the sizes of Eps-neighborhoods correctly, there is associated the NeighborsNo field with each point in D. The NeighborsNo field of point q stores the number of the q's neighboring core points that were earlier deleted from D. More specifically, whenever a core point p is deleted, the NeighborsNo field is incremented for all points in $N_{Eps}(p)$. Now, the real size of neighborhood of each point r remaining in D is counted as the sum of the cardinality of $N_{Eps}(r)$ found in the reduced set D and the value of the NeighborsNo field of r.

The ExpandCluster function can be optimized further by skipping redundant determination of neighborhood of points already classified as NOISE.

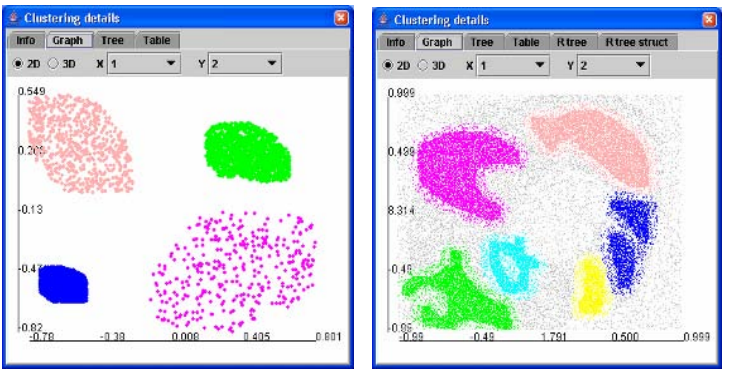
Our optimized version of the ExpandCluster function is called ERCP-ExpandCluster (ExpandCluster using Early Removal of Core Points). The code of the ERCP-ExpandCluster function is provided below.

function ERCP-ExpandCluster(D, point p, ClId, Eps, MinPts)

```
seeds = Neighborhood(D, p, Eps);
p.NeighborsNo = p.NeighborsNo + |seeds|;
if p.NeighborsNo < MinPts then
    p.ClusterId = NOISE;
    return FALSE
else
    p.ClusterId = ClId;
    move p from D to D';          // clustered core points will be stored in D'
    delete p from seeds;
    for each point q in seeds do
        if q.ClusterId = NOISE then
            q.ClusterId = ClId;
            /* N_Eps(q) of noise q shall not be determined */
            delete q from seeds;
        else
            q.ClusterId = ClId;
            q.NeighborsNo = q.NeighborsNo + 1;
        endif;
    endfor
    while |seeds|> 0 do
        curPoint = first point in seeds;
        curSeeds = Neighborhood(D, curPoint ,Eps);
        curPoint.NeighborsNo = curPoint.NeighborsNo + |curSeeds|;
            if curPoint.NeighborsNo >= MinPts then
            for each point q in curSeeds do
                if q.ClusterId = UNCLASSIFIED then
                    q.ClusterId = ClId;
                    q.NeighborsNo = q.NeighborsNo + 1;
                    append q to seeds;
                elseif q.ClusterId = NOISE then
                    q.ClusterId = ClId;          // q is a border point
                endif
            endfor
            move curPoint from D to D';
        endif
        delete curPoint from seeds;
    endwhile
    return TRUE
endif
```

5 Experimental Results

To examine the practical relevance of our proposal, we performed an experimental evaluation of the optimized algorithm and compared it to that of original DBSCAN. We tested data of dimensionality from 2 up to 8. We have also examined how the usage of spatial indexes for evaluating Eps-neighborhoods influences the algorithms' performance. Namely, we investigated the usefulness of applying R tree index [6], R*-tree index [4], and UB-tree index [2,3] for clustering with DBSCAN. Table 1 presents the experimental results. The table shows that the optimized version performs faster then the original DBSCAN by up to 50%. The best speed-up can be observed for data sets



(a) (b)

Fig. 3. (**a**) Data set d1-2D. (**b**) Data set ds1-2D

containing little noise. In particular, such a speed-up can be observed for little noise data set d1-2D, which is presented in Figure 3a. On the other hand, for large noise data set d1s-2D, which is presented in Figure 3b, speed-up does not exceed 40%. This phenomenon can be justified as follows: Most of the points in data sets with little noise are core points and are removed from the set immediately after their neighborhood is determined, which makes the evaluation of neighborhood of the remaining points faster. A simplified theoretical cost model of neighborhood evaluation can also lead us to this result. Consider f(n) to be the cost of evaluating Eps-neighborhood of a given point in the set of n points[4]. Original DBSCAN has to perform such evaluation for every point in the set giving the total cost $F_1(n) = n \times f(n)$. Optimized version also evaluates neighborhood for every point, but the cost of single evaluation decreases while subsequent points are removed from the data set.

[4] Notice that for simplicity we assumed that the cost of determining Eps-neighborhood of a given point is the same for every point and depends only on the size of data set.

Table 1. Results of experiments

Clustering time [s]				
DataSet	Index	Original DBSCAN	Optimized DBSCAN	Optimized/ Original
d1-2D	R-tree	3.23	1.65	51.08%
size: 5550	R*-tree	2.87	1.44	50.17%
MinPts: 5	UB-tree	2.66	1.44	54.14%
Eps: 0.1	no index	4.77	2.48	51.99%
ds1-2D	R-tree	10.5	7.49	71.33%
size: 50000	R*-tree	8.8	6.2	70.45%
MinPts: 5	UB-tree	22.2	13.4	60.36%
Eps: 0.1	no index	313	253	80.83%
ds2-2D	R-tree	18.7	11.5	61.50%
size: 50000	R*-tree	13.9	8.68	62.45%
MinPts: 5	UB-tree	30.8	18.2	59.09%
Eps: 0.1	no index	318	266	83.65%
ds3-2D	R-tree	5.17	3.94	76.21%
size: 40000	R*-tree	3.97	2.86	72.04%
MinPts: 5	UB-tree	9.8	6.7	68.37%
Eps: 0.1	no index	201	168	83.58%
d1-4D	R-tree	5.76	3	52.08%
size: 5550	R*-tree	5.46	2.78	50.92%
MinPts: 5	UB-tree	6.89	5.31	77.07%
Eps: 0.1	no index	7.91	4.05	51.20%
ds1-4D	R-tree	21.7	13.1	60.37%
size: 5550	R*-tree	16.8	10	59.52%
MinPts: 5	UB-tree	103	77	74.76%
Eps: 0.1	no index	409	303	74.08%
d1-8D	Rtree	9.32	5.09	54.61%
size: 5550	R*-tree	9.5	5.33	56.11%
MinPts: 5	UB-tree	58.8	46	78.23%
Eps: 0.1	no index	7.94	4.39	55.29%

In an ideal case, every point in the data set is a core point and can be removed from it as soon as its neighborhood has been found. Therefore, the total cost of neighborhoods' evaluation by the optimized version of DBSCAN $F_2(n) = f(n) + f(n1) + \ldots + f(2) + f(1)$. In the case when index is not used, we have $f(n) = n$. Hence, $F_1(n) = n \times n = n^2$ and $F_2(n) = n + (n-1) + \ldots + 2 + 1 = n \times (1+n)/2 \approx 1/2 \times F_1(n)$.

Our optimization is particularly useful for data of high dimensionality. Indexes on such data are known to be of little use in finding Eps-neighborhood. Our experiments confirm this fact. Table 1 shows that using an index for 8 dimensional data set d1-8D results in slower clustering of data than clustering without index. This observation holds both for clustering data by means of original DBSCAN and its optimized version. The achieved experimental results did not point out which index is the best in terms of DBSCAN efficiency. Usefulness of an index depends on data and input parameters.

6 Conclusions

A method for improving the performance of DBSCAN has been offered. It is based on early removal of core points. The experiments show that using the proposed method speeds up DBSCAN's performance by up to 50%. As a future work, we plan to enhance this method by enabling early removal of border and noise points as well.

References

1. Ankerst, M., Breunig, M.M. et al. (1999) OPTICS: Ordering Points To Identify the Clustering Structure. SIGMOD, 49-60
2. Bayer, R. (1997) The Universal B-Tree for Multidimensional Indexing. WWCA, 198-209
3. Bayer, R. (1997) UB-Trees and UB-Cache, A new Processing Paradigm for Database Systems. Technical Report TUM-I9722
4. Beckmann, N., Kriegel, H.P. (1990) The R*-tree: An Efficient and Robust Access Method for Points and Rectangles. SIGMOD, 322-331
5. Ester, M., Kriegel, H.P. et al. (1996) A Density-Based Algorithm for Discovering Clusters in Large Spatial Database with Noise. KDD, 226-231
6. Guttman, A.(1984) R-Trees: A Dynamic Index Structure For Spatial Searching. SIGMOD, 47-57
7. Han, J., Kamber, M.(2000) Data Mining: Concepts and Techniques. Morgan Kaufmann Publishers
8. Hinneburg, A., Keim, D.A. (1998) An Efficient Approach to Clustering in Large Multimedia Databases with Noise. KDD, 58-65
9. Milenova, B.L., Campus, M.M. (2002) O-Cluster: Scalable Clustering of Large High Dimensional Data Set. ICDM, 290-297
10. Xu, X., Ester, M. et al. (1998) A Distribution-Based Clustering Algorithm for Mining in Large Spatial Databases. ICDE, 324-331

Innertron: New Methodology of Facial Recognition, Part I

Rory A. Lewis[1] and Zbigniew W. Raś[1,2]

[1] UNC-Charlotte, Computer Science Dept., Charlotte, NC 28223, USA
[2] Polish Academy of Sciences, Institute of Computer Science, Ordona 21, 01-237 Warsaw, Poland

Abstract. On October 10, 2001, the US President identified the most wanted persons sought by the United States of America. Agencies such as FBI, CIA and Homeland Security spread images of the most wanted persons across the United States. Even though US citizens saw their images on television, the Internet and posters, computers had, and still have for that matter, no ability at all to identify these persons. To date FBI, CIA and Homeland Security depend entirely on human beings, not computers, to identify persons at borders and international airports. In other words, facial recognition remains an incompetent technology.

Accordingly, authors first succinctly show the weaknesses of the current facial recognition methodologies, namely Eigenface Technology, Local Feature Analysis (from the classical 7 point to the 32-50 blocks approach), the Scale-Space Approach, Morphological Operations and industrial or patented methodologies such as ILEFIS™, Viisage™, Visionics™ and Cognitec's FaceVACS-Logon™, Identix™ and Neven Vision™. Secondly, they introduce a completely new, simple and robust methodology called *Innertron*.

1 Introduction: Commercial Attempts at Facial Recognition

Biometrics is the study of measurable biological characteristics that computer can identify. Biometric identification schemes include those of the face, fingerprint, hand, retina, iris, vein and voice. Biometrics covers Identification and Verification. The latter is when a computer compares an image to a database of images yielding a resultant list of probable matches. Conversely, in Identification the image is unknown and may not even exist in the database. In other words, Identification is much more complex than Verification. This paper first analyzes Verification Facial Recognition after which it proposes a methodology that encompasses both Verification and Identification facial recognition.

In September 24, 1999, the National Institute of Justice worked with its Office of Law Enforcement Technology Commercialization (*OLETC*) to initiate the commercialization of a 3D facial identification technology called *ILEFIS*, or Integrated Law Enforcement Face-Identification System. The developers of the technology claimed the ability to identify, while screening millions of images a second, a facial image from angled photos and video

images. Wheeling Jesuit University developed and patented the technology under contract to the National Institute of Justice. West Virginia's Missing Children Clearinghouse tested the technology to look for missing children. The process included first scanning pictures found on the internet and then comparing it to the images of children stored in its database. However, the results that the Wheeling Jesuit University proffered may have worked with its own, narrowly defined database, however, the system failed when subjected to a broader dataset that included facial changes in expressions, hair, yaw of the face or lighting conditions, i.e., the real world. *OLETC* and West Virginia's Missing Children Clearinghouse abandoned the project sometime in 2001.

The aforementioned scenario is the atypical facial recognition modus operandi. Step one, use classical algorithms such as $Eigenface$, the 7-point or 32-50 block Local Feature Analysis, change it slightly and prove it works well on a given, set of faces within well predefined limits in lighting, pixilation, distance from camera, height of camera, facial expression, facial hair, and angle of the face. Step two, sell it, market it, and maybe even sell it to the *DOD*. Step three, fail miserably when the system attempts to match persons that are not in the database, covering parts of their faces or falling, in any way, outside the parameters of its original predefined database.

According to Forbes Magazine in 2004 [1] and the Frost & Sullivan 2003 report [2], *Viisage* has 47% market share in the commercial facial recognition market. *Viisage* recently helped National Geographic verify the identity of the *Afghan Girl*, Sharbut Gula, by comparing recent photos of her with the famous image that graced the cover of the magazine 17 years ago [3]. Amongst *Viisage's* claims to fame is: [1] Acquisition of Europe's leading facial recognition company for 13.2 million dollars; [2] Implementation of *Viisage* at Logan International Airport.

Initially the aforementioned sounds hopeful, that facial recognition is a reality. However, upon closer inspection, it is clear that the *Viisage's* system has failed terribly. Quoting the Boston Globe: *A test at Boston's Logan International Airport has found the machines were fooled when passengers turned their heads in certain directions, and screeners became overtaxed by the burdens of having to check passengers against a large pool of faces that closely resemble theirs* [4]. Here, the database consisted of a mere 40 airport employees who played potential terrorists attempted to pass through two security checkpoints incorporating the *Viisage* system cameras. Over the course of three-months and 249 tests the system failed 96 times to detect the volunteers. The airport's report called the rate of inaccuracy *excessive* [5]. Accordingly, Boston airport removed the *Viisage* system. Again, the hype and marketing on Wall Street was fantastic, but upon implementing the system on a dataset falling outside the parameters of its predefined experimental database, the system failed completely.

In 2001, the Tampa Police Department installed several dozen cameras, assigned staff to monitor them, and installed a face recognition application called Face-IT^{TM} manufactured by the Visionics^{TM} Corporation of New Jersey in cooperation with Identix^{TM} [6]. On June 29, 2001, the department began scanning the faces of citizens as they walked down Seventh Avenue in the Ybor City neighborhood. The system never correctly identified a single face in its database of suspects, let alone resulted in any arrests. The Police department suspended the system on August 11, 2001, and has not been in operation since [7]. Business Week reported that the well-known security consultant Richard Smith investigated FaceIT^{TM} at Ybor. Smith concluded that changes in lighting, the quality of the camera, the angle from which a face was photographed, the facial expression, the composition of the background of a photograph, the donning of sunglasses or even regular glasses rendered the system worthless. [8]. To make the matter worse, The New York Times reported that *the most damaging publicity [for Viisage's facial technology] came from tests of face-recognition software and video-surveillance cameras used to spot criminal suspects on the streets of Tampa, Florida, and Virginia Beach* [9].

Similarly, Cognitec's *FaceVACS-Logon* system incorporates a Support Vector Machine (SVM) to capture facial features. In the event of a positive match, the authorized person is granted access to a PC. However, a short video clip of a registered person outfoxed the system [10]. More so, still images taken on a digital camera also proved effective in gaining back door access [11].

E.H. Neven, of *Neven Vision*, while directing the USC Laboratory for Biological and Computational Vision, the team won the DARPA sponsored 1997 *FERET* test on face recognition systems [12]. In 2002, Neven's *Eyematic* team achieved high scores in the Face Recognition Vendor Test. *Neven Vision* claims that its facial recognition module yields an unrivalled *false accept rate* of one in 500,000. The system combines its technology with Graphco^{TM} technology that identifies individuals by the sound and rhythm of their voices, not by speaking particular words [13]. There is no doubt that *Neven Vision's* system seems to be fantastic. However, Neven technology has an Achilles heel, namely *Eigenface Technology* - the subject of the next section.

2 Commercialization and Eigenface technology

In May 2001, K. Okada and H. Neven [14], explained in detail their methodology of using Eigenface values, which Neven claims has *an unrivalled false accept rate of one in 500,000* [13]. As first demonstrated by Graham and Allinson [16], Okada [15] explains that the Neven methodology trains an RBF network which reconstructed a full manifold representation in a universal eigenspace from a single view of an arbitrary pose.

Essentially, the Graham and Allinson's system is a linear combination of manifolds where the RBF network reconstructed a full manifold in eigenspace.

Herein, is the weakness of the Neven system. Okada, explains that the system requires images of 20 people whose head poses varied between 0 and 90 degrees along one depth-rotation axis in 10-degree intervals. Accordingly, it is not surprising that after shooting 9 images of each person, all at exactly 10 degrees, the results are fantastic.

Okada then explains the next step of the Neven process, where nine manifolds derived from these views interpolate a virtual manifold of an unknown person. The system yielded a 93.1% average correct-identification rate with a database of 20 known persons. Again, as evidenced before in this paper, the Neven system indeed works wonderfully inside of its well-defined world consisting of nine perfect shots at exactly 10 degrees for each person in the 20-person database. The issue is, what about the real world, i.e., the place where 300,000,000 fuzzy, small, 2D images each represent only one person?

Okada does mention that Wieghardt and von der Malsburg [17] recently proposed a methodology for constructing a parametric eigenspace representing an object without using explicit knowledge of 3D head angles. Here they incorporated a similarity-based clustering algorithm to construct view-specific local eigenspaces. Wieghardt then pieced together the eigenspaces in a global coordinate space by using multi dimensional scaling (MDS). What is lacking however is a report of the identification performance of their system.

3 Eigenface Technology: the Foundation of Facial Recognition?

In 1991, Turk and Pentland wrote a paper that changed the facial recognition world. They called it *Face Recognition Using Eigenfaces* [18]. A continuation of this paper soon thereafter won the *Most Influential Paper of the Decade* award from the *IAPR MVA* at the 2000 conference [19], [20]. In short, the new technology reconstructed a face by superimposing a set of *eigenfaces*. The similarity between the two facial images is determined based on the coefficient of the relevant *eigenfaces*. They consist of eigenvectors which are also known as *proper vectors* or *latent vectors* that are typically associated with a linear system of equations such as a matrix equation [21]. Accordingly, *eigenfaces* consist of a set of the aforementioned linear eigenvectors wherein the operators are non-zero vectors which result in a scalar multiple of themselves [22]. It is this scalar multiple which is referred to as the eigenvalue associated with a particular eigenvector [23].

Turk theorized that any human face is merely a combination of parts of the eigenfaces. One person's face may consist of 25% from face 1, 25% from face 2, 4% of face 3, n% of face X [25]. Typically the process is as follows:

- Obtain a large set of digitized images of human faces: a. shot under the same lighting conditions, b. shot using same yaw.
- Normalize to line up key features.

- Re-sample at the same pixel resolution (say mn) [25].
- Treat the result from the previous step as (mn)-dimensional vectors whose components are the values of their pixels.
- Extract the eigenvectors of the covariance matrix located in the statistical distribution of eigenvectors.
- Weight the eigenfaces and sum to create an approximate gray-scale rendering of a human face.

In spite of its popularity, the methodology based on Eigenfaces has problems because of the way its approximate gray-scale rendering of a human face is constructed. For instance person's face would not obtain an eigenvector match if he covered portions of his face with a scarf, hand or any other object. His face would not obtain an eigenvector match if he grew facial hair or grew long hair that covered portions of his mouth. His face would not obtain an eigenvector match if he made a change in facial expression such as smiling or lifting up his eyebrows. His face would not obtain an eigenvector match if there was a shadow covering a portion of his face. . . and so on.

Finally, a methodology dealing with facial recognition has to take into consideration another practical issue (which is not in favour of eigenfaces): the data base of 300,000,000 people in the United States and of all the terrorists contains mere, out of focus, inconsistent, fuzzy 2 dimensional images. To propose a successful facial recognition strategy, it has to be tested on real, sometime very incomplete and unprecise data.

4 Innertron: A new methodology of facial recognition

Step 1 - 01 is the first step in the first portion of the *Innertron* methodology, the pseudo-first-stage-sort. Looking at the two faces we may be familiar with, we ask: *What do humans perceive neurologically that makes it so easy for us to distinguish, and recognize these two faces?* How is that either one could do a wig, grow facial hair, smile, frown, put a scarf over their mouth or subject themselves to a host of various lighting conditions and we, human beings, can instantly, recognize and distinguish between the two faces?

This paper asserts that there is two-part recognition. Think of the times you may have seen someone that you were convinced was somebody else. You may even have approached them and looked carefully into the person's eyes. You may even have spoken to them and still we convinced. This is proof that there was a second phase. But for the fact that your recognition system allowed you to go from one place in recognition perception - to - another place of intense or further investigation is suffice to prove that one went beyond the first step.

This paper asserts that humans look at generalities such as length of face, height, skin color, age, possible gender before moving onto step two. Computers cannot do this so a pseudo step in recognition is created. We call

it *Innertron*. *Tron* as in computers and *Inner* as inside of us, biological as in biometrics - facial recognition. The following steps resolve a two-dimensional image of a profile. Analysis of input involving profiles and various levels of yaw are forthcoming.

Step 1-01 is simply the retrieval of the input. This will include at later dates the retrieval from various forms of video shot, mpeg1, scanned images and images received from the internet.

Step 1-02 is the analysis of the histogram whereby the levels are set to Grayscale, 8 bits per channel 000,0.39,189. Thus far this level is functioning well. However an optimal histogram level will be forthcoming. The main purpose is to keep only the most significant attributes of the three important groups being whites, blacks and mid-tone grey.

Step 1-03 simply incorporates Kanade's technology of face - finding [26], [27] where Kanade's methodology overcomes variations in faces by incorporating a two-part strategy. For variations in pose, Kanade uses a view-based approach with multiple detectors each specialized to a specific orientation or yaw of the face. Thereafter a statistical modelling promulgates itself within each of the detectors. This last step of the Kanade methodology accounts for the remaining variations of facial positions or poses. The most significant aspect of *Step 1-03* is that both the eyes and face is detected. It is true that if the subject looking far left or far right that an adjustment will be made to equalize the eyes, or, set them looking straight ahead.

Step 1-04 uses an estimation of the location of the lateral rectus [28] muscle located at the outer portion of the eye. At this estimated point it searches for a convergence of two lines or patterns. Here it is in fact searching for the lateral commissure [29]. Commissures are the locations where the upper and lower eyelids fuse together. The medial commissure being that point nearest the nose and the lateral commissure, as mentioned, is on the other side closest to the temple. As seen on the illustration for *Step 1-04*, the program places a vertical line at the left lateral commissure. It is significant to note that this point is later positioned at (-5.0, 0) on the x-y axis. Furthermore, the program also takes into consideration the fact that there is more often than not a shadow or wrinkle that extends beyond the point of the left lateral commissure; this is the reason why it first considers the location of the lateral rectus muscle as the left lateral commissure is approximately at this spot. If the program continues searching for the left lateral commissure along the extended shadow, it corrects itself and researches for the left lateral commissure with a lower level of tolerance in the histogram.

Step 1-05 consists of two steps. The program first places a grid over the subject image. The grid covers the area (-6.0, 7.0) to (6.0, -8. 0). Each grid consists of 10 pixels per unit on the axis. The area of the photo will extend a border around said grid. If the grid is too big, the picture is simply enlarged and vice-versa. The second step with *Step 1 - 05* consists of positioning the

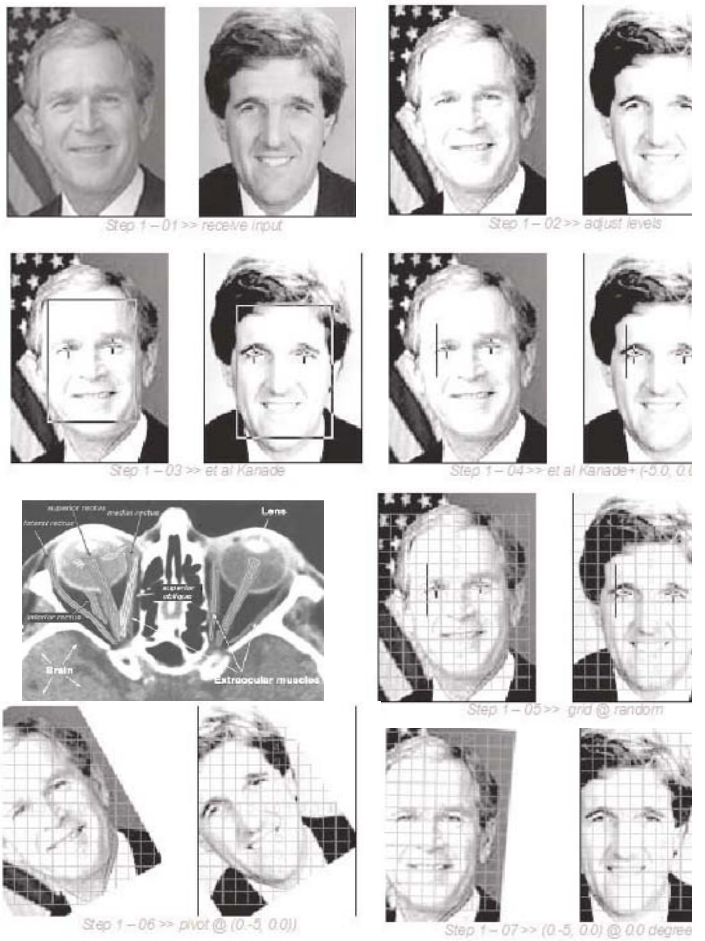


Fig. 1. Innertron: First Part Recognition

left lateral commissure obtained in *Step 1 - 04* and placing it exactly at (-5.0, 0) on the x-y axis.

Step 1-06 consists of pivoting the image with the pivot point at the left lateral commissure located at (-5.0, 0) on the x-y axis. For consistency, the first pivot extends to 45 degrees as shown in *Step 1 - 6* in order to compensate for the varying angles in the pictures. It should also be noted that the angle is determined by the angle between the left lateral commissure at (-5.0, 0) on the x-y axis and the approximate location of right lateral commissure from the x-axis at $y = 0$. It is approximate because the program actually utilizes the right Kanade point originally located in *Step 1 - 03* and adds onto it the distance from the left Kanade point to the left lateral commissure.

Step 1-07 places the right lateral commissure exactly onto the x-axis at y = 0, One may note that the left hand image looks as if it is now skew, however, the eyes of the subject are now level - the goal of *Step 1 - 07*. However, it is also clear that the distance from the left lateral commissure to the right lateral commissure on both of these images are different to one another.

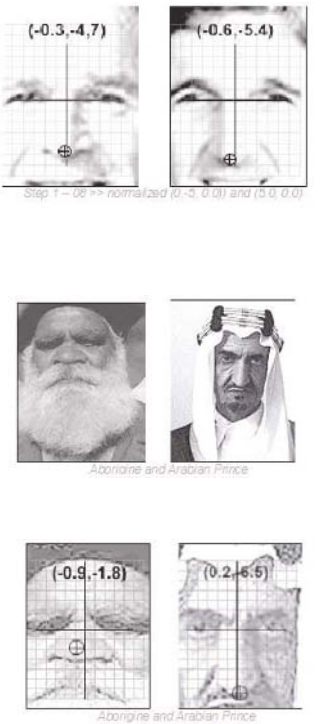


Fig. 2.

Step 1 - 08 is the final step in stage 1 - Normalization. As shown in the *Step 1 - 08* illustration (Figure 2), the right lateral commissure is now located at (5.0, 0.0). Considering that the left lateral commissure is located at (-5.0, 0.0) meaning that the distance between the left and right lateral commissures is exactly 10 points along the x-axis. It is pivotal that one realizes that every face will always have a distance between the left and right lateral of exactly ten points. It is true that the distance in reality will not be equal in all faces on planet earth, however, making this distance a constant 10 points eliminates a problem encountered by all previous facial recognition systems i.e., the distance that the subject is from the camera. Using the aforementioned methodology of normalization voids the need to know or care about

the distance the subject is from the camera at the time the original photo was shot, meaning the one located in the database, or, the distance between the camera and the subject when shot for identification purposes. The program now searches, while travelling down the y-axis from (0.0, 0.0) towards (0.0, -6.0) for a circular area . It will also consider where the demarcation between the bridge of the nose intersects with the shadow of the side of the nose - ends. The program also searches in-between two vertical anomalies at 2 - 3 points both left and right of the y-axis at x=o as these could be the outside of the subject's nostrils. The program also runs a horizontal scan from the same points to search for an continuation of a line extending substantially from both points, if it finds it, it knows it missed the circular area as that was the base of the nose - which is below the circle or tip of the nose. Once it has located the tip of the nose it simply registers this position as the first two points of recognition of the subjects face. The variance between the two subjects: (-0.3,-4.7) - (-.0.6, -5.4) = (0.3, 0.7). This is equivalent to 30 pixels on the x axis and 70 pixels on the y axis. One can appreciate the significant this pseudo-first-stage-sort plays when observing variance of faces in the following example. Take for example the Aborigine and the Arabian Prince (Figure 2). The tip of the Aborigine's nose is 18 pixels below the x-axis while the tip of the Arab's nose is 65 pixels below the x-axis. The variance between the Aborigine and the Arab is (-0.9,-1.8) - (0.2, -6.5) = (1.1, 4.7) or 11 pixels along the x-axis and 47 pixels along the y-axis.

References

1. Patsuris, P. (2004), *Homeland Security: Viisage Puts On A Good Face*, 04.22.04, 8:00 AM ET, See: http://www.forbes.com/technology/2004/04/22/cx_pp_0421visage_ii.html
2. OMIT, http://www.forbes.com/technology/2004/04/22/cx_pp_0421visage_ii.html
3. See: http://www.frost.com/prod/servlet/search-results.pag?srchid=28266978
4. Morrison, P. (2002), *More Than a Face in the Crowd*, WPI Transofmations, See: http://www.wpi.edu/News/Transformations/2002Spring/face.html
5. Bray, H. (2002), The Boston Globe, Jul 17, See: http://www.boston.com/dailyglobe2/198/metro/_Face_testing_at_Logan_is_found_lacking+.shtml
6. Willing, R. (2003), *Airport anti-terror systems flub tests*, USA TODAY, Posted 9/1/2003, 11:14 PM, Updated 9/2/2003, 7:53 AM, See: http://www.usatoday.com/news/nation/2003-09-01-faces-usat_x.htm
7. See: http://www.identix.com/products/pro_security_bnp_argus.html
8. Stanley, J., Steinhardt, B. (2002), *Drawing a Blank: The failure of facial recognition technology*, in Tampa, Florida, AN ACLU SPECIAL REPORT, Jan 3
9. Black, J. (2001), *Face It, Face-Cams Are Here to Stay*, BusinessWeek Online, NOVEMBER 15, 2001. See: http://www.businessweek.com/bwdaily/dnflash/nov2001/nf20011115_3919.htm
10. Feder B. (2004), *Technology Strains to Find Menace in the Crowd; Face-Recognition Technology Comes of Age. The New York Times*, 06/04/2004, See: http://www.policeone.com/police-technology/software/facial/articles/88418/

11. Thalheim, L., Krissler, J., Ziegler, P. (2002), *Body Check: Biometric Access Protection Devices and their Programs*, Put to the Testc't 11/2002, page 114 Biometrie, See: http://www.heise.de/ct/english/02/11/114/
12. Leyden, J. (2002), *The Register, Security - Biometric sensors beaten senseless in tests*, published Thursday, 23rd May, 2002, 05:44 GMT. See: http://www.theregister.com/2002/05/23/biometric_sensors_beaten_senseless/
13. Neven, H. (2004), *Neven Vision Machine Technology*, See: http://www.nevenvision.com/co_neven.html
14. Pasquerello, M. (2004), *Neven Vision and Graphco Blend Face and Voice Recognition to Secure Military, Government and Transportation Facilities*, in BiometritchNews, May 12, 2004, See: http://www.tmcnet.com/usubmit/2004/May/1041332.htm
15. Okada, K. (2001), *Analysis, Synthesis and Recognition of Human Faces with Pose Variations*, Ph.D. Thesis, Univ. of Southern California, Computer Science
16. Graham, D., Allinson, N. (1998), *Characterizing virtual eigensignatures for general purpose face recognition*, in Face Recognition: From Theory to Applications, Springer-Verlag, 446 - 456
17. Graham, D., Allinson, N. (1998), *Face recognition from unfamiliar views: Subspace methods and pose dependency*, in Proceedings of Third International Conference on Automatic Face and Gesture Recognition, 348 - 353
18. Wieghardt, J., von der Malsburg, C. (2000), *Pose-independent object representation by 2-D views*, in Proceedings of IEEE International Workshop on Biologically Motivated Computer Vision, May
19. Turk M. and Pentland, A. (1991), *Face recognition using eigenfaces*, in Proc. IEEE Conference on Computer Vision and Pattern Recognition, Maui, Hawaii
20. Turk M. and Pentland, A. (1991), *Eigenfaces for recognition*, in Journal of Cognitive Neuroscience, Vol. 3, No. 1, 71-86
21. See: http://www.cs.ucsb.edu/etc/news/past/2000.shtml
22. Marcus., Minc. 1988 Page 144.
23. Pentland A., Moghaddam, B., Starner, T., Oliyide, O., Turk, M. (1993), *View-Based and Modular Eigenspaces for Face Recognition*, Technical Report 245, MIT Media Lab.
24. Pentland L., Kirby, M. (1987), *Low-dimensional procedure for the characterization of human faces*, Journal of the Optical Society of America, No. 4, 519 - 524
25. Kirby, M., Sirovich, L. (1990), *Application of the Karhunen-Loeve procedure for the characterization of human faces*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 12, No. 1, 103 - 108
26. Turk M. and Pentland, A. (1991), *Eigenfaces for Recognition*, Journal of Cognitive Neuroscience, Vol. 3, No. 1
27. Rowley A., Baluja, S., Kanade, T. (1997), *Rotation invariant neural network-based face detection*, CS Technical Report, CMU-CS-97-201, CMU, Pittsburgh
28. Schneiderman H., Kanade, T. (1998), *Probabilistic modeling of local appearance and spatial relationships for object recognition*, in Proc. IEEE Conference on Computer Vision and Pattern Recognition, Santa Barbara, 4551
29. John Moran Eye Center University of Utah, See: http://webvision.med.utah.edu/imageswv/scan.jpeg
30. See: http://before.eyelids-blepharoplasty.com/sub_anatomy.html
31. See: http://www.drmeronk.com/blephnews/anatomy.html#external

Innertron: New Methodology of Facial Recognition, Part II

Rory A. Lewis[1] and Zbigniew W. Raś[1,2]

[1] UNC-Charlotte, Computer Science Dept., Charlotte, NC 28223, USA
[2] Polish Academy of Sciences, Institute of Computer Science, Ordona 21, 01-237 Warsaw, Poland

Abstract. The first part of the Innertron methodology for facial recognition, presented in [1], removes those faces that certainly do not fall within the constraints of the nose domain with an upper limit at (2.0, -1.5) to (-2.0, -1.5) and a lower domains at (3.0, -7.5) to (-3.0, -7.5). The 300 square pixels will contain the nose of the subject sought. The second part of the Innertron methodology, presented in this paper, in essence knows that the subject sought is somewhere *close by* and now starts looking at features such as the eyebrows, angles of the eyes, thickness of the nose, shape of the nostril. Seven regions are distinguished that take into account the ability to lift eyebrows and most importantly the ability to cover portions of the face and still garner a *hit* in the database.

1 Introduction: Innertron methodology for facial recognition

Herein we introduce a methodology - the Innertron methodology that overcomes the problems of shadow on a face, expressions on a face, covering portions of the face with facial hair, hands and other objects. Eigenvectors, eigenfaces, eigenvalues and Fishervalues are no longer a factor, a new methodology exists that overcomes the aforementioned problem that tagged along with the previous methodologies.

2 Innertron methodology, second part

Step 2-01 (see Figure 1) is the first step in the second portion of the Innertron methodology. The Innertron methodology distinguishes seven areas of relevance. It starts at #3 (Figure 1) because it already has #1 and #2 from stage one [1].

- Region #3 consists of a 20×20 pixel area centered at (-5.0, 0.0) and (5.0, 0.0). This means that #3 contains the left and right lateral commissures centered in each of the two 20×20 boxes.

- Region #4 extends from (0.0, 0.0) 30 pixels along the x-axis and 10 pixels below and above the x-axis. One region travels to the left Medial Commissure and the other towards the right Medial Commissure [See Appendix 2][2]. The medial commissure is the point which is nearest the nose. Each medial commissure falls about 66% of the distance away from (0.0, 0.0). The Innertron methodology requires this because it also measures the thickness of the bridge of the nose in the first half of the Region #4.

- Regions #5, #7 and #9 are the three vertically elongated regions sitting on the x-axis extending vertically 30 pixels. These three areas measure the eyebrows and account for the possibility of lifting up the eyebrows. Focusing on Region #5, it is centered vertically on the medial commissure extending left and right of it 10 pixels in each direction. Vertically it extends 30 pixels in height.

- Region #6 is the Innertron methodology of defining the thickness of the bridge of the nose and the angle at which it intersects the cheekbones. The program utilizes a spiral rather than a box because there is no definitive place one can consistently place a box and secondly the box would always be a different length as it sits in-between whatever distance is between the tip of the nose and the medial commissures.

- Region #7 are two thin 5×30 rectangles that simply find the height the eyebrows are sitting and then it adds a variance because it does not know whether the eyebrows were or are raised or not. This still finding a match even if they were raised, i.e., at a different height above the eyes in the database compared image on record.

- Region #8 is centered vertically on the lateral commissure extending left and right of it 10 pixels in each direction. Vertically it extends 30 pixels in height.

- Region #9 is centered horizontally on the tip of the nose and captures the nostrils.

Step 2-02 merely separates the overlapping areas by raising Regions #3 and #4 to the top of the page. This step is for illustrative purposes and for debugging the C.

Step 2-03 is also a step shown for illustrative purposes. One can identify how the seven regions' left and right areas lie next to one another, centered vertically on the page. At this point however the regions are mirrored against one another.

Step 2-04 is similar to the previous step however, the regions on right side are flipped horizontally in order to have both sides of the regions falling into the same plane as the regions lying on the left side of the face.

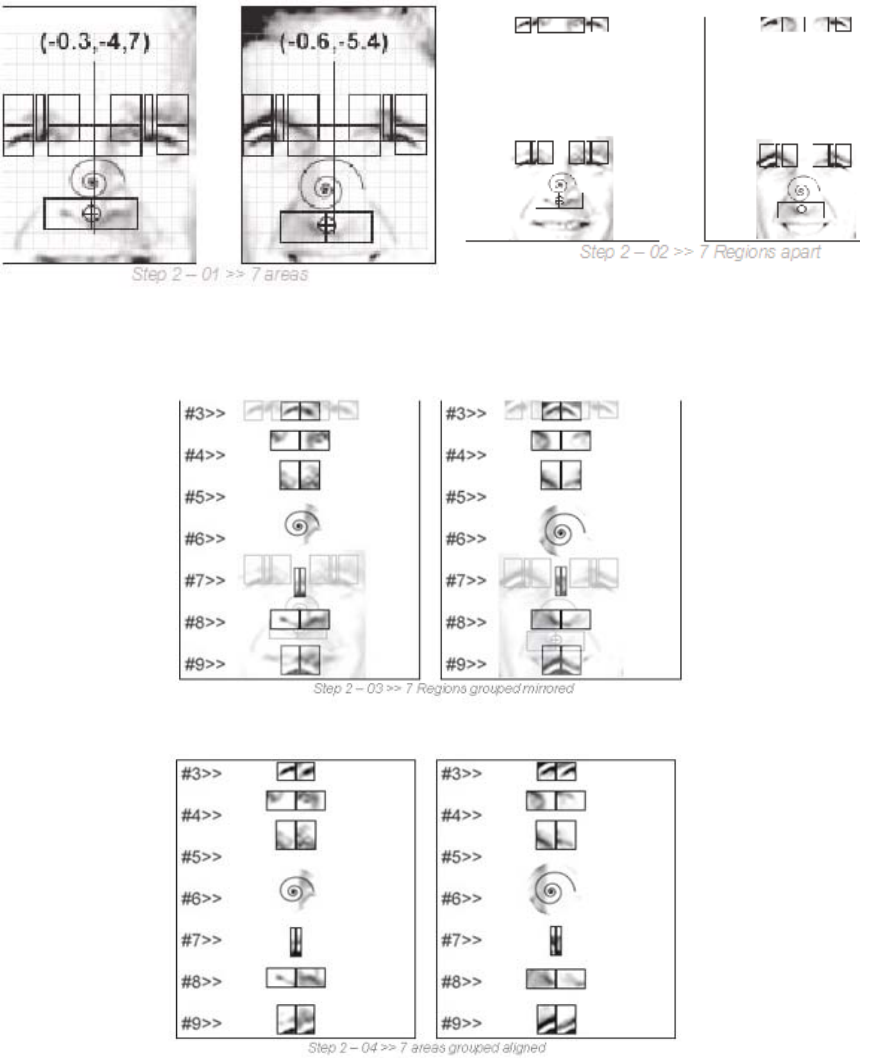


Fig. 1. Innertron: Second Part

Step 2-05 (see Figure 2) converts the flipped right side to a *Difference* zone. Whereby the difference between its histogram at a particular pixel, to the level of the pixel below it, i.e., the right regions form the new image.

Step 2-06 also inverts the underlying regions of the left side of the face. With the difference zone over it, the Innertron methodology herein overcomes the problem of shadow over one region, or an object covering only one region such as a scarf or an eye-patch. Innertron methodology still yields a numerical

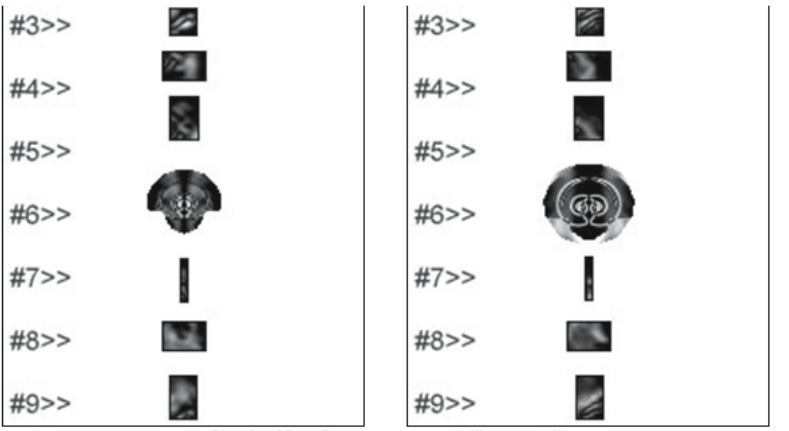


Fig. 2. Innertron: Second Part (Continuation)

representation through only looking and having the ability to ignore the polluted region.

Fig. 3.

Step 2-07 incorporates a scalar 4×4 box comprising 10×10 pixels numbered as follows. First the Innertron methodology transforms each region into 10×10 pixels with distortion. First outlining, then averaging, then straight lining in-between the outer vectors one realizes a simple yet robust grid (see Figure 3, left-top picture):

(-0.3,-4.7)@[12,42][31,34][42,12][32,34][12][41.41][12,42] /George Bush/,

(-0.6,-5.4)@[21,42][32,24][42,14][31,44][12][42.32][11,42] /John Kerry/.

3 Conclusion

The first sort, named the pseudo-first-sorts' upper domains range is about where *Aborigine Man* (see Part 1 of this paper [1]) lies [from (2.0, -1.5) to (-2.0, -1.5)]. The pseudo-first sorts lower domains at about the area where *Arab Prince* lies [from (3.0, -7.5) to (-3.0, -7.5)]. Accordingly, this area constitutes 300 square pixels, which fits in nicely to the 300,000,000 in the United States. There is of course a Gaussian distribution amongst the races as evidenced with the Arab and Aborigine gentlemen [1]. However, thus far in our testing we have established that in the worst case scenario in the pseudo-first-sorts phase, using the most common length nose amongst Caucasians (-4.32 to -4.96) on the y-axis, it will narrow the search from 300,000,000 to about 44% or 132,000,000. This 132,000,000 then travel through the Innertron methodology consisting of 7 regions consisting of 16 blocks each = 1207 possible combinations. After sorting through the knowledge base and obtaining the closest matches, a printout is given, to humans worst case scenario is 300, (where its a Caucasians, middle of the Gaussian curve and has an entire region with 40,000 other likenesses) humans can see who is black, old, in jail, too young and so forth. The list will rank individuals based upon the search criteria.

References

1. Lewis, R.A., Ras, Z.W., *Innertron: New methodology of facial recognition, Part II*, in *Intelligent Information Processing and Web Mining*, Advances in Soft Computing, Proceedings of the IIS'2005 Symposium, Gdansk, Poland, Springer-Verlag
2. See: http://before.eyelids-blepharoplasty.com/sub_anatomy.html

Appendix 01

Radial Basis Function Network.

A Radial Basis Function Network (see Figure 3, right-top picture) is a technique for curve fitting or surface fitting. It uses a sample data set to

approximate the curve or surface and then interpolates remaining data. As demonstrated in the Neven methodology, researchers use it in pattern recognition problems. Some basic facts:

- Both "O" and "X" are the objects (sometimes referred to as "units" or "layers".
- "H" is the function.
- The first layer "O" or "Object" is a hidden layer. It is non-linear and constructed by a set of basis functions.
- The output layer "X" forms a weighted linear combination of the hidden units' activations.
- A typical Basis Function could be $\Phi_j(X) = exp\{-\frac{\|X-\tilde{\mu}_j\|^2}{2\tilde{\sigma}_j^2}\}$
- A typical algorithm for training a Radial Basis Function Network consists of two independent stages. In the fist "Object" or "Hidden Layer" one first determine, using input data such as K-means (EM algorithms) to perform an unsupervised learning method:
  - the basis functions centers $\tilde{\mu}_j$
  - their widths $\tilde{\sigma}_j$
- The output layer determines the weights and performs a supervised learning method.

If, for example, one has a goal to compute a function $F(x)$ that equals a particular result: $F(\vec{x_i}) = d_i$.

Then, $F(\vec{x_i}) = \sum_{i=1}^{N} w_i \cdot \phi(\| \vec{x} - \vec{x_i} \|)$, $1 \leq i \leq N$,

where ϕ defines a set of basis functions $\phi(\| \vec{x} - \vec{x_i} \|)$, $1 \leq i \leq N$.

Appendix 02

External Anatomy of the Eyelids and Eye

Figure 3, Picture 3:

A Medial Commissure: Inner corner where eyelids join.

B Lateral Commissure: Outer corner where eyelids join.

C Medial Canthus:Tissues just beyond medial commissure.

D Lateral Canthus: Tissues just beyond lateral commissure.

E Upper Eyelid Crease: Indentation or fold in upper eyelid.

F Lower Eyelid Margin: Edge of eyelid.

G Nasojugal Fold: Indentation extending from lid down along nose.

H Sclera: White layer of eyeball I Iris: Colored layer inside of eyeball.

J Pupil: Hole in iris that lets in light.

K Palpebral Fissure: (Not labelled) Opening between the eyelids.

Figure 3, Picture 4:

A Orbital Septum: Layer holding back orbital fat.
B Levator Aponeurosis: Eyelid muscle tendon seen through septum.

Figure 3, Picture 5:

A Orbicularis Muscle: (palpebral portion) Eyelid closing muscle.

B Orbicularis Muscle: (orbital portion) Eyelid closing muscle.

C Frontalis Muscle: Forehead muscle.

D Procerus Muscle: Muscle that lowers brows.

E Corrugator Muscle: Muscle that brings brows together.

F Midfacial Muscles: Muscles of the cheek.

G Malar Fat Pad: Large cheek fat pad.

H Suborbicularis: Fat (SOOF) Fat pad beneath orbicularis muscle.

I Temporalis Muscle and Fascia: Muscle of temple.

Approximating a Set of Approximate Inclusion Dependencies

Fabien De Marchi[1] and Jean-Marc Petit[2]

[1] LIRIS, FRE CNRS 2672, Univ. Lyon 1, 69622 Villeurbanne, France
[2] LIMOS, UMR CNRS 6158, Univ. Clermont-Ferrand II, 63177 Aubière, France

Abstract. Approximating a collection of patterns is a new and active area of research in data mining. The main motivation lies in two observations : the number of mined patterns is often too large to be useful for any end-users and user-defined input parameters of many data mining algorithms are most of the time almost arbitrary defined (e.g. the frequency threshold).

In this setting, we apply the results given in the seminal paper [11] for frequent sets to the problem of approximating a set of approximate inclusion dependencies with *k inclusion dependencies*. Using the fact that inclusion dependencies are "representable as sets", we point out how approximation schemes defined in [11] for frequent patterns also apply in our context. An heuristic solution is also proposed for this particular problem. Even if the quality of this approximation with respect to the best solution cannot be precisely defined, an interaction property between IND and FD may be used to justify this heuristic.

Some interesting perspectives of this work are pointed out from results obtained so far.

1 Introduction

In the relational model, inclusion dependencies (INDs) convey many information on the data structure and data semantics, and generalize in particular foreign keys [1].

Their utility is recognized for many tasks such as semantic query optimization [3], logical and physical database design and tuning [18,2,15,5] or data integration [21]. In practice, the implementation of these technics is generally made impossible by the ignorance of satisfied INDs in the database. One can then be interested in the problem of discovering INDs satisfied in a database [10].

Some algorithms have been recently proposed for this data-mining task [4,12,6]. They output either the whole set of satisfied INDs or only the largest satisfied INDs from which the whole collection of INDs can be inferred. Nevertheless, the interest of discovering only satisfied INDs may be limited in practice: indeed, algorithms being applied on real-life databases, many of them can be qualified as "dirty" databases since constraints may not have been defined. To take into account some data inconsistencies, algorithms have been extended to the discovery of *approximate inclusion dependencies* [4,6]. The idea is to define an error measure from which approximate INDs can be

rigorously defined with respect to an user-defined threshold. The proposed algorithms find exactly all INDs having an error measure lower than the threshold.

Moreover, approximating a collection of patterns is a new and active area of research in data mining. The motivating remarks done in [11] for approximating a collection of frequent itemsets apply for approximate INDs :

1. the output depends on an user-defined threshold. In general, it is almost arbitrary chosen by a user and therefore justifies the approximation of the output.
2. a complete result can be prohibitive in certain cases, since the potential number of satisfied INDs might be very large. For instance, large set of constraints are going to be useless for experts and/or database administrators in order to analyze/maintain the database.

In this paper, we apply the results given in the seminal paper [11] for frequent sets to the problem of approximating a set of approximate inclusion dependencies in a database. Using the fact that inclusion dependencies are "representable as sets" [20,6], we point out how approximation schemes defined in [11] for frequent patterns also apply in our context. An heuristic solution is also proposed for this particular problem. Even if the quality of this approximation with respect to the best solution cannot be precisely defined, an interaction property between IND and FD may be used to justify this heuristic.

Some interesting perspectives of this work are pointed out from results obtained so far.

This work is integrated in a more general project devoted to DBA maintenance and tuning, called "DBA Companion" [5]. We have proposed methods for discovering functional dependency and keys, inclusion dependencies and foreign key and for computing small database samples preserving functional and inclusion dependencies satisfied in the database. A prototype on top of Oracle RDBMS has been developed from these techniques [16].

Paper organization. Section 2 points out some preliminaries of this work. Section 3 gives the main result of this paper on the approximation of a set of approximate INDs. Related works are discussed in section 4. We conclude in section 5.

2 Preliminaries

We assume the reader is familiar with basic concepts from relational database theory (see e.g. [19,14]).

An *inclusion dependency* (IND) over a database schema $\mathbf{R}$ is a statement of the form $R_i[X] \subseteq R_j[Y]$, where $R_i, R_j \in \mathbf{R}, X \subseteq R_i, Y \subseteq R_j$ The size (or

arity) of an IND $i = R[X] \subseteq R[Y]$, noted $|i|$ is such that $|i| = |X| = |Y|$. We call *unary inclusion dependency* an IND of size 1.

An inclusion dependency is said to be *satisfied* in a database, if all values of the left hand-side appears in the right-hand side. If I_1 and I_2 are two sets of INDs, I_1 is a *cover* of I_2 if $I_1 \models I_2$ (this notation means that each dependency in I_2 holds in any database satisfying all the dependencies in I_1) and $I_2 \models I_1$.

To evaluate approximate INDs, an error measure called g'_3 has been defined in [17]:

$$g'_3(R[X] \subseteq S[Y], \mathbf{d}) = 1 - \frac{max\{|\pi_X(r')| \mid r' \subseteq r, (\mathbf{d} - \{r\}) \cup \{r'\} \models R[X] \subseteq S[Y]\}}{|\pi_X(r)|}$$

Intuitively, g'_3 is the proportion of distinct values one has to remove from X to obtain a database $\mathbf{d'}$ such that $\mathbf{d'} \models R[X] \subseteq S[Y]$. An INDs i is said to be approximately satisfied if $g'_3(i)$ is lower than a given threshold.

Given $i = R[X] \subseteq S[Y]$ and $j = R[X'] \subseteq S[Y']$ two INDs over a database schema $\mathbf{R}$, we say that i *generalizes* j (or j *specializes* i), denoted by $i \preceq j$, if $X' =< A_1, ..., A_n >$, $Y' =< B_1, ..., B_n >$, and there exists a set of index $k_1 < ... < k_l \in \{1, ..., n\}$ with $l \leq n$ such that $X =< A_{k_1}, ..., A_{k_l} >$ and $Y =< B_{k_1}, ..., B_{k_l} >$ [20,4]. For example, $R[AC] \subseteq S[DF] \preceq R[ABC] \subseteq S[DEF]$, but $R[AB] \subseteq S[DF] \not\preceq R[ABC] \subseteq S[DEF]$.

Let $\mathbf{d}$ be a database over $\mathbf{R}$, the approximate satisfaction of INDs is monotone with respect to $\preceq$ since $i \preceq j \Rightarrow g'_3(i) \leq g'_3(j)$ [6].

For an precise definition of "problem representable as sets", the reader is referred to the theoretical framework defined in [20]. In [6], we detailed how an isomorphism can be defined between a set of IND under the partial order $\preceq$ and a set of unary INDs under the partial order $\subseteq$. In other words, INDs are said to be *representable as sets*.

Basically, given an IND i, the representation as sets of i is defined by $f(i) = \{j | j \preceq i, |j| = 1\}$, the function f being bijective.

A set I of INDs is *downwards closed* if $\forall i \in I$, we have $j \preceq i \Rightarrow j \in I$. A downwards closed set I can be expressed by its *positive border* or border, denoted by $\mathcal{B}d^+(I)$, which is the set of the most specialized elements in I.

3 Approximating a set of INDs

The data mining problem underlying this work can be stated as follows:

> [`All-ApproxIND` problem] Given a database d and a user-specified threshold ϵ, find the set of approximate INDs i such that $g'_3(i) \leq \epsilon$.

Algorithms do exist to perform this task [4,6] and we assume that such a set of approximate INDs is available.

In this paper, we are only interested in the approximation problem of such a set, denoted by `Approx-All-ApproxIND`, and defined as follows:

[`Approx-All-ApproxIND` problem] Given a downwards closed set I of approximate INDs, find a set S of k INDs approximating I as well as possible.

We only consider as input the whole set I of INDs answering the `All-ApproxIND` problem. Note that the positive border of I could have been considered instead of I itself [11].

We extend the propositions made in [11] in our context: First, a natural way of representing a set of INDs is to consider k INDs from which all more general INDs are generated. Before going into much details, the following notations will be needed: Let i be an IND. The downwards closed set of i, denoted by $\rho(i)$, is defined by $\rho(i) = \{j | j \preceq i\}$.

Let $I_1, \ldots I_k$ be a set of k INDs from I. An approximation of I can be represented with $S(I_1, \ldots I_k)$ (or simply S whenever $I_1, \ldots I_k$ is clear from context) defined by

$$S(I_1, \ldots I_k) = \bigcup_{i=1}^{k} \rho(I_i)$$

$S(I_1, \ldots I_k)$ is said to be a *k-spanned set of INDs*.

Second, we have to decide how to define S with respect to I. Two main alternatives exist: either elements of S have to belong to I or can be chosen outside I. For the problem of approximating a set of approximate INDs, a natural choice seems to be the first alternative, i.e. $S \subseteq I$ or more accurately, $S \subseteq \mathcal{B}d^+(I)$. This choice is justified by the "global structure" of a set I answering the `All-ApproxIND` problem, leading from an interaction property between FD and IND. This point is discussed in much details in the following subsection.

Third, in order to define the approximation made between I and S, we borrow the *coverage measure* $C(S, I)$ [11] defined as the size of the intersection between S and I. Since $S \subseteq I$, maximizing the coverage measure is equivalent to maximazing the size of S.

Describing approximatively the downwards closed set I of approximate INDs with a set S of k INDs can be defined as follows:

- the size of S is equal to k, usually much smaller than the size of $\mathcal{B}d^+(I)$,
- for some $i \in I, \exists j \in S$ such that $i \preceq j$, i.e. S is a succinct representation of I,
- for some $i \in I$, $\nexists j \in S$ such that $i \preceq j$, i.e. some information is lost,

With the assumptions made, the problem `Approx-All-ApproxIND` can now be re-formulated as follows:

Given a downwards closed set I of approximate INDs, find a k-spanned set S of INDs such that $C(S, I)$ is maximized.

Theorem 1. *`Approx-All-ApproxIND` is NP-hard.*

Proof. This problem is shown to be NP-hard for frequent itemsets [11]. Since INDs are representable as sets of unary INDs, there is a bijective function between a set of INDs and a set of set of unary INDs. The transformation is clearly polynomial. Now, if we see unary INDs as items, and set of unary INDs as itemsets, any set of INDs can be transformed into a set of itemsets. In the same way, any set of itemsets can be transformed into a set of unary INDs. The `Approx-All-ApproxIND` problem is thus equivalent to the same problem for frequent itemsets.

Let I be a downwards closed set of approximate INDs and let S^* be an optimal solution for the problem `Approx-All-ApproxIND`.

Theorem 2. *A k-spanned set S of INDs can be found in polynomial time such that*

$$C(S, I) \geq (1 - \frac{1}{e})C(S^*, I)$$

Proof. Since INDs are isomorphic to the set of unary INDs, the problem can be posed in term of set theory. Within the framework of set theory, the problem turns out to be an instance of the Max k-Cover, from which a greedy algorithm does exist and provide a $(1 - \frac{1}{e})$ approximation ratio to Max k-Cover [9]. A solution can be computed with the greedy algorithm and translated back to INDs since the function is bijective.

To sum up, despite the NP-hardness result, polynomial-time approximation algorithms can be used effectively to answer our `Approx-All-ApproxIND` problem.

3.1 A heuristic

Since spanners of a set I of INDs are chosen within the border of I, one may be tempted to select the k first IND whose arity is larger in the border of I.

We now point out that, in practice, a set I of (approximatively) satisfied INDs has a particular structure. This consideration leads from a well-known interaction between INDs and functional dependencies, given in Table 1.

$$\left.\begin{array}{c} R[XY] \subseteq S[UV] \\ R[XZ] \subseteq S[UW] \\ S : U \rightarrow V \end{array}\right| \Rightarrow R[XYZ] \subseteq S[UVW]$$

Table 1. An interaction between FD and IND

Intuitively, consider two INDs i and j of large arity. Let U be the set of common attributes in the right-hand sides of i and j. The more large $|U|$ is, the more U is likely to be a left hand side of functional dependencies. In other words, a new IND is likely to be satisfied from the property given in Table 1, and i and j are not anymore in the border of I.

As already discussed, this reasoning has a very strong impact on the global structure of the downwards closed set I of INDs to be approximated. It cannot be compared with the global structure of a downwards closed set of frequent sets where no similar property (of that given in Table 1) does exist. We derive two points from this consideration:

- There is no need to define S outside I. Indeed, an IND $i \notin I$ would be interesting if it covers a great number of elements of I and only a few number of elements not in I. But we argue that, from property in Table 1, i has great chances to belong to I in this case.
- We suggest the following very simple heuristic:
  Let S_k be a subset of I made up of the k largest INDs of I
  Such a set is straightforward to compute from I and offers an approximate solution of I.

Hopelessly, the solution S_k cannot be compared with other solutions obtained so far. Nevertheless, due to the property 1, we conjecture that the solution S_k is also a good approximation of I.

Remark also that this heuristic gives the best solution, i.e. $S_k = S^*$, whenever the border of I is made up of "disjoint" INDs, i.e. no attribute appears in more than one IND of $\mathcal{B}d^+(I)$. Nevertheless, this case seems to be a very constrained one in practice.

Another justification comes from experiments performed on two data mining problems: IND discovery [6] and maximal frequent item set discovery [7]. From our experimental tests, we have studied the *global structure* by doing a quantitative analysis of the positive and the negative borders of the result. In all cases, the global structure was quite different, justifying both our choice to define S from I and our heuristic.

4 Related works

In [11], authors introduce a framework for approximating a collection of frequent sets by a set of k sets. They consider two cases for choosing their approximation: either from the collection, or *outside* the collection. Moreover, they extend their results by considering the positive border of their collection of frequent sets instead of the whole collection.

Another point of view for approximating frequent sets consists in computing the *top-k most frequent sets*, as proposed in [8] for frequent closed sets.

In the setting of approximate IND discovery, [13] proposed a set of heuristics. The basic idea is to improve the pruning by considering some criteria from which interestingness of INDs can be evaluated, for example using the number of distinct values. In [17], approximation is seen with respect to SQL workloads, i.e. join path expressions reveal "interesting" INDs. These approaches are rather empirical though.

5 Conclusion and perspectives

We studied in this paper approximating techniques of a set of approximate inclusion dependencies in the following setting: what are the k INDs that best approximate the whole set of INDs ? From results given in [11] for frequent itemsets, we focus on two of them : 1) the problem of finding the best approximation spanned by k sets is NP-hard and 2) an approximation can be computed in polynomial time within a factor of $(1 - \frac{1}{e})$ with respect to the best solution. We mainly show in this paper how these results may apply for approximating a set of INDs with k INDs. These results follow from the fact that INDs are "representable as sets", i.e. the search space of IND is isomorphic to a subset lattice of some finite set.
We proposed also an heuristic exploiting the global structure of a set of approximate INDs satisfied in a database. Since no formal comparison can be made with respect to the best solution as we did before, we plan to do in future work experiments in order to evaluate the quality of this heuristic.

An interesting corollary may be also easily deduced since any problem representable as sets can indeed re-used the propositions made in [11] as for example functional dependencies or learning boolean functions.

From a data mining point of view, algorithms for discovering approximation of discovered patterns from the data have to be designed, instead of discovering first the set of all patterns and then applying approximation schemes.

References

1. M. Casanova, R. Fagin, and C. Papadimitriou. Inclusion dependencies and their interaction with functional dependencies. *Journal of Computer and System Sciences*, 24(1):29–59, 1984.
2. M. A. Casanova, L. Tucherman, and A. L. Furtado. Enforcing inclusion dependencies and referencial integrity. In F. Bancilhon and D. J. DeWitt, editors, *International Conference on Very Large Data Bases (VLDB'88), Los Angeles, California, USA*, pages 38–49. Morgan Kaufmann, 1988.
3. Q. Cheng, J. Gryz, F. Koo, T. Y. Cliff Leung, L. Liu, X. Qian, and B. Schiefer. Implementation of two semantic query optimization techniques in DB2 universal database. In M. P. Atkinson, M. E. Orlowska, P. Valduriez, S. B. Zdonik, and M. L. Brodie, editors, *International Conference on Very Large Data Bases (VLDB'99), Edinburgh, Scotland, UK*, pages 687–698. Morgan Kaufmann, 1999.
4. F. De Marchi, S. Lopes, and J.-M. Petit. Efficient algorithms for mining inclusion dependencies. In C. S. Jensen, K. G. Jeffery, J. Pokorný, S. Saltenis, E. Bertino, K. Böhm, and M. Jarke, editors, *International Conference on Extending Database Technology (EDBT'02), Prague, Czech Republic*, volume 2287 of *Lecture Notes in Computer Science*, pages 464–476. Springer, 2002.
5. F. De Marchi, S. Lopes, J.-M. Petit, and F. Toumani. Analysis of existing databases at the logical level: the DBA companion project. *ACM Sigmod Record*, 32(1):47–52, 2003.

6. F. De Marchi and J-M. Petit. Zigzag : a new algorithm for discovering large inclusion dependencies in relational databases. In *International Conference on Data Mining (ICDM'03), Melbourne, Florida, USA*, pages 27–34. IEEE Computer Society, 2003.
7. F. Flouvat, F. De Marchi, and J-M. Petit. Abs: Adaptive borders search of frequent itemsets. In *FIMI'04*, 2004.
8. J. Han, J. Wang, Y. Lu, and P. Tzvetkov. Mining top-k frequent closed patterns without minimum support. In *International Conference on Data Mining (ICDM 2002), Maebashi City, Japan*, pages 211–218. IEEE Computer Society, 2002.
9. Dorit Hochbaum. *Approximation algorithms for NP-hard problems*. PWS Publishing Compagny, 1997.
10. M. Kantola, H. Mannila, K. J. Räihä, and H. Siirtola. Discovering functional and inclusion dependencies in relational databases. *International Journal of Intelligent Systems*, 7:591–607, 1992.
11. W. Kim, R. Kohavi, J. Gehrke, and W. DuMouchel, editors. *Approximating a collection of frequent sets*. ACM, 2004.
12. A. Koeller and E. A. Rundensteiner. Discovery of high-dimentional inclusion dependencies (poster). In *Poster session of International Conference on Data Engineering (ICDE'03)*. IEEE Computer Society, 2003.
13. A. Koeller and E. A. Rundensteiner. Heuristic strategies for inclusion dependency discovery. In R. Meersman and Z. Tari, editors, *CoopIS, DOA,and ODBASE, OTM Confederated International Conferences, Napa, Cyprus, Part II*, volume 3291 of *Lecture Notes in Computer Science*, pages 891–908. Springer, 2004.
14. M. Levene and G. Loizou. *A Guided Tour of Relational Databases and Beyond*. Springer, 1999.
15. M. Levene and M. W. Vincent. Justification for inclusion dependency normal form. *IEEE Transactions on Knowledge and Data Engineering*, 12(2):281–291, 2000.
16. S. Lopes, F. De Marchi, and J.-M. Petit. DBA Companion: A tool for logical database tuning. In *Demo session of International Conference on Data Engineering (ICDE'04)*, `http://www.isima.fr/~demarchi/dbacomp/`, 2004. IEEE Computer Society.
17. S. Lopes, J.-M. Petit, and F. Toumani. Discovering interesting inclusion dependencies: Application to logical database tuning. *Information System*, 17(1):1–19, 2002.
18. H. Mannila and K.-J. Räihä. Inclusion dependencies in database design. In *International Conference on Data Engineering (ICDE'86), Los Angeles, California, USA*, pages 713–718. IEEE Computer Society, 1986.
19. H. Mannila and K. J. Räihä. *The Design of Relational Databases*. Addison-Wesley, second edition, 1994.
20. H. Mannila and H. Toivonen. Levelwise Search and Borders of Theories in Knowledge Discovery. *Data Mining and Knowledge Discovery*, 1(1):241–258, 1997.
21. R. J. Miller, M. A. Hernández, L. M. Haas, L. Yan, C. T. H. Ho, R. Fagin, and L. Popa. The clio project: Managing heterogeneity. *ACM SIGMOD Record*, 30(1):78–83, 2001.

Part XI

Invited Session: Recent Developments in Bioinformatics

Informatic Approaches to Molecular Translation

David H. Ardell

Linnaeus Centre for Bioinformatics, Box 598, 751 24 Uppsala, Sweden

Abstract. Recent results on the evolution of the genetic code are informally put in the context of Von Neumann's theory of self-reproducing automata and the classical treatment of error-correcting codes in information theory. I discuss the sufficiency of genetic descriptions for self-reproduction. This implies a duality, a division of the information for maintenance and reproduction of organisms into two components, which is fundamental to life as we know it. The two components are programmatic and operational in nature, and they require the transmission and transduction of information to cooperate. In this context, I review recent results on the coevolution of genes and genetic codes. These results show how desirable informatic properties can evolve in systems of tapes and tape-players that are mutually co-dependent for their reproductive success.

1 Foreword

This paper serves as a loose and informal sketch of ideas putting my studies of a few years ago on the origin of the genetic code in the contexts of Von Neumann's work on the theory of self-reproducing automata on one hand, and on information and coding theory on the other.

The article assumes of the reader a fairly detailed account of the molecular biology of translation and replication. Suitable starting points to attain this are [12]and [16]. Some further tips to biological literature for those wishing to follow-up certain points are provided along the way.

2 The Sufficiency of Programmatic-Operational Duality for Self-Reproduction

The solution of the structure of DNA [19] solved an outstanding puzzle of its day: how information might be encoded in a biological macromolecule as the physical basis of heredity. Watson and Crick wrote coyly,

> It has not escaped our notice that the specific pairing we have postulated immediately suggests a possible copying mechanism for the genetic material.

Some particular and defining aspects of DNA as hereditary medium are its linearity, its composition from discrete components, and its specialization

as such, in that it does not participate in metabolic activity nor any other besides the storage of information. For an interesting account of the history of experimental molecular biology leading up to and following the solution of the structure of DNA, see [13].

The polymath John Von Neumann was credited quite early [14] for having anticipated Watson and Crick's celebrated discovery by several years; that is, he is credited for having intuited and proven the advantages and sufficiency for an organism in self-reproduction to possess or contain a relatively static ("quiescent") description of itself. There were some partial precedents to Von Neumann's insight. Francois Jacob [8] (one of the authors of the operon theory of gene regulation), in his history of heredity, credits Buffon for suggesting in the latter half of the eighteenth century that something like "internal moulds", analogous to those used by a sculptor may be the means by which organisms take form, noting that it is easier to copy and transfer information arrayed in lower than three dimensions. August Weissmann, at the end of the nineteenth century, is credited for experimentally establishing that inheritance occurs through a specialized cell lineage only, the other cells not contributing in any way to what is passed on, and for having considered some of the theoretical implications of this fact (see list below).

Yet nobody before Von Neumann was able to so clearly formulate and answer the question: "can any automaton construct other automata that are exactly like it?" [17,18]. He showed that it was sufficient to split a self-reproducing automaton E into two primary components: an active and programmable constructing and copying component, $D = (A + B + C)$ and a quiescent descriptive component $L_D = \phi(A + B + C)$, where $\phi(X)$ is a function that maps a constructible object X to a discrete representation of an algorithm for constructing X from a finite set of axiomatic operations. In self-replication, the following reactions occur (leaving out details necessary for mass and energy balance):

1. A control component C of the automaton instructs a specialized copying component B to make a copy of its description L_D:

$$B + L_D \rightarrow B + 2L_D. \tag{1}$$

2. The control component then instructs the programmable constructor A to construct a new automaton from a large pool of fundamental components using the description L_D:

$$A + L_D \rightarrow A + L_D + D. \tag{2}$$

3. Finally, the control component instructs the constructor A to allocate the description copy to its daughter

$$A + D + L_D \rightarrow A + E. \tag{3}$$

In this way the automaton E reproduces itself:

$$E \rightarrow 2E. \tag{4}$$

Crucial for the remainder of our arguments is that there is a duality between the effective component, which in biological terms could be called the "soma" and the quiescent description, called the "germ-line." In this duality the germ-line plays a dual role: both as a source of programmatic information and as a passive source of data for copying. The description represents, and instructs the creation of, the machines that copy and express it. I assert that living information flows are almost always representational (if not self-representational) and therefore always consist of a duality between a mediated signal and a machine that must transduce the information in that signal.

To my knowledge, the aforementioned *Scientific American* article to the contrary, neither Von Neumann nor anyone else ever proved that this duality was necessary for self-reproduction, that is whether genetic descriptions are a requirement for self-reproduction. An organism could also use itself directly as the source of information necessary to construct itself. Yet Von Neumann and others make some compelling arguments about the advantages of genetic descriptions for self-reproduction and other powers:

Homogeneity of components: the information has its basis in a small number of like components. Thus the problem of replication (the reproduction of information) can be simplified tremendously.

Defined state: the whole concept of an active, dynamic entity directly using itself as a source of information for reproduction is complicated by the fact that the stimulation associated with observation and necessary for copying may change the state of other components of the automaton. At best this would reduce the fidelity of self-reproduction and at worst could lead to undefined conditions. This shows clearly why Von Neumann emphasized "quiescence" as an aspect of the description.

Consistency: Von Neumann constructs self-reproduction in the stronger context of universal construction. He suggests that it is unlikely that a machine with the power of universal construction can be programmed to obtain descriptions of itself directly from itself without entering logical fallacy.

Completeness: By containing an independently copied representation of itself, the automaton avoids representing its own representation *ad infinitum* to maintain this representation for its offspring.

Generalizability: Von Neumann constructs his scheme for self-reproduction in a context that is immediately generalizable to the reproduction of a more general automaton, itself plus a generalized component, and this is demonstrated through the use of the description. In this way Von Neumann argued that quiescent descriptions allow self-reproducing automata

to construct automata more complex themselves, something machines generally cannot do.

Evolvability: Variations in quiescent descriptions can occur without altering the integrity of the automaton itself, facilitating the generation of variation to be acted on by natural selection.

These properties are strikingly reminiscent of design principles of programs and programming languages.

3 From Central Dogma to Dynamic System

The widely familiar "Central Dogma" is thus incomplete as a representation of the flow of information in life. One missing component is clear in light of the central point of Von Neumann's answer to the paradox of self-reproduction. Namely, one must remember that it is the indirectly inherited soma component of a living organism that causes the replication of DNA and the expression of genes into proteins. DNA does not catalyze its own replication and expression. A further missing component is natural selection, which is an information flow from the combined soma plus germ-line to its reproductive success, in a specific population and environmental context.

By no means is this an indictment of Francis Crick. There is good historical evidence that he never intended the Central Dogma to become dogmatic. Despite the clarity of the Central Dogma for learning the basic facts of molecular biology, its essential message that information only flows from DNA to protein is contradictable, true only in the limited sense of template-dependent transfer of information. For instance, even with respect to inheritance we now understand the increasing importance of epigenetic (*lit.*"outside the gene") factors for development and evolution (see e.g. [9] and [10] for a recent demonstration of the importance of epigenetic factors for successful cloning).

Yet today we live in a "blueprint" oriented era, one of genetic engineering, cloning, and genomics. Furthermore, the discovery that RNA can perform catalytic functions (reviewed in [5]) has made the "RNA world" the dominant paradigm today for the origin of life, since both components of the germ-soma duality can be mediated by one and the same class of molecules. A division of labor and specialization of the genetic and catalytic functions into molecules better suited to either (DNA and proteins, respectively) is considered an advantageous later development (see e.g. [7]).

In contradiction to this dominating RNA-oriented paradigm for the origin of life, the physicist Freeman Dyson put forward the interesting argument (before the discovery of catalytic RNA) that polynucleic acid-based replication systems could not have originated without preexisting support from a catalytic metabolism. He uses Von Neumann's description of the dual components of self-reproducing automaton without embracing the central message that self-reproduction without a genetic component has certain disadvantages. Dyson suggests that an autocatalytic, protein-based self-reproducing

soma (metabolism) had to arise first, which later symbiosed with a replicating genetic system replication machinery later linked by the advent of translation. Thus, he postulates that the duality that is potentially implicit in self-reproduction is also implicit in a dual origin of life [6]. Dyson usefully distinguishes "reproduction" (such as the doubling and division of a cell) from "replication" (genetic copying), where the latter is more precise in duplication of information than the former. Dyson establishes that the autocatalytic reproduction of a metabolism does need to be as precise as genetic replication because an individual metabolism is a population of molecules, and the "metabolism of a population depends only on the catalytic activity of a majority of the molecules."

To sum this section up, even the most basic problems in biology require consideration of two more information flows than the Central Dogma suggests. In addition to template-dependent gene expression and copying, we have the programmatic flow of the machines that carry out these operations influencing among other things the fidelity of these operations. An third flow of information we have not yet discussed is the evolutionary flow of information in populations that translates the performance of a combined gene-soma system into reproductive success (specifically, frequencies or probabilities of inheritance).

4 Coevolution of Tapes and Tape-Players

If the duality of genetic description and expressing metabolism is an essential aspect of self-reproduction for life on earth, if not in general, this creates a clear motivation to understand how such systems evolve and change in general. It is important to realize that this intrinsic duality implies that the information that makes up an organism is divided into two components: its genetic description and the part of the soma that acts and expresses this description. The genetic description is useless without a somatic machine to interpret it. Of course, the expressing soma is itself encoded in part of the description. But this means that one part of the description is essential for the realization of every other part. This is a kind of genetic mutual dependency — a kind of epistasis. Epistasis, depending on the nature of the dependency, can constrain evolution, by requiring change among interacting parts to be coordinated in order to maintain compatibility.

The genetic description is a representation. There are infinitely many ways to perfectly encode and decode a representation, but not all are equivalent with regard to time and memory complexity, error-resistance, energy costs, flexibility, source validation and other performance characteristics. However, one only has to look at economics in today's media age to understand that once a media-based system becomes established it is hard to change, because valuable information accumulates in particular formats.

An anecdotal example comes from the videotape industry. As videotape technology began to be broadly adopted by consumers, they had a choice of two mutually incompatible formats: Sony's proprietary Betamax format and the VHS format. Despite the smaller form factor of Betamax and its allegedly superior qualities for video reproduction, VHS was widely adopted while Betamax all but completely disappeared. The cause was that the burgeoning video rental market was flooded, or perceived to be flooded, with movies in the VHS format. Consumers adopted the technology that they perceived would give them access to the most content. This fed forward to increased economic incentive to produce content in that format alone. Examples abound today of companies in different industries that seek to lock consumers into branded, copyrighted and patented technology on the basis of format compatibility with existing content.

In this context it is interesting to ask how certain desirable performance characteristics can evolve anew within a system constrained to preserve a specific format. We say that such systems are systems of tapes and tape-players, and we ask, what changes can such systems endure while still preserving the information encoded within them?

5 DNA and RNA as Tapes and the Genetic Code as Tape-Player

In the following we give two examples of how certain improvements to a dual information system of a tape and a tape-player can evolve without disrupting the information embodied in them. Both examples stem from fairly recent work modeling the coevolution of genes (the tapes) and the genetic code (part of the tape-player).

5.1 The Origin of Redundancy in the Genetic Code

The genetic code is a term often used in the popular press to mean the genetic description of an entire organism: the genome. Here I mean by genetic code the mapping of codons to amino acids in translation. There are 61 sense codons encoding amino acids and only 20 canonical amino acids, so we say the genetic code exhibits "redundancy". We have found that the ultimate level of redundancy in the genetic code, measured in the number of distinct amino acids or in the chemical range of physicochemical properties they possess, is among other things a function of the historical rates of mutation in genes [3].

A more general way to look at a redundancy in a genetic code is its "expressivity, " a term sometimes informally applied to programming languages. Informally defined, "expressivity" means the extent, dimensionality and fine-grainedness of meanings that are potentially expressed with a code. Redundancy and expressivity can be traded-off in favor of one or the other: increased expressivity increases the range, kind and precision with which a

meaning can be expressed. However, increased expressivity costs coding capacity that can otherwise be allocated to increase the robustness of a code to error. Furthermore, errors can be more catastrophic when they occur because a larger expressivity increases the potential contrast between an encoded and an expressed meaning (this implies a metric space on meanings; see next section).

What we found in this study was a way in which tapes and tape-players can coevolve from an initial state of total redundancy to a final state of expressivity without disrupting the meanings of the tapes along the way. The level of expressivity of final codes was always less than maximal and depended on parameters such as mutation rate that determined the overall level of error in the system. This is to my knowledge the first model to show how errors in the reproduction of tapes determines the final level of redundancy in the tape-players. A flip-side of our results were that expressivity increased in our system until a point at which no more new meanings could be encoded.

5.2 The Error-Mitigating Genetic Code

In a subsequent article I suggested that the genetic code can be modeled from the perspective of Hamming codes (so-called sphere-packing codes), more precisely, the minimum Hamming-distance decoding scheme without parity check-bits. A cartoon description of this is shown on the left in figure 1. In thw present note I wish to be more precise and specific about this comparison and introduce the concept of an "error-mitigating" code; I leave a more formal treatment, however, for later.

In a classical treatment of error-correcting codes, the Fundamental Theorem of Information Theory is concerned with proving that, under certain conditions, an arbitrarily small probability of decoding error may be achieved in the context of noisy transmission of information along a channel (see e.g. [4]). This could add one more potential advantage of self-reproduction by genetic description to the list above: arbitrarily small probabilities of decoding error can be achieved at a fixed rate of transmission.

Yet this advantage is not realized in the genetic code. We see that missense errors in translation occur at fairly high rates [15]. The translation system appears not to have evolved as if substituting one amino acid for another in proteins is tantamount to poison, to be avoided at any cost. Rather, in *E. coli*, hyperaccurate mutants have a growth-rate disadvantage [11]. Codons are not embedded in a larger sequence space, with "check bases" to enable the correction of errors in translation (I postulate that it is doubtful whether an error-correction scheme of this type could work at all to correct mutations from replication, because in mutation-selection balance mutant codons would accumulate at the boundaries of the spheres).

Rather, the genetic code allocates physicochemically similar amino acids to the more error-prone dimensions of the translational channel (see e.g. [1]. That is, the rates of translational error are different in the three different

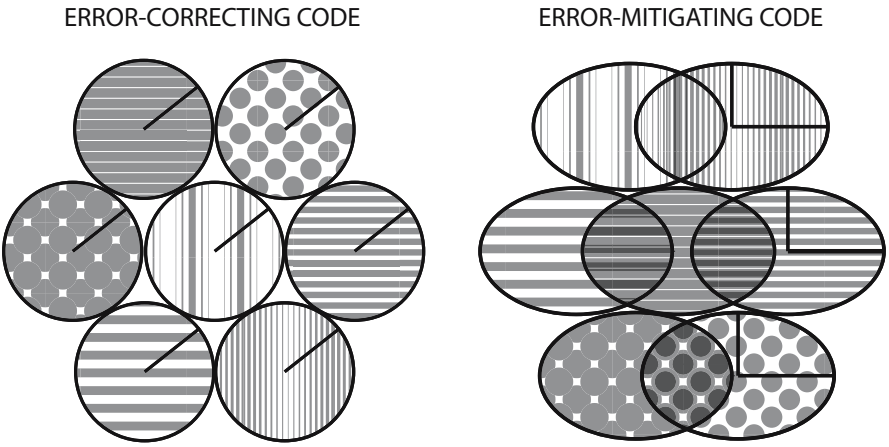


Fig. 1. Cartoon illustrations of error-correcting and error-mitigating codes. Patterns indicate meanings of the codewords after decoding. See text for more details

codon positions, and the physicochemical redundancy of encoded amino acids is greater in the more error-prone dimensions. In terms of coding theory, translation is an "asymmetric channel." I call this "anisotropy" and it is indicated by the squashing of the spheres into ellipsoids on the right side of figure 1.

In classical treatments of error-correcting codes, since absolute avoidance of error is the objective, no attention is given to modeling the meanings of the codewords, the objects that they represent. This is represented by the arbitrary placement of patterns on the left side of figure 1. It is a simple manner to introduce a metric space over meanings so as to be able to define a cost function for the substitution of one meaning when another is encoded. Then, rather then maximizing the rate of transmission and the absolute number of codewords while minimizing the absolute rate of error, one can maximize the rate of transmission and the expressivity (see discussion from previous section) while minimizing the cost of error. Here I call such a cost-minimizing code an "error-mitigating" code.

In [3] we showed how such an error-mitigating pattern may have evolved in the genetic code, from a totally redundant initial state. This is the first and only work that actually explains how this widely-noted pattern in the genetic code actually may have came about. Although I have framed the discussion above in terms of translational error, the evolution of error-mitigation has to be considered in the context of the coevolution of tape and tape-player for three reasons: first, as noted before translation is an asymmetric channel, and therefore its informatic properties depend on the distribution of symbols at the source. Second, because the source distribution is determined by natural selection, which is based on the combined performance of the decoding

channel with a specific source. Third and last, as previously noted, because the source distribution determines the fitness of changes in the code.

Why is the genetic code so small, in the sense of so few code-words? There has been theoretical work asking about rate-accuracy trade-offs in the size of codons, claiming that the current size of three is an optimal solution. This notwithstanding, perhaps Nature could not have picked a larger code to accommodate a larger number of potentially encodable amino acids or a greater tolerance to error since this would have been anticipatory evolution. The coevolution of genes and genetic codes has occurred within essentially immutable historical constraints, codon length may have been one of them.

6 Summary

I have sketched an argument that an essential aspect of self-reproduction in life, at least as we know it, is an intrinsic duality between an active soma and a quiescent genetic description or germ-line. The information that comprises an organism is split into these mutually dependent parts, like a video tape and tape-player that depend on a specific format. The realization of the information in the genetic description can be modeled by coding theory but the biological case has some properties that call for different assumptions and parameters of optimization than classical treatments. Finally, models for the coevolution of tapes and tape-players show how desirable properties of such dual systems can evolve despite the constraint of mutual dependency.

References

1. Ardell, D.H. (1998). On error-minimization in a sequential origin of the standard genetic code. J. Mol. Evol. **47**(1):1-13.
2. Ardell, D.H., Sella, G. (2001). On the evolution of redundancy in genetic codes. J. Mol. Evol. **53**(4/5):269-281.
3. Ardell, D.H., Sella, G. (2002). No accident: genetic codes freeze in error-correcting patterns of the standard genetic code. Phil. Trans. Roy. Soc. Lond. Ser. B **357**:1625-1642.
4. Ash, R.B. (1965) Information Theory. Dover, Mineola, NY, U.S.A.
5. Doudna J, Cech, T. (2002). The natural chemical repertoire of natural ribozymes. Nature 418:222-228.
6. Dyson, F. (1985) Origins of Life. Cambridge Univ. Press, U.K.
7. Gesteland, R.F., Cech, T.R. Atkins, J.F. (1999). The RNA world : the nature of modern RNA suggests a prebiotic RNA. 2nd. Ed. Cold Spring Harbor Lab. Press, New York, U.S.A.
8. Jacob, F. (1973). The Logic of Life: a History of Heredity. trans. Spillmann B.E. Princeton Univ. Press, Princeton, NJ, U.S.A.
9. Jablonka, E., Lamb, M. (1995). Epigenetic inheritance and evolution : the Lamarckian dimension. Oxford Univ. Press, Oxford, U.K.
10. Kono, T., Obata, Y., et al. (2004). Birth of parthenogenetic mice that can develop into adulthood. Nature **428**(6985):860-4.

11. Kurland, C.G., Hughes, D., Ehrenberg, M. (1996). "Limitations of translational accuracy." In Escherichia coli and Salmonella typhimurium: Cellular and molecular biology, F. C. Neidhardt, Curtiss, R. III, et al. ed., American Society for Microbiology Press, Washington, DC, U.S.A.
12. Lewin, B. (2004) Genes VIII. 8th. Ed. Oxford Univ. Press, Oxford, U.K.
13. Maaløe, O., Kjeldgaard, N.O. (1966). Control of Macromolecular Synthesis. W.A. Benjamin, New York Amsterdam.
14. Molecular Basis of Life: An Introduction to Molecular Biology. with introd. by Haynes R.H. and Hanawalt P.C. in series "Readings from Scientific American." W.H.Freeman, San Francisco, U.S.A.
15. Parker, J. (1989) Errors and alternatives in reading the universal genetic code. Microbiol. Rev. 53: 273-298.
16. Stryer, L., Berg J.M., Tymoczko, J.L. (2002). Biochemistry. 5th Ed. W.H. Freeman, New York, U.S.A.
17. Von Neumann, J. (1951). The General and Logical Theory of Automata. in J. Von Neumann, Collected Works, ed. Taub A.H., Vol. 5, MacMillan, New York.
18. Von Neumann, J. (1966). Theory of Self-Reproducing Automata. ed. Burks A.W. University of Illinois Press, Urbana London.
19. Watson, J.D., Crick, F.C. (1953). Genetical implications of the structure of deoxyribose nucleic acid. Nature, **171**:964.

An Approach to Mining Data with Continuous Decision Values

Hung Son Nguyen[1,2], Marta Łuksza[1,2], Ewa Mąkosa[1,2], and Henryk Jan Komorowski[1]

[1] The Linnaeus Centre for Bioinformatics, Uppsala University
Husargatan, 3, Uppsala, SE-751 24, Sweden.
[2] Faculty of Mathematics, Informatics and Mechanics, Warsaw University,
Banacha 2, 02-097, Warsaw, Poland

Abstract. We propose a novel approach to discover useful patterns from ill-defined decision tables with a real value decision and nominal conditional attributes. The proposed solution is based on a two-layered learning algorithm. In the first layer the preference relation between objects is approximated from the data. In the second layer the approximated preference relation is used to create three applications: (1) to learn a ranking order on a collection of combinations, (2) to predict the real decision value, (3) to optimize the process of searching for the combination with maximal decision.

1 Introduction

Mining data sets with continuous decision attribute is one of the most challenging problems in KDD research. The most well known approach to this problem is a regression method which constructs approximation of the decision attribute (unknown variable) with a function(linear, quadratics or other) of the conditional attributes (known variables).

Many data mining problems (e.g. in bioinformatics) force us to deal with ill-defined data, i.e., data sets with few objects but a large number of attributes. In addition, attributes may be of different types: some of them are nominal and the other continuous. The regression method can deliver a solution for the prediction task, but the output model is very hard to interpret and to draw conclusions. Moreover, in statistical terms, the small number of examples makes the regression model less significant. Furthermore, in many applications, the description task is even more important than the prediction task, since it helps discover patterns (rules) revealing the real mechanisms hidden behind the data.

In this paper we propose an alternative method designed to manage with these problems. Our method is based on a layered learning idea and decision rule techniques. Instead of searching for the direct description of decision attribute, we decompose this task into several learning subtasks. The first subtask is to approximate the preference relation between objects from the data. Using approximate preference relation we solve other subtasks such as,

for instance, learning ranking order, prediction of continuous decision value, or minimization of the cost of maximal decision value searching process.

2 Basic notions

We use the notion of decision table to describe the data. Intuitively, a decision table is a rectangular data table with rows containing description of objects. Usually columns are called attributes (or features or variables) among which one distinguished column is called decision.

Therefore, decision table consists of a collection of condition attributes $A = \{a_1, ...a_k\}$ and one decision attribute *dec*. All of them are determined on a finite set of objects U. Formally, decision table (see [2]) is a pair $\mathbb{S} = (U, A \cup \{dec\})$, where U is a non-empty, finite set of *objects* and A is a non-empty, finite set, of *attributes*. Each $a \in A \cup \{dec\}$ corresponds to a function $a : U \to V_a$, where V_a is called the *value set* of a.

In this paper we consider a special type of decision tables with nominal condition attributes and continuous decision attribute. This kind of decision tables is hard to deal with and has proven to be too difficult for many data mining methods. For example, regression methods requires conversion of nominal attributes in to continuous ones, while rule-based methods require a discretization of the decision attribute.

3 A rule-based approach to continuous decision

Layered learning is a well known idea in machine learning study and has many successful applications, see [4]. The main principle of layer learning is based on a decomposition of the complex learning task into subtasks and construction of their solutions. Usually, these subtasks are located in a hierarchical tree, and solutions (outputs) of subtasks in the lower layers are used to resolve the subtasks in the higher layers.

The proposed solution to the problem of data with continuous decision is based on layered learning. Instead of searching for a direct description of the decision attribute, we decompose this task into several learning subtasks.

3.1 Learning the preference relation

Let a decision table $\mathbb{S} = (U, A \cup \{dec\})$ with continuous decision, i.e., $V_{dec} = \mathbb{R}^+$, be given.

As it is in the standard classification problem setting, the value of the decision attribute *dec* is determined only on a subset of the universe $\mathcal{X}$ of all possible condition attribute values combinations.

We define a parameterized relation $PREF_\varepsilon \in \mathcal{X} \times \mathcal{X}$ as follows:

$$(x, y) \in PREF_\varepsilon \Leftrightarrow dec(x) - dec(y) > \varepsilon$$

where ε is called a tolerance parameter. We call the relation $PREF_\varepsilon$ a *preference relation*, since $x\ PREF_\varepsilon\ y$ means x is more preferred than y. Let us define a function $\theta_\varepsilon : \mathcal{X} \times \mathcal{X} \to \{-1, 0, 1\}$ as follows:

$$\theta_\varepsilon(x,y) = \begin{cases} 1 & \text{if } dec(x) - dec(y) > \varepsilon \\ 0 & \text{if } |dec(x) - dec(y)| \leq \varepsilon \\ -1 & \text{if } dec(x) - dec(y) < -\varepsilon \end{cases}$$

Then the preference relation $PREF_\varepsilon$ can be defined by the function θ_ε by

$$(x,y) \in PREF_\varepsilon \Leftrightarrow \theta_\varepsilon(x,y) = 1$$

Our learning algorithm for preference relation is performed on a new decision table $\mathbb{S}^* = (U^*, A^* \cup \{d^*\})$. This new table, called *the difference table*, is constructed from the given decision table $\mathbb{S} = (U, \{a_1, ..., a_k, dec\})$ as follows:

$$\begin{aligned} U^* =& U \times U = \{(x,y) : x, y \in U\} \\ A^* =& \{a_j^* : U \times U \to V_{a_j} \times V_{a_j} \text{ for } j = 1, ...k\} \\ & a_j^*(x,y) = (a_j(x), a_j(y)) \text{ for any pair of objects } x, y \in U \\ & d^* : U \times U \to \{-1, 0, 1\} \\ & d^*(x,y) = \theta_\varepsilon(x,y) \text{ for any pair of objects } x, y \in U \end{aligned}$$

One can apply any learning algorithm to the difference table. Our experiments were done by using decision rule (based on rough set algorithms), Naive Bayes algorithm, nearest neighbors, decision tree, and boosting algorithms for nearest neighbors and decision tree. The results are obtained by using Rosseta [1] and WEKA [5] systems.

If the original decision table contains n objects, then the difference table will have n^2 objects. We considered this transformation a way of dealing with the small sized input sets. Using the same learning algorithm, one can expect a higher statistical significance of results obtained from difference table than of those obtained by the original decision table.

Another very interesting problem is how to evaluate the quality of the preference learning algorithm. Based on the difference table, every learning algorithm constructs a classification algorithm, called a classifier. Usually, classifiers are more general than the function d^*, i.e., they are determined also for those pairs (x, y) which not necessarily belong to $U \times U$. In our consideration, such classifiers can be treated as approximations of the preference relation $PREF_\varepsilon$.

Let us denote by $\pi_{\mathbf{L},U}$ the classifier extracted from difference table $\mathbb{S}^*$ by using learning algorithm $\mathbf{L}$. Such classifier is a function

$$\pi_{\mathbf{L},U} : \mathcal{X} \times \mathcal{X} \to \{-1, 0, 1, unknown\}$$

determined for any pair of objects from $\mathcal{X}$.

Let $t \in \mathcal{X}$ be a test case, the accuracy of learning algorithm $\mathbf{L}$ on the object t is defined by

$$accuracy_{\mathbf{L},U}(t) = \frac{|\{u \in U : \pi_{\mathbf{L}}(u,t) = \theta_{\varepsilon}(u,t)\}|}{|U|}$$

The accuracy of the learning algorithm $\mathbf{L}$ on the test set $V \subset \mathcal{X}$ is computed as an average accuracy on test objects:

$$accuracy_{\mathbf{L},U}(V) = \frac{1}{|V|} \sum_{t \in V} accuracy_{\mathbf{L},U}(t)$$

3.2 Ranking learning

Ranking learning can be understood as a problem of reconstruction of the correct ranking list of a set of objects. In this section we present a simple algorithm for reconstruction of a ranking list using approximated preference relation, which has been described in the previous section.

Let us assume that $\mathbb{S} = (U, A \cup \{dec\})$ is a training data set and $(u_1, ..., u_n)$ is an ordered sequence of objects from U according to dec, i.e.,

$$dec(u_1) \leq dec(u_2) \leq ... \leq dec(u_n).$$

The problem is to reconstruct the ranking list of objects from a test data set $\mathbb{S}' = (V, A \cup \{dec\})$ without using decision attribute dec.

Our algorithm is based on the round robin tournament system which is carried out on the set of objects $U \cup V$. Similarly to football leagues, every object from V playing the tournament obtains a total score which summarizes its all played games. The objects from V are sorted with respect to their scores.

Assume that we have applied the learning algorithm $\mathbf{L}$ on the difference table constructed from training data set $\mathbb{S} = (U, A \cup \{dec\})$, and let $\pi_{\mathbf{L},U}$ be the output classifier of this algorithm. The classifier can be treated as a referee in the match between two objects in our tournament system. The *total score* of an object $x \in V$ is computed by

$$Score(x) = \sum_{y \in U \cup V} w(y) \cdot \pi_{\mathbf{L},U}(x,y)$$

where $w(y)$ is a weighting parameter that measures the importance of the object y in our ranking algorithm. We propose to define those weights by

$$w(y) = \begin{cases} 1 & \text{if } y \text{ is a test object, i.e., } y \in V; \\ 1 + \frac{i}{n} & \text{if } y = u_i \in U. \end{cases}$$

This ranking algorithm gives a higher score to those objects that recorded most victories over high valued objects.

The algorithm can be applied for all the objects from $U \cup V$. In such situation we say that objects from V are *embedded* in the ordered sequence $(u_1, u_2, ..., u_n)$.

To measure the quality of a ranking algorithm it is necessary to have a method to express the agreement between two ranking lists (one original and one computed by our algorithm) for the same set of objects. There are several well known "compatibility tests" for this problem, e.g., Spearman R, Kendall τ, or Gamma coefficients, see [3]. If the proper ranking list of V is denoted by $X = (x_1, x_2..., x_k)$, then the ranking list obtained from our algorithm can be treated as a permutation of elements of V, and represented by $Y = (x_{\sigma(1)}, x_{\sigma(2)}, ..., x_{\sigma(k)})$, for some permutation $\sigma : \{1, ..., k\} \to \{1, ..., k\}$. The Spearman coefficient is computed by using the formula:

$$R = 1 - \frac{6 \sum_{i=1}^{k} (\sigma(i) - i)^2}{k(k-1)(k+1)} \tag{1}$$

The Spearman coefficient can take any value of the interval $[-1; 1]$.

3.3 Prediction Problem

In this section we present a decision prediction algorithm which uses the approximate preference relation and ranking learning algorithm as components.

Let the training set of objects $U = \{u_1, ...u_n\}$ be given. The prediction algorithm computes the decision value of the test object $x \notin U$ in two steps:

algorithm 1 Prediction algorithm

Input: The set of labeled objects U and unlabeled object x;
parameters: learning algorithm $\mathbf{L}$;

Output: A predicted decision for x;

1: Embed the object x into the sequence $(u_1, u_2, ..., u_n)$ by applying ranking algorithm for objects from $\{x\} \cup U$ using $\mathbf{L}$ and decision table for U;
2: Let us assume that x is embedded between u_i and u_{i+1};
3: Return $prediction(x) = \frac{dec(u_i) + dec(u_{i+1})}{2}$ as a result of prediction.

The error rate of the prediction algorithm on the set of testing objects is measured by

$$error(V) = \frac{1}{card(V)} \sum_{x \in V} |dec(x) - prediction(x)|$$

It is obvious that the smaller the error rate, the better the prediction algorithm is.

3.4 Dynamic ranking algorithm

The quality of ranking algorithm can be low due to the small number of objects. In many applications the number of training objects is increasing in time, but it is connected with certain cost of examination. As an example may serve biological experiment data, where every training object needs to be confirmed in expensive laboratory tests.

In this section we treat a ranking problem as an optimization problem, in which we wish to get the highest value element using as low as possible the number of requests, i.e., to minimize the number of examinations and the cost of the whole process.

We propose another ranking method based on active learning approach. Instead of using a randomly selected test set, the learning algorithm has access to the set of unlabeled objects and can request the labels for some of them. In the biological research example, unlabeled objects correspond to the samples that were not examined yet; a request for a label responds to performing single laboratory test that gives the decision value of the requested sample. In the *dynamic ranking algorithm*, the unlabeled objects are requested according to the actual ranking list. Then, after obtaining the labels of new objects, the ranking list will be corrected.

Algorithm 2 presents the main framework of our idea. The notion of STOP CONDITION is defined differently for each optimization problem and refers to cost limits for the considered process.

algorithm 2 The dynamic ranking algorithm

Input: The set of labeled objects U and unlabeled objects V;
parameters: learning algorithm **L** and positive integer *request_size*;
Output: A list of objects to be requested; Ranking of elements in the U_2 in the *RankList*;

```
1: U1 ← U; U2 ← V;
2: RankList ← [ ];   //the empty list
3: while not STOP CONDITION do
4:    Rank elements of U2 by using L and decision table for U1; Let this ranking
      list be: (x1, x2, ...);
5:    for i = 1 to request_size do
6:       RankList.append(xi)
7:       U1 ← U1 ∪ {xi};   U2 ← U2 \ {xi};
8:    end for
9: end while
```

4 Experimental results

Our method was tested on an artificially generated decision table. The table consisted of six nominal condition attributes of values from the set $\{1, 2\}$ and

a real decision attribute. As each of the condition attributes had two possible values, the whole space of objects was of size $2^6 = 64$. Values of the decision attribute were calculated from the function:

$$dec = e^{a_1{}^{a_2}} + (a_1 + a_2 + a_3 + a_4 + a_5) * a_6/a_3 + \sin(a_4) + \ln(a_5) + noise$$

where $a_1, a_2, a_3, a_4, a_5, a_6$ are the attributes' values, and *noise* is a random value from the interval $[-1, 1]$ obtained using the uniform distribution on this interval. All the resulting decision values were in the interval $[5.772, 32.732]$. The tolerance parameter ε was set to 0.7.

Several different algorithms used in the model training step were compared in the tests:

- Decision rule method based on the rough set approach (see [1]) with the SAVgenetic reducer. In the result tables referred to as "Decision Rules".
- Naive Bayes algorithm from the Rosetta system (referred to as "Naive bayes")
- Nearest neighbor like algorithm using non-nested generalized exemplars, implemented in the WEKA system. It is referred to as "Nnge" in Table 1.
- Multi-boosting of the Nnge algorithm in the WEKA system. Multi-boosting is an extension to the AdaBoost technique for forming decision committees. Multi-boosting can be viewed as combining AdaBoost with wagging. It is designed to harness both AdaBoost's high bias and variance reduction with wagging's variance reduction. For more information, see [6]. In the result tables referred to as "MultiBoosting Nnge".
- C4.5 decision tree, impemented in WEKA. For more information, see [7]. Referred to as "C4.5".
- Multi boosting of C4.5 in the WEKA system. ("MultiBoosting C4.5")

We tested the quality of the rankings obtained by computing the Spearman coefficient of the correlation with the true ranking using Equation 1. In every iteration of experiment, the original data set was partitioned with proportion (50%:50%) into the training set U and test set V. The mean Spearman coefficient value and mean model accuracy for every learning algorithm are computed as average results over 10 iterations of experiments and reported in Table 1.

To test the real decision value prediction accuracy there were 7-fold cross validation tests performed on the original data set. The mean absolute error is reported.

The quality of the dynamic ranking algorithm is measured by the position of the best candidate in the resulting ranking list. The lower the resulting position, the better. Random shuffle of elements would result in expected position of this element in the middle index. The original data set was split in half to form the train data set U and the test data set V. Table 1 shows the positions of the best candidates in the ranking list constructed by our

approach. The experiment with every learning algorithm was repeated 10 times and we return the average position of the best candidate within those 10 repetitions. The tests were performed with *request_size* value set to 2. In order to compare the dynamic ranking approach with the static one, we calculated the Spearman coefficient.

Table 1. Experiment results achieved using six different learning algorithms. The first column reports the results ranking quality tests: Spearman coefficient and model accuracy on change table; second column reports the Pearson correlation coefficient and prediction mean absolute error; third column summarizes the results obtained while testing the dynamic ranking algorithm, average position and final ranking Spearman coefficient.

Learning	Ranking		Prediction		Dynamic Ranking	
algorithm	Spearman	acc.(%)	Pearson	pred.error	pos.	Spearman coef.
Decision Rules	0.893	83.28%	0.9653	1.4547	1.3	0.9501
Naive Bayes	0.7984	78.52%	0.5948	3.8336	1.3	0.8540
Nnge	0.777	77.19%	0.9178	1.8245	2.5	0.9165
MultiBoosting Nnge	0.8318	80.27%	0.9184	1.6244	1.6	0.9433
C45	0.7159	75.7%	0.8372	2.2108	2.7	0.8736
MultiBoosting C45	0.8536	80.74%	0.9661	1.3483	1.6	0.9475

5 Conclusions

We presented a novel approach to mining ill-defined data with continuous decision. Experimental results seem to be very interesting and promising. We plan to make more experiments on other types of data, especially on those coming from the proteochemometrics study, to confirm the power of the proposed method.

Acknowledgements:

The authors would like to thank Helena Strömbergsson and Peteris Prusis for valuable discussions and suggestions during preparation of this paper.

The research has been done during the visit of Dr Nguyen Hung Son at the The Linnaeus Centre for Bioinformatics, Uppsala University. This research was partially supported by the ARI project, the grant 3T11C00226 from Ministry of Scientific Research and Information Technology of the Republic of Poland, and a grant from the Knut and Alice Wallenberg foundation.

References

1. Komorowski, J. Skowron, A. and Ohrn, A. 2000. The ROSETTA system, Klosgen W and Zytkow J, Handbook of Data Mining and Knowledge Discovery, Oxford University Press, ch. D.2.3.

2. Pawlak Z.: *Rough sets: Theoretical aspects of reasoning about data*, Kluwer Dordrecht, 1991.
3. S. Siegel, N.J. Castellan (1988). Nonparametric statistics for the behavioral sciences (2nd ed.) New York: McGraw-Hill.
4. Stone, P.: Layered Learning in Multi-Agent Systems: A Winning Approach to Robotic Soccer. The MIT Press, Cambridge, MA (2000)
5. I. H. Witten, E. Frank. Data Mining: Practical machine learning tools with Java implementations. Morgan Kaufmann, San Francisco, 1999.
6. Geoffrey I. Webb (2000). MultiBoosting: A Technique for Combining Boosting and Wagging. Machine Learning, 40(2): 159-196, Kluwer Academic Publishers, Boston
7. Ross Quinlan (1993). C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers, San Mateo, CA.

Soft Computing Approach to the Analysis of the Amino Acid Similarity Matrices

Witold R. Rudnicki[1] and Jan Komorowski[2]

[1] Interdisciplinary Centre for Mathematical and Computational Modelling,Warsaw University
Pawinskiego 5a, 02-106, Warsaw, Poland
[2] The Linnaeus Centre for Bioinformatics, Uppsala University BMC, Box 598, Husargatan 3, SE-751 24 Uppsala, Sweden

Abstract. Amino acid similarity matrices are used for protein sequence comparison. A new method for studying connections between amino acid properties and similarity matrices is proposed. A representation of the amino acid similarity matrices, by means of the equivalence classes between amino acids, is proposed. It is shown that mappings between these equivalence classes and contiguous subsets of the amino acid property space are possible. These mappings are consistent between equivalence classes. There is a possibility of practical applications of this method in sequence similarity searches.

1 Introduction

Methods for assessing protein sequence similarity are widely used in genetics and molecular biology. Amino acid similarity matrices (*AASMs*) known also as the amino acid substitution matrices are used by most popular algorithms, such as BLAST [1] or FASTA [9]. A large number of AASMs was developed to date, based on various assumptions and different methodologies. Among those, matrices belonging to the so-called BLOSUM family, are used most often and therefore are also used in this study. BLOSUM matrices were developed from studies on amino acid variation in the well aligned sequences of related proteins [4]. Each element of the matrix is indexed by two amino acids and it's value is related to the probability of finding these two amino acids in the same position in the sequence of similar proteins.

$$BLOSUM[i,j] = 2\log_2 \frac{p(A_iA_j)}{p(A_i)p(A_j)}, \tag{1}$$

where,
A_i, A_j are the *i-th* and *j-th* amino acid, respectively, $p(A_iA_j)$ is probability of finding amino acids A_i and A_j in the same position, $p(A_i)$ is the frequency of occurrence of amino acid A_i in the sequence database.

Amino acid properties have been studied for many years. Several hundred properties, which have been determined experimentally for the twenty amino acids that are the normal components of proteins in all living organisms,

are stored in the aaindex the database of amino acid properties compiled by Kanehisa and coworkers [5]. It has been shown that with the help of the principal component analysis most of the variability in the properties of the amino acids can be attributed to five independent properties [11,13]. First three principal components correlate well with hydrophobicity, size and polarity, respectively. These three amino acid properties are traditionally considered important for their characterization. The remaining two components don't have simple interpretation, as they are compound properties with many components.

Relationships between AASMs and properties of amino acids have already been studied quite extensively. A good summary of the work in this field and detailed analysis of these relations is given by Tomi and Kanehisa [12]. It has been shown that for the BLOSUM matrices the absolute values of linear correlations between differences in amino acid properties and the terms in the similarity matrices are close to 0.7. In this study we propose a method that enables us to study relationships between amino acid properties and the amino acid similarity matrices in a greater detail. In our approach we build a representation of the AASM using equivalence classes between amino acids. It is assumed that these equivalence classes arise from the properties of the amino acids. Our goal is to identify these properties. We approach this goal with soft computing methods.

2 Materials and Methods

AASMs are square matrices, Element M_{nk} of the AASM is related to the probability that amino acids n and k will occupy the same position in the sequence in two related proteins. Positive values correspond to a high probability of occurrence of corresponding amino acids in the same position in the sequence (amino acids are similar to each other), negative values correspond to a small probability. The maximal value for each column and row is on the diagonal, because each amino acid is most similar to itself. In the case of BLOSUM family of matrices all elements are integers.

The reconstruction of AASMs in terms of equivalence classes of amino acids requires few steps. First, AASMs need to be transformed to the non-negative form by adding constant to each element. It results in matrix, that is non-negative, symmetric and for each row and column the maximal value lays on the diagonal. The upper left triangle of the original and transformed BLOSUM 62 matrix is presented in Table 1.

Let us assume that we have a set of N bit-vectors, where a bit-vector is defined as a vector for which each component can be either *1* or *0* like in the bit-vectors B_1,B_2,B_3:

$\boldsymbol{B_1} = [0,1,1,1,0,0,0,0,0,1,0,0,0,0,0,0,1,0,0,0,1,0]$
$\boldsymbol{B_2} = [0,0,0,1,0,0,1,0,0,1,0,0,0,0,1,0,1,0,0,0,0,0]$
$\boldsymbol{B_3} = [0,0,0,0,0,0,1,1,1,0,0,0,0,0,0,0,1,0,0,0,0]$

Table 1. The upper-left corner of the Blosum 62 matrix.

	A	R	N	D	C	Q	E
A	4	-1	-2	-2	0	-1	-1
R	-1	5	0	-2	-3	1	0
N	-2	0	6	1	-3	0	0
D	-2	-2	1	6	-3	0	2
C	0	-3	-3	-3	9	-3	-4
Q	-1	1	0	0	-3	5	2
E	-1	0	0	2	-4	2	5

	A	R	N	D	C	Q	E
A	8	3	2	2	4	3	3
R	3	9	4	2	1	5	4
N	2	4	10	5	1	4	4
D	2	2	5	10	1	4	6
C	4	1	1	1	13	1	0
Q	3	5	4	4	1	9	6
E	3	4	4	6	0	6	9

One may define square matrix M_{ij} for which element $[i, j]$ is obtained as a scalar product of the *i-th* and *j-th* bit-vector:

$$M_{ij} = \boldsymbol{B}_i \cdot \boldsymbol{B}_j \tag{2}$$

For example, using bit-vectors defined earlier we may obtain the following 3×3 matrix:

$$\begin{vmatrix} 6 & 3 & 0 \\ 3 & 5 & 1 \\ 0 & 1 & 4 \end{vmatrix}$$

One may note that such $N \times N$ matrix has all necessary properties: it is symmetric, it's diagonal and largest elements are on the diagonal. We thus represent amino acids as bit-vectors and obtain from these bit-vectors a matrix with desired properties.

One should note that two amino acid equivalence classes are defined for each position in the bit-vector representation. One equivalence class will consist of amino acids with *1* at this position, another one is formed by the remaining ones. We may assume that each position of the bit-vector represents some abstract property.

All matrices obtained from bit-vectors in the way described above will be symmetric, non-negative and with dominant diagonal; however, the inverse statement that all matrices with these properties can be obtained from the set of bit-vectors is not true. Therefore finding the set of bit-vectors that would generate accurately the matrices from the BLOSUM family may be very difficult if not impossible. Taking into account the experimental origin of the AASMs and inherent experimental errors and uncertainties this is not a very serious limitation, if only the matrices that are very close to the original matrix can be generated. In this study the Euclidean distance between matrices is used as a measure of the distance between the matrices. The algorithm for the construction of the similarity matrices is based on the simulated annealing procedure [6], with the Metropolis Monte Carlo [8,7] scheme used to search configuration space, the Euclidean distance between

matrices as the system energy and the values for the system temperatures adjusted experimentally. The initial set of amino acid representations was selected randomly. For each bit-vector the number of non-zero bits was equal to the diagonal value of the non-negative representation of the similarity matrix. New states were generated by random swap of positions of two bits in a single bit-vector. For each matrix 200 simulations were run. For each of the runs the best representation was stored for further analysis.

A simplified algorithm used for the generation of the representations can be summarized in the following pseudo-code:

```
for 50 iterations {
  for each temperature in [300,100,50,20,10] {
    for 10 000 steps (temperature) {
      for each amino acid representation {
        compute d1= dist(Original AASM,Derived AASM)
        move randomly position of random non-zero bit
        compute Modified Derived AASM
        compute d2= dist(Original AASM, Modified Derived AASM)
        if (d2<d1) { accept move }
        else { accept move with probability exp(-(d2-d1)/T) }
      }
    }
  }
}
```

The goal of this study is to expand the knowledge about relationships between properties of amino acids and AASMs. We do this by finding the properties. which can be attributed directly to the equivalence classes between amino acids. To this end we use the notion of *generalized properties (GPs)*, which are defined here as the subsets of the property space. The generalized properties are constructed by selecting the set of defining properties and selecting valid range of values for each of them. A detailed description of the algorithm for the derivation of the general properties coherent with the equivalence classes was given elsewhere [10].

The notion of the generalized property can be illustrated with the following example. Let us define the following generalized property: *small and hydrophobic.* The atomic mass may be used as a descriptor for the size and hydrophobic scale [3] as the descriptor for the hydrophobicity. One can arbitrarily decide that *small* amino acids have atomic mass smaller than 110 a.m.u and an amino acid regarded as *hydrophobic* when it's hydrophobicity index is higher than 1.0. The values of these parameters for the amino acids are presented in Table 2.

With such definition the following amino acids belong to the class *small and hydrophobic:* **pro, cys, val.** Let us notice that a small increase in the atomic mass parameter (from 110 to 120 a.m.u) leads to a larger class *small*

Table 2. Hydrophobicity and molecular weight for naturally occuring amino acids

	ala	arg	asn	asp	cys	gln	glu	gly	his	ile
hydrophobicity	0.61	0.60	0.06	0.46	1.07	0.00	0.47	0.07	0.61	2.22
mol. wt	89.1	74.2	132.1	133.1	121.2	146.2	147.1	75.1	155.2	131.2
	leu	lys	met	phe	pro	ser	thr	trp	tyr	val
hydrophobicity	1.53	1.15	1.18	2.02	1.95	0.05	0.05	2.65	1.88	1.32
mol. wt	131.2	146.2	149.2	165.2	115.1	105.1	119.1	204.2	181.2	117.2

and hydrophobic: **pro, cys, val, ile, leu**. Thus the generalized property defined here as *small hydrophobic* can be obtained by setting limits to the values of size and hydrophobicity. In a more formal language the generalized property generating the equivalence class is a minimal compact subset of the property space, with boundaries orthogonal to the axes, in which all amino acids in this subset belong to the equivalence class, and all amino acids belonging to the equivalence class belong to the subset. In our approach no more than three elementary properties can be used to define boundaries. Using larger number of elementary properties would lead to a very large number of possible combinations of properties. Taking into account that there are only twenty natural amino acids, this would certainly inrease the risk of finding artifacts. The generalized properties can be grouped in the *types*. A single type consists of all generalized properties based on the same basic properties.

The algorithm for generation of the equivalence classes can be written in simplified version in the form of the following pseudo-code:

```
For each EqClass_i  {
  For each property P_j {
    Construct generalized property GP_j
    Check if EqClass_i is explained by GP_j
    For each property P_k {
      Construct generalized property GP_k
      Check if EqClass_i is explained by GP_j x GP_k
      For each property P_l {
        Construct generalized property GP_l
        Check if EqClass_i is explained by GP_j x GP_k x GP_l
      }
    }
  }
}
```

This procedure, described in greater detail in [10], is able to generate generalized properties, however, it may discover real properties as well as artifacts. A verification procedure is necessary to ensure that the results represent true relationships between amino acid properties and AASMs. To this

end the statistical properties of the generalized properties constructed with the randomized values of the basic properties were analyzed.

A first obvious test is the comparison between the number of the equivalence classes which can be explained in terms of the generalized properties for the real and randomized properties. If the latter number was small this test would be sufficient. However, initial tests have shown that the number of equivalence classes, which can be explained by random properties, is not negligible. Therefore additional tests were required.

One should note, that five basic properties can be used to generate 85 types of generalized properties (five GPs based on single property, twenty GPs constructed from pairs of basic properties and sixty GPs constructed from three basic properties). However, it is unlikely that more than few of these generalized properties are meaningful. Therefore one should be able to explain most equivalence classes using just few combinations of the basic properties. On the other hand, in the case of random properties this should not be the case. Therefore the number of equivalence classes reconstructed from few types of the generalized properties shall be a good test, whether the relationships between equivalence classes and amino acid properties are real or artifacts.

Another test is based on a similar idea. It is natural to assume, that similar equivalence classes should arise due to similar generalized properties. Similar equivalence classes can be clustered and these clusters should be explainable in terms of a single generalized property type if the generalized properties are real. Therefore, clusters based on the real generalized properties should be significantly larger than those based the randomized properties.

For all tests we performed 100 simulations using equivalence classes obtained from the BLOSUM AASMs and randomized properties.

3 Results and Discussion

Two hundred simulations was performed for each AASM in the BLOSUM family. We were unable to obtain exact AASM from the BLOSUM family using bit-vector representations. However, in all cases the matrices obtained as a result of simulated annealing were very close to the original ones, in most cases differing only in few elements by no more than 1. All 200 representations were unique combination of the equivalence classes, however, most of the equivalence classes in each of the representations were observed in numerous representations.

We observed 9346 unique equivalence classes, 1487 of them appeared more than 10 times, 489 appeared more than 50 times and 132 appeared more than 200 times in AASM representations.

The complete results of the research carried out within this project, including the detailed analysis of the biological implications, will be presented elsewhere. The results presented below illustrate an implementation of the

methodology and give an answer to the very important question, whether the procedure described above gives results that are significantly different from the statistical noise. The analysis is performed for:

- the total number of equivalence classes reconstructed with the generalized properties;
- the number of equivalence classes reconstructed with the generalized properties belonging to limited numbers of types (between 3 and 7);
- the number of large clusters of equivalence classes that are reconstructed using the same properties;
- the number of equivalence classes in these clusters.

The results of the analysis are presented in Table 3. These result show clearly that the connection between equivalence classes and generalized properties is real. For all measured variables the values obtained for real properties are significantly different from the random noise. On the other hand, equivalence classes can be constructed from random data.

Table 3. Statistical properties of the equivalence classes based on the random data compared to the results for the real equivalence classes

Measured variable	Mean	StDev	Real data	Confidence level
Equivalence classes reconstructed using:				
3 types of GPs	436	29	711	1 6.8e-16
4 types of GPs	501	32	791	1 5.6e-15
5 types of GPs	555	35	845	1 2.8e-13
6 types of GPs	604	36	885	1 3.0e-13
7 types of GPs	644	38	922	1 3.3e-12
Reconstructed equivalence classes (total)	1164	57	1378	1 1.5e-4
Number of large clusters	9.2	2.6	21	1 8.0e-06
Equivalence classes in clusters	274	58	679	1 1.6e-10
Total size of clusters	395	110	1380	1 9.7e-15

The difference between real and randomized properties is largest in the two cases for the number of the equivalence classes constructed from the limited number of types of generalized properties and for the number of equivalence classes in clusters. It suggests that further analysis of the biological importance of the generalized properties should be limited to a small number of properties, for which clusters of equivalence classes are easily constructed.

Relationships between amino acid equivalence classes and amino acid properties can be utilized in the similarity searches. PSI-Blast suite of programs [2], which is the standard tool in this area, utilizes the so-called sequence specific similarity matrices. These matrices can be easily converted to the equivalence classes, which in turn can be used to enhance quality of scoring functions. The possibility of applications of the methods presented in

this study in the sequence similarity searches is the subject of the ongoing project in our laboratories.

Acknowledgments:

This project received partial support from the Human Research Potential & the Socio-economic Knowledge Base: Access to Research Infrastructure program under the HPRI-CT-2001-00153 contract, The Knut and Alice Wallenberg Foundation and The Swedish Foundation for Strategic Research. The computations were carried out at the Interdisciplinary Centre for the Mathematical and Computational Modelling, Warsaw University (ICM), grant G26-11.

References

1. Altschul, S.F., Gish, W., Miller, W., Myers, E.W. and Lipman D.J. (1990) Basic local alignment search tool, *J. Mol. Biol.,* **215,** 403-410.
2. Altschul, S.F., Madden, T.L., Schaffer, A.A., Zhang, J., Zhang, Z., Miller, W., Lipman, D.J. (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.* **25,** 3389-402.
3. Argos, P., Rao, J.K.M. and Hargrave, P.A. (1982) Structural prediction of membrane-bound proteins *J Eur. J. Biochem.* **128,** 565-575.
4. Henikoff, S. and Henikoff, J.G. (1992) Amino acid substitution matrices from protein blocks, *Proc. Natl. Acad. Sci. USA,* **89,** 1091510919.
5. Kawashima, S. and Kanehisa, M. (2000) AAindex: Amino acid index database, *Nucleic Acids Res.* **28,** 374-374.
6. Kirkpatrick, S., Gelatt Jr., C.D. and Vecchi, M.P. (1983) Optimization by simulated annealing. *Science* **220,** 671680.
7. Metropolis, N.A., Rosenbluth, W., Rosenbluth. M.N., Teller, A.H. and Teller, E. (1953) Equation of State Calculations by Fast Computing Machines. *J. Chem. Phys.* **21** 1087-1092.
8. Metropolis, N. and Ulam, S. (1949) The Monte Carlo method. *J. Am. Statist. Ass.* **44,** 335-341.
9. Pearson, W.R. and Lipman, D.J. (1988) Improved tools for biological sequence comparison, *P. Natl Acad. Sci. USA,* **85,** 2444-2448.
10. Rudnicki W.R. and Komorowski J., (2004) Feature synthesis and extraction for the construction of generalized properties of amino acids. In Lecture Notes in Computer Science **3066/2004,** RSCTC 2004, Uppsala, Sweden, June 1-5, 2004. Proceedings. Tsumoto S., Sowiski R. and Komorowski J., Eds. pp.786 - 791. Heidelberg
11. Sandberg, M., Eriksson, L., Jonsson, J. Sjstrm, M., and Wold, S., (1998) New Chemical Descriptors Relevant for the Design of Biologically Active Peptides: A Multivariate Characterization of 87 Amino Acids *J. Med. Chem.* **41,** 2481-2491.
12. Tomii, K. and Kanehisa, M. (1996) Analysis of amino acid indices and mutation matrices for sequence comparison and structure prediction of proteins, *Protein Eng.* **9,** 27-36.
13. Venkatarajan, M.S. and Braun, W. (2001) New quantitative descriptors of amino acids based on multidimensional scaling of a large number of physicalchemical properties. *J. Mol. Model.* **7,** 445453.

Part XII

Special Session: Artificial Immune Systems

Computing with Idiotypic Networks

Francisco Martins[1] and Neva Slani[2]

[1] Departamento de Matemática, Universidade dos Açores,
Rua da Mãe de Deus, Ponta Delgada, Portugal
[2] Katedra za matematiku i nacrtnu geometriju,
Fakultet strojarstva i brodogradnje, Ivana Lučića, Zagreb, Croatia

Abstract. The paper presents a computer experiment inspired by the immune metaphor and based on the work of Farmer, Packard, and Perelson [FPP86]. We develop a model influenced by the way the immune system works that is well-suited to address a particular class of NP-hard problems. We discuss the results obtained when applying the model to an artificial vision problem denoted *the museum problem*, where artificial agents successfully accomplish a surveillance assignment to protect the pieces of an art exhibition from bad behaved visitors.

1 Introduction

The immune system is a highly complex system responsible for the defense of our body (and of vertebrates in general) against foreign material (or *antigens*). This task is accomplished identifying each molecular shape encountered and mounting the adequate immune response. Therefore, the immune system needs to distinguish the molecules of the body from antigens, *i.e.*, it must classify each molecule as *self* or *non-self*. The self/non-self classification problem is one of its major assignments.

The model we present is inspired by the clonal selection theory introduced by Sir John MacFarlane Burnet [Jer55,Bur59,AN87] and by the idiotypic network hypothesis, introduced by Niels Jerne [Jer74]. These two hypothesis provide the mathematical framework for describing the immune system as a dynamic system of interacting species.

From a computational point of view, the immune system is comparable to the neural system, namely, it possesses the capacity to learn, memorise, and recognise patterns, which are emergent properties of the idiotypic network and of the clonal selection theory.

In this paper, we exploit a model based on the immune system to solve a surveillance problem (artificial vision) of a museum, referred as *the museum problem*. The model follows Farmer *et al* [FPP86] and is used to train a population of artificial agents to detect bad behaved museum visitors.

The experiments we did with the developed model produced very promising results. We were able to evolve and maintain a population of artificial agents, well fitted to the environment, that classified successfully self and non-self material. For a complete discussion on the model and on the results we obtained please refer to [Mar00].

The paper is organised as follows. The next section presents a brief introduction to the immune system focusing on its dynamics and meta-dynamics. Section 3 is devoted to the experiment. We characterise the problem and explain the conducted computer simulation. Last section discusses the results we have achieved with the simulation of the model.

2 Dynamics and meta-dynamics of the Idiotypic Network

The model we present belongs to a class of models introduced by Farmer *et al* [FPP86] and emphasises the two major aspects of the immune system: its *dynamics* and its *meta-dynamics*. The *dynamics* describes the changes over time of the concentration of lymphocyte species (and secreted molecules). The *meta-dynamics* models the continuous process of clonal insertion and elimination of lymphocyte species in the population. Notice that the model does not intend to describe the immune system, but simply to exploit its computational aspects.

We describe the characteristics of the idiotypic network using a system free of antigens, since these do not interfere directly in the idiotypic phenomena. Thus, the environment has no longer the role of training the system (by presenting antigens) and exerts the role of moderator. The repertoire adaptation is done by *reinforcement*, *i.e.* without receiving any information from the outside world (via antigens).

2.1 The artificial immune system

In order to define a computational model based on the immune system we must define its actors, namely, the antigenic environment, lymphocytes, antigens, and the interactions between them.

Since all molecular shapes are settled by polypeptidic chains and all interactions result from chemical bindings of such chains, we encode both features in our model. We adopt the representation in [FPP86] and code the specificity of the immune actors as binary strings. Unlike Farmer we consider only an antigenic determinant for lymphocytes. Therefore, all the immune actors are coded similarly.

In what concerns molecular interactions, there must exist a sufficient affinity between two molecules for them to react. We measure the molecular affinity between two artificial molecules as the complementarity of the binary strings that represent them. A natural choice is to consider that 0 and 1 are complementary units. So, the distance (complementarity) between two binary stings i and j, denoted $d(i, j)$, is the logical *exclusive or* between them.

We associate with each lymphocyte species an activation threshold, D_i, that sets its lower bound of reaction. If the distance between lymphocyte

species i and another molecule is superior to D_i, then the lymphocyte is activated. This means that it is stimulated to clone itself and start an immune response.

2.2 Dynamics

Following [FPP86] we model the variation on the concentration of lymphocyte species as a dynamic system. The concentration of lymphocyte species i, denoted by x_i, is described by the following non-linear differential equation:

$$\dot{x}_i = x_i \left(k_1 S_i + k_2 F_i - k_3 x_i \right), \qquad x_i(0) = c_{min}.$$

The first summand ($k_1 S_i$) is the *suppression coefficient* resulting from the idiotypic interactions (S_i is always a negative value). The second—the *fitness effect*—contributes to increase the species concentrations. Finally, $k_3 x_i$ corresponds to a *death factor* and models the ageing of cells. The constants k_1, k_2, and k_3 allow us to fine tune the relative influence of each part in the equation. Value c_{min} is the initial concentration of each lymphocyte species.
Idiotypic interactions. Niels Jerne hypothesis is a central notion in our work and provides important intuitions on how the immune system regulates itself via idiotypic interactions. The idiotypic interaction measures the suppression coefficient of one species relatively to another. Its intensity reflects the overlapping of pattern recognition between species. Therefore, we define it as a function of the activation threshold and of the distance $d(i, j)$ between the bit strings representing species.

The suppression that lymphocyte species j exerts on i is denoted by m_{ij} and computed as

$$m_{ij} = \begin{cases} 0, & \text{if } D_i + D_j - d(i,j) \leq 0 \\ \frac{d(i,j) - D_i - D_j}{2D_i}, & \text{if } 0 < D_i + D_j - d(i,j) \leq 2D_i \\ -1, & \text{if } 2D_i < D_i + D_j - d(i,j) \end{cases}.$$

The suppression is minimal (0) when species i and j do not interfere (recognise) with each other, and is maximal (-1) when species i is totally overcome by species j. When species intersect, the suppression coefficient is proportional to the intersecting area $d(i, j) - D_i - D_j$.

Notice that the suppression value is not symmetric. The species with a bigger activation threshold exerts more suppression over the other one.

The idiotypic suppression that the network employs on lymphocyte species i is given by

$$S_i = \sum_{j \neq i} m_{ij} x_j,$$

where x_j is the concentration of species j.
Fitness. The fitness measures the adaptability of a lymphocyte species to the antigenic environment. The balance between the activation threshold and

the distance to self material seems a reasonable fitness value, since the model rejects auto-immune species.

Let p_i denotes the bit string that codes the receptor for species i and e_j the coding of a self element. The fitness $f(i,j)$ between p_i and e_j is given by

$$f(i,j) = \frac{D_i}{d(p_i, e_j)}.$$

The fitness of a lymphocyte species i is computed as

$$F_i = \frac{\sum_{j \in \mathcal{P}} f(i,j)}{\#\mathcal{P}}.$$

where $\mathcal{P}$ denotes the set of self material.

2.3 Meta-dynamics

An essential aspect of the immune system is the constant adjustment of its immune repertoire, since the diversity of the antigenic environment is by far superior than the recognition capacity of the immune system. Therefore, besides the variation in the concentration of the species, the immunologic repertoire is itself dynamic. This phenomenon is referred as *meta-dynamics*.

The meta-dynamics of an idiotypic network consists in the study of the topological variations induced by inclusion—*recruitment*—and exclusion of lymphocyte species. The impact in the dynamics includes the change of the number of equations that describe the concentration flow and the idiotypic interactions.

To exploit this double plasticity of the immune system the meta-dynamic phase should occur sparsely when compared to dynamics variation. Hence, permitting the dynamic system to acquire some stability before suffering new perturbations caused by the variation of its population. The stabilisation of the system is important, since it quantitatively reflects each lymphocyte species' fitness and the suppression from the idiotypic network in its concentration.

Recruitment. The genesis of new lymphocyte species copies the biological processes of the immune system: (a) bone marrow production and (b) clonal selection. Bone marrow production of lymphocyte species is simulated by the generation of random sequences that represent their receptors. The goal is to maintain a diverse repertoire and to uncover new evolutionary streams.

Clonal selection, associated with high mutation rates, is an important mechanism in the plasticity of the immune system. We use processes inspired by biological procedures (genetic algorithms) to mate and produce new species from the immune repertoire, namely, selection techniques, crossover, and mutation.

The generation of new species, either by random means or using genetic operators, aims at the refinement of the immune repertoire, searching for

new species that are more suitable to cope with the antigenic environment. The selection between both reproduction processes is random, but we give priority to the reproduction using genetic operators.

New lymphocyte species undergo a maturation phase that filters those that are auto-immune. We test new species against self material and eliminate the ones that react with self fragments (mimicking apoptose).

Elimination. The elimination of lymphocyte species allows the system to dispose species that were overcome by new ones. There are two ways in which a lymphocyte species may be eliminated: (a) it reveals as auto-immune or (b) it possesses a null concentration. The former models the removing (by apoptose) of the agents that interfere with self material. There may be some auto-immune agents in the population because the recruitment phase only shows a fragment of self material. So, when we detect an auto-immune agent it is removed immediately from the population. The latter contributes to an optimisation in the computational algorithm, since a species with no (or very low) concentration has no influence in the system.

3 An experiment with the immune system

This section defines the problem we want to address and explains the decisions we made when planning the computer simulation.

3.1 The museum problem

The problem we propose is an instance of a class of NP-hard problems that can be suitable addressed using techniques inspired by the immune system.

The *museum problem* is defined as follows. Imagine that we want to install a surveillance system on a museum visited regularly by different groups of people. A classical approach implies the recruitment of well-trained security officers, helped with surveillance devices such as cameras and motion detectors. But suppose that we want to perform this task using artificial agents that, after a training phase, become capable to accomplish the surveillance assignment with satisfactory results.

The proposed problem consists in learning the behaviour of groups of visitors during a *guided tours to the museum*, where artificial agents must be able to detect possible "non-normal" behaviours among the visitors. We want to develop an artificial vision system capable to analyse and classify behaviours as self or non-self. From an algorithmic point of view, the problem consists in selecting a population of artificial agents that are tolerant to self behaviours and are able to cover the entire antigenic space (recognise all non-self behaviours).

The motivations to use an algorithm inspired by the immune system seem clear for problems as the one we present.

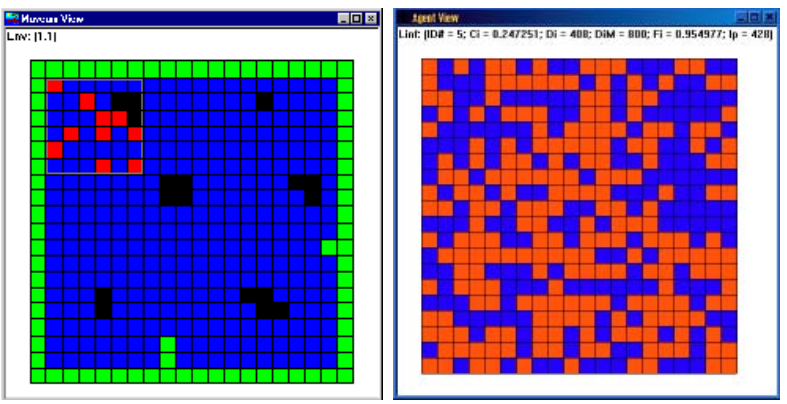
3.2 The experiment

The computer simulation we develop allow us to test the immune algorithm we propose, namely the adequacy of the algorithm select agents that learn and classify patterns.

Coding the environment. The antigenic environment we consider has dimension 20×20. Since there are three relevant subjects: the museum structure, the objects in exhibition, and the visitors, each bit string representing the self is coded as two bits of information per locus, meaning:

00,	an empty space;	01,	a wall;
10,	an object;	11,	a visitor.

To establish a link between the real and artificial environments we represent the self strings as matrices. Figure 1 depicts an environment as shown by the computer simulation.



(a) A self element. (b) An agent.

Fig. 1. Matrix representations of simulation actors.

Coding the artificial agents. Each lymphocyte is coded as a binary string that represents its immune specificity. For the experiment we use binary strings of the same dimension as for the environment. Each position is either 00 or 11. Figure 1 illustrates the matrix representation of an agent.

Coding the guided tour. The behavioural pattern that we learn is "a guided tour" to a museum room. Besides the representation of the room and the visitors, it is important to learn how the tour proceeds, *i.e.*, it is necessary to define the set of rules that govern the guided tour. The aim of the training phase is to select the agents that are able to learn this behaviour.

A guided tour consists in the movement of a group of people around a museum room (place where the exhibition is taking place). The group is fixed at the beginning of the simulation and moves always in the same way, avoiding possible obstacles. The group is always suited in a delimited area (listening to the explanations of the guide). In the present case, there are 10 visitors delimited by a 6×6 square.

Computing the dynamic equation. The non-linear, autonomous dynamic system referred in section 2 is not integrable by algebraic means. Therefore, we compute a numeric approximation using Euler's method that suffices for the study of its dynamics. The concentration of each lymphocyte species is approximated in each simulation step, using

$$x_i(t+1) = x_i(t) + hf(\mathbf{x}(t)).$$

Since we can take a discrete simulation time, and the concentration of all species are computed at each instant, we let $h = 1$. Hence, the approximate variation of the concentration of a lymphocyte species i between time t and $t+1$ is given by

$$x_i(t+1) = x_i(t) + x_i(t)\left(k_2 F_i - k_3 x_i(t) + k_1 \sum_{j \neq i} m_{ij} x_j(t)\right).$$

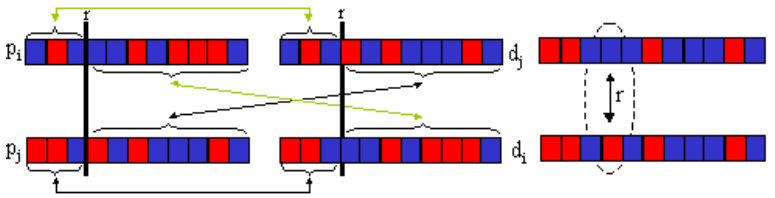
Generation of new lymphocyte species. The generation of new species sets three attributes: (1) the bit string that represents the receptor's molecules; (2) the initial concentration; and (3) the activation threshold.

The process inspired by the bone marrow produces agents with a random generated receptor bit string (the probability of the fragment 00 is equal to the probability of the fragment 11). The initial concentration and the activation threshold are set to the simulations parameters C_{in} and D_{in}, respectively, fixed along the simulation. The results we present were computed with $C_{in} = 0.2$ and $D_{in} = 408$.

The production of agents influenced by clonal selection is based on genetic algorithms, as described in [Gol89], identified as the *simple genetic algorithm*. The algorithm may be decomposed in two major parts: (1) random selection of the well-fitted agents and (2) production of new agents using genetic operators, namely crossover and mutation.

The selection of mates is random, but proportional to the concentration of each species. Therefore, dominating species are likely to be selected.

The operations of crossover and mutation are standard. For the former, we randomly pick a cutting position and generate two offsprings as illustrated in figure 2(a). The concentration of each offspring corresponds to that of the (genetically) closest parent, multiplied by its fitness coefficient. Thus, the offspring's concentration reflects the adequacy of its parents to the antigenic environment. Similarly, the activation threshold is computed from the activation threshold of the (genetically) closest parent incremented with the



(a) Genetic crossover schema.

(b) Genetic mutation.

Fig. 2. Genetic operators used for the production of lymphocyte species.

simulation parameter D_d. In the simulation we use $D_d = 4$. As for mutation (figure 2(b)), we apply it simultaneously with crossover. When cloning the parent strings we make some "mistakes" and copy a 00 as 11 or vice-versa. The probability of a mutation that we use is 0.02, which corresponds to around eight fragments per crossover. Mutation is needed to bring diversity to the population.

Recruitment tests. We run a *recruitment test* to evaluate the plausibility of brand new agents either in what concerns the recognition of self material or its acceptance by the idiotypic network. We perform a test similar to what happens in the thymus (*c.f.* [BS93]). So, every time a new agent is generated we test it against self elements. If the agent recognises some self material, it fails the test and is eliminated. The inclusion of this test improved the results we obtained.

Elimination of agents. The elimination of an agent happens when it reveals auto-immune or when its concentration falls below some minimum threshold (defined by $C_{min} > 0$). In any case, the suppression of some species from the idiotypic network removes all the interaction both with the antigenic environment and the idiotypic network.

3.3 The immune algorithm

The immune algorithm describes the interplay between the dynamics and meta-dynamics of the system. The algorithm is inspired by the immune metaphor and combines ideas from the John Holland's *classifier system* described in [Gol89,Hol75]. Moreover, Farmer *et al* [FPP86] settle a parallel between the immune system and the classifier system.

Figure 3 depicts the immune algorithm we use in the simulation software. There is a generation of an initial population before the iterative process begins, observing the assumptions of the clonal selection and idiotypic network theories.

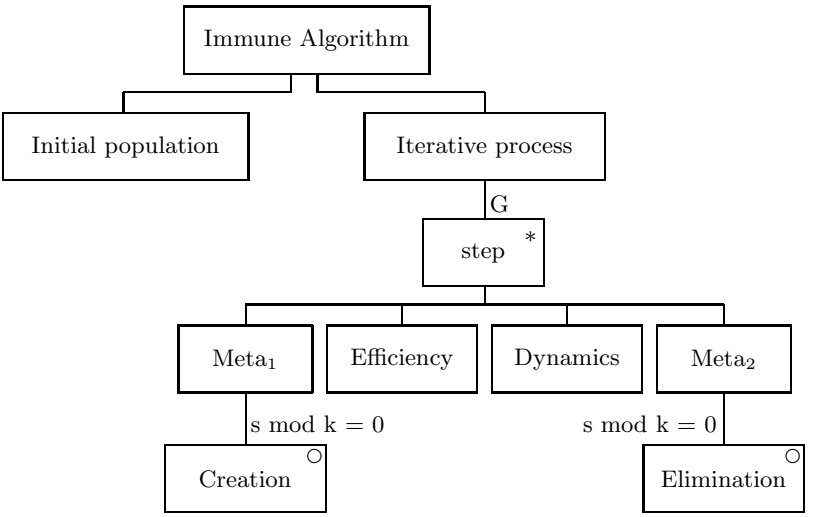


Fig. 3. Michael Jackson's diagram describing the immune algorithm.

The iterative process amounts four major steps. Meta-dynamics is divided into $Meta_1$ and $Meta_2$ steps that perform introduction and elimination of new species, respectively. Notice that the meta-dynamics runs only on each k^{th} step. Processes *efficiency* and *dynamics* carry out, as their names suggest, the updating of efficiency and of the concentration of each species.

This straightforward algorithm is the core of the simulation software we use to test the model. Next section presents the results.

The pattern learning that took place during the training phase is based on a technique called *negative selection* [FPAC95,D'h95,DFH96,D'h96], where the agents that recognise given patterns are discarded. Therefore, the agents are trained not to recognise any self material instead of being trained to recognise non-self material. This technique allows for the learning phase to be performed in an antigen free environment, meaning that, in the case of a surveillance system, the agents are trained in their security task without the presence of intruders.

4 Conclusions

The results we present correspond to a simulation for the parameters described above, with 7500 meta-dynamic steps. The meta-dynamics only takes place when two (artificial) stability conditions are met: (a) the maximal variation in the concentration of the species is inferior to a given threshold (0.001) or (2) after a finite number of tentatives to stabilise the dynamic system.

We discuss the evolution and the adaptation of the immune population during the training phase, and the classification ability of the population.

4.1 Analysis of the immune repertoire

The immune repertoire starts with 150 lymphocyte species and stabilises around 90 species. The existence of an (almost) constant number of individuals in the population reinforces the auto-control of the system. Neither the population grows in an uncontrolled way, nor one species dominates all the other species. The reduction in the number of agents is related to the influence of the idiotypic network. Agents become more and more adapted to the environment and therefore suppress similar ones.

In what concerns the fitness of the repertoire the values are between 94% and 98%. The values do not indicate a clear tendency to optimise as the simulation proceeds. This behaviour is explained by the constant renewal of the population with unfitted agents that are created randomly.

Another interesting indicator to examine is the diversity of the population. The measurements we perform indicate that the population remains diverse during the training phase. This reveals that the genetic operators we use maintain an heterogeneous population during simulation.

So, from a quantitative point of view the results are plausible in what concerns the number of agents, its diversity, and its fitness.

4.2 The cognitive evolution

Now we analyse the model from a qualitative perspective, *i.e.* the capacity of the selected repertoire to classify patterns as self or non-self. The results show the interaction between agents and non-self patterns generated randomly. Figure 4 shows the results of non-self recognition.

As for recognition of self material, the positive reactions are nearly zero. This happens because of the recruitment test we perform. Without this recruitment test some auto-immune agents were incorporated in the idiotypic network and the self recognition was around 10%. The test also added more stability to the number of elements of the population.

To test for non-self recognition we generate random patterns but keep track of the number of relevant positions (11) in the pattern, because we want to cover the entire antigenic space. The model is sensible to the size of the "perturbation" of a non-self element. It seems especially hard to identify non-self elements that posses the same number of relevant positions as the number of visitors used during the training phase. We show the results by perturbation level. Therefore, the response of the system can be measured with accuracy. We consider that the system triggers an alert if at least one agent in the population recognises a pattern.

The results are presented in figure 4. The 3D graphic shows a *time-number of points-recognition rate* relation. Therefore, we can observe, for a selected number of points, the evolution in the recognition of antigens of this class. Also, for a given moment in time, we can check how the immune repertoire covers the antigenic space.

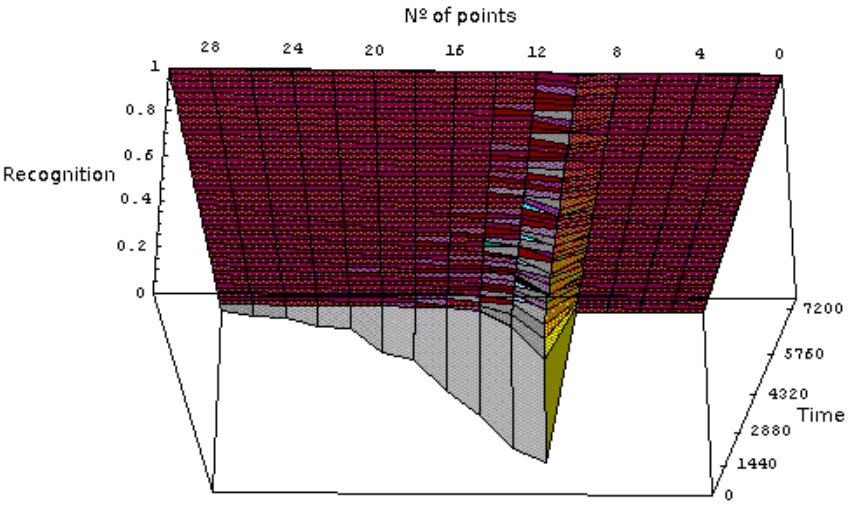


Fig. 4. Non-self recognition rates.

As time passes, the classification of the non-self material increases. The qualitative evolution of the repertoire is interesting in the class [10], which contains the same number of relevant points as the number of visitors. The figure reveals a very good evolution in the selection of the adequate population. Notice that the results for recognition of non-self material are near 100% for all the antigenic space.

As an overall we conclude that the model is suited for NP-hard problems with this kind of configuration. Both the qualitative and the quantitative behaviours allow us to conclude that it is possible to select a stable set of artificial agents that covers an antigenic space, in the present case a museum exhibition, and that we are able to distinguish between normal behaviours (the guided tour to the museum) and any non plausible behaviour.

References

[AN87] Gordon L. Ada e Sir Gustav Nossal. The clonal-selection theory. *Scientific American*, 257(2):50–57, 1987.

[BS93] Hugues Bersini e Gregory Seront. Optimizing with the immune recruitment mechanism. Tech. report TR/IRIDIA/93-6, Institut de Recherches Interdisciplinaires et de Développements en Intelligence Artificielle, 1993.

[Bur59] Frank MacFarlane Burnet. *The Clonal Selection Theory of Acquired Immunity.* Vanderbilt University Press, Nashville, Tennessee, 1959.

[DFH96] Patrik D'haeseleer, Stephanie Forrest e Paul Helman. An immunological approach to change detection: Algorithms, analysis and implications. In *IEEE Symposium on Research in Security and Privacy*, 1996.

[D'h95] Patrik D'haeseleer. A change-detection method inspired by the immune system: Theory, algorithms and techniques. Tech. report CS95-6, Department of Computer Science, University of New Mexico, 1995.

[D'h96] Patrik D'haeseleer. An immunological approach to change detection: Theoretical results. In *The 9th IEEE Computer Security Foundations Workshop*, Dromquinna Manor, County Kerry, Ireland, 1996.

[FPAC95] Stephanie Forrest, Alan S. Perelson, Lawrence Allen e Rajesh Cherukuri. A change-detection algorithm inspired by the immune system. *IEEE Transactions on Software Engineering*, 1995.

[FPP86] J. Doyne Farmer, Norman H. Packard e Alan S. Perelson. The immune system, adaptation, and machine learning. *Physica D*, 22:187–204, 1986.

[Gol89] David E. Goldberg. *Genetic Algorithms in Search, Optimization & Machine Learning*. Addison-Wesley, 1989.

[Hol75] John Holland. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. MIT Press, 4^{th} edition, 1975.

[Jer55] N.K. Jerne. The natural selection theory of antibody formation. *Proceedings of the National Academy of Sciences USA*, 41:849–856, 1955.

[Jer74] N.K. Jerne. Towards a network theory of the immune system. *Ann. Immunol. (Inst. Pasteur)*, 125(C):373–389, 1974.

[Mar00] Francisco Martins. Computation with the Immune System. Masters thesis, Universidade dos Açores, 2000.