Studies in Computational Intelligence, Volume 118

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 volumes of this series can be found on our homepage: springer.com

Vol. 97. Gloria Phillips-Wren, Nikhil Ichalkaranje and Lakhmi C. Jain (Eds.)
Intelligent Decision Making: An AI-Based Approach, 2008
ISBN 978-3-540-76829-9

Vol. 98. Ashish Ghosh, Satchidananda Dehuri and Susmita Ghosh (Eds.)
Multi-Objective Evolutionary Algorithms for Knowledge Discovery from Databases, 2008
ISBN 978-3-540-77466-2

Vol. 99. George Meghabghab and Abraham Kandel
Search Engines, Link Analysis, and User's Web Behavior, 2008
ISBN 978-3-540-77468-6

Vol. 100. Anthony Brabazon and Michael O'Neill (Eds.)
Natural Computing in Computational Finance, 2008
ISBN 978-3-540-77476-1

Vol. 101. Michael Granitzer, Mathias Lux and Marc Spaniol (Eds.)
Multimedia Semantics - The Role of Metadata, 2008
ISBN 978-3-540-77472-3

Vol. 102. Carlos Cotta, Simeon Reich, Robert Schaefer and Antoni Ligeza (Eds.)
Knowledge-Driven Computing, 2008
ISBN 978-3-540-77474-7

Vol. 103. Devendra K. Chaturvedi
Soft Computing Techniques and its Applications in Electrical Engineering, 2008
ISBN 978-3-540-77480-8

Vol. 104. Maria Virvou and Lakhmi C. Jain (Eds.)
Intelligent Interactive Systems in Knowledge-Based Environment, 2008
ISBN 978-3-540-77470-9

Vol. 105. Wolfgang Guenthner
Enhancing Cognitive Assistance Systems with Inertial Measurement Units, 2008
ISBN 978-3-540-76996-5

Vol. 106. Jacqueline Jarvis, Dennis Jarvis, Ralph Rönnquist and Lakhmi C. Jain (Eds.)
Holonic Execution: A BDI Approach, 2008
ISBN 978-3-540-77478-5

Vol. 107. Margarita Sordo, Sachin Vaidya and Lakhmi C. Jain (Eds.)
Advanced Computational Intelligence Paradigms in Healthcare - 3, 2008
ISBN 978-3-540-77661-1

Vol. 108. Vito Trianni
Evolutionary Swarm Robotics, 2008
ISBN 978-3-540-77611-6

Vol. 109. Panagiotis Chountas, Ilias Petrounias and Janusz Kacprzyk (Eds.)
Intelligent Techniques and Tools for Novel System Architectures, 2008
ISBN 978-3-540-77621-5

Vol. 110. Makoto Yokoo, Takayuki Ito, Minjie Zhang, Juhnyoung Lee and Tokuro Matsuo (Eds.)
Electronic Commerce, 2008
ISBN 978-3-540-77808-0

Vol. 111. David Elmakias (Ed.)
New Computational Methods in Power System Reliability, 2008
ISBN 978-3-540-77810-3

Vol. 112. Edgar N. Sanchez, Alma Y. Alanís and Alexander G. Loukianov
Discrete-Time High Order Neural Control: Trained with Kalman Filtering, 2008
ISBN 978-3-540-78288-9

Vol. 113. Gemma Bel-Enguix, M. Dolores Jiménez-López and Carlos Martín-Vide (Eds.)
New Developments in Formal Languages and Applications, 2008
ISBN 978-3-540-78290-2

Vol. 114. Christian Blum, Maria José Blesa Aguilera, Andrea Roli and Michael Sampels (Eds.)
Hybrid Metaheuristics, 2008
ISBN 978-3-540-78294-0

Vol. 115. John Fulcher and Lakhmi C. Jain (Eds.)
Computational Intelligence: A Compendium, 2008
ISBN 978-3-540-78292-6

Vol. 116. Ying Liu, Aixin Sun, Han Tong Loh, Wen Feng Lu and Ee-Peng Lim (Eds.)
Advances of Computational Intelligence in Industrial Systems, 2008
ISBN 978-3-540-78296-4

Vol. 117. Da Ruan, Frank Hardeman and Klaas van der Meer (Eds.)
Intelligent Decision and Policy Making Support Systems, 2008
ISBN 978-3-540-78306-0

Vol. 118. Tsau Young Lin, Ying Xie, Anita Wasilewska and Churn-Jung Liau (Eds.)
Data Mining: Foundations and Practice, 2008
ISBN 978-3-540-78487-6

Tsau Young Lin
Ying Xie
Anita Wasilewska
Churn-Jung Liau
(Eds.)

Data Mining: Foundations and Practice

Springer

Dr. Tsau Young Lin
Department of Computer Science
San Jose State University
San Jose, CA 95192
USA
tylin@cs.sjsu.edu

Dr. Anita Wasilewska
Department of Computer Science
The University at Stony Brook
Stony Brook, New York 11794-4400
USA
anita@cs.sunysb.edu

Dr. Ying Xie
Department of Computer Science
and Information Systems
Kennesaw State University
Building 11, Room 3060
1000 Chastain Road
Kennesaw, GA 30144
USA
yxie2@kennesaw.edu

Dr. Churn-Jung Liau
Institute of Information Science
Academia Sinica
No 128, Academia Road, Section 2
Nankang, Taipei 11529
Taiwan
liaucj@iis.sinica.edu.tw

ISBN 978-3-540-78487-6 e-ISBN 978-3-540-78488-3

Studies in Computational Intelligence ISSN 1860-949X

Library of Congress Control Number: 2008923848

Cover design: Deblik, Berlin, Germany

Printed on acid-free paper

9 8 7 6 5 4 3 2 1

springer.com

Preface

The IEEE ICDM 2004 workshop on the Foundation of Data Mining and the IEEE ICDM 2005 workshop on the Foundation of Semantic Oriented Data and Web Mining focused on topics ranging from the foundations of data mining to new data mining paradigms. The workshops brought together both data mining researchers and practitioners to discuss these two topics while seeking solutions to long standing data mining problems and stimulating new data mining research directions. We feel that the papers presented at these workshops may encourage the study of data mining as a scientific field and spark new communications and collaborations between researchers and practitioners.

To express the visions forged in the workshops to a wide range of data mining researchers and practitioners and foster active participation in the study of foundations of data mining, we edited this volume by involving extended and updated versions of selected papers presented at those workshops as well as some other relevant contributions. The content of this book includes studies of foundations of data mining from theoretical, practical, algorithmical, and managerial perspectives. The following is a brief summary of the papers contained in this book.

The first paper "Compact Representations of Sequential Classification Rules," by Elena Baralis, Silvia Chiusano, Riccardo Dutto, and Luigi Mantellini, proposes two compact representations to encode the knowledge available in a sequential classification rule set by extending the concept of closed itemset and generator itemset to the context of sequential rules. The first type of compact representation is called classification rule cover (CRC), which is defined by the means of the concept of generator sequence and is equivalent to the complete rule set for classification purpose. The second type of compact representation, which is called compact classification rule set (CCRS), contains compact rules characterized by a more complex structure based on closed sequence and their associated generator sequences. The entire set of frequent sequential classification rules can be re-generated from the compact classification rules set.

A new subspace clustering algorithm for high dimensional binary valued dataset is proposed in the paper "An Algorithm for Mining Weighted Dense Maximal 1-Complete Regions" by Haiyun Bian and Raj Bhatnagar. To discover patterns in all subspace including sparse ones, a weighted density measure is used by the algorithm to adjust density thresholds for clusters according to different density values of different subspaces. The proposed clustering algorithm is able to find all patterns satisfying a minimum weighted density threshold in all subspaces in a time and memory efficient way. Although presented in the context of the subspace clustering problem, the algorithm can be applied to other closed set mining problems such as frequent closed itemsets and maximal biclique.

In the paper "Mining Linguistic Trends from Time Series" by Chun-Hao Chen, Tzung-Pei Hong, and Vincent S. Tseng, a mining algorithm dedicated to extract human understandable linguistic trend from time series is proposed. This algorithm first transforms data series to an angular series based on angles of adjacent points in the time series. Then predefined linguistic concepts are used to fuzzify each angle value. Finally, the Aprori-like fuzzy mining algorithm is used to extract linguistic trends.

In the paper "Latent Semantic Space for Web Clustering" by I-Jen Chiang, T.Y. Lin, Hsiang-Chun Tsai, Jau-Min Wong, and Xiaohua Hu, latent semantic space in the form of some geometric structure in combinatorial topology and hypergraph view, has been proposed for unstructured document clustering. Their clustering work is based on a novel view that term associations of a given collection of documents form a simplicial complex, which can be decomposed into connected components at various levels. An agglomerative method for finding geometric maximal connected components for document clustering is proposed. Experimental results show that the proposed method can effectively solve polysemy and term dependency problems in the field of information retrieval.

The paper "A Logical Framework for Template Creation and Information Extraction" by David Corney, Emma Byrne, Bernard Buxton, and David Jones proposes a theoretical framework for information extraction, which allows different information extraction systems to be described, compared, and developed. This framework develops a formal characterization of templates, which are textual patterns used to identify information of interest, and proposes approaches based on AI search algorithms to create and optimize templates in an automated way. Demonstration of a successful implementation of the proposed framework and its application on biological information extraction are also presented as a proof of concepts.

Both probability theory and Zadeh fuzzy system have been proposed by various researchers as foundations for data mining. The paper "A Probability Theory Perspective on the Zadeh Fuzzy System" by Q.S. Gao, X.Y. Gao, and L. Xu conducts a detailed analysis on these two theories to reveal their relationship. The authors prove that the probability theory and Zadeh fuzzy system perform equivalently in computer reasoning that does not involve

complement operation. They also present a deep analysis on where the fuzzy system works and fails. Finally, the paper points out that the controversy on "complement" concept can be avoided by either following the additive principle or renaming the complement set as the conjugate set.

In the paper "Three Approaches to Missing Attribute Values: A Rough Set Perspective" by Jerzy W. Grzymala-Busse, three approaches to missing attribute values are studied using rough set methodology, including attribute-value blocks, characteristic sets, and characteristic relations. It is shown that the entire data mining process, from computing characteristic relations through rule induction, can be implemented based on attribute-value blocks. Furthermore, attribute-value blocks can be combined with different strategies to handle missing attribute values.

The paper "MLEM2 Rule Induction Algorithms: With and Without Merging Intervals" by Jerzy W. Grzymala-Busse compares the performance of three versions of the learning from example module of a data mining system called LERS (learning from examples based on rough sets) for rule induction from numerical data. The experimental results show that the newly introduced version, MLEM2 with merging intervals, produces the smallest total number of conditions in rule sets.

To overcome several common pitfalls in a business intelligence project, the paper "Towards a Methodology for Data Mining Project Development: the Importance of Abstraction" by P. Gonzálex-Aranda, E. Menasalves, S. Millán, Carlos Ruiz, and J. Segovia proposes a data mining lifecycle as the basis for proper data mining project management. Concentration is put on the project conception phase of the lifecycle for determining a feasible project plan.

The paper "Finding Active Membership Functions in Fuzzy Data Mining" by Tzung-Pei Hong, Chun-Hao Chen, Yu-Lung Wu, and Vincent S. Tseng proposes a novel GA-based fuzzy data mining algorithm to dynamically determine fuzzy membership functions for each item and extract linguistic association rules from quantitative transaction data. The fitness of each set of membership functions from an itemset is evaluated by both the fuzzy supports of the linguistic terms in the large 1-itemsets and the suitability of the derived membership functions, including overlap, coverage, and usage factors.

Improving the efficiency of mining frequent patterns from very large datasets is an important research topic in data mining. The way in which the dataset and intermediary results are represented and stored plays a crucial role in both time and space efficiency. The paper "A Compressed Vertical Binary Algorithm for Mining Frequent Patterns" by J. Hdez. Palancar, R. Hdez. León, J. Medina Pagola, and A. Hechavarría proposes a compressed vertical binary representation of the dataset and presents approach to mine frequent patterns based on this representation. Experimental results show that the compressed vertical binary approach outperforms Apriori, optimized Apriori, and Mafia on several typical test datasets.

Causal reasoning plays a significant role in decision-making, both formally and informally. However, in many cases, knowledge of at least some causal

effects is inherently inexact and imprecise. The chapter "Naïve Rules Do Not Consider Underlying Causality" by Lawrence J. Mazlack argues that it is important to understand when association rules have causal foundations in order to avoid naïve decisions and increases the perceived utility of rules with causal underpinnings. In his second chapter "Inexact Multiple-Grained Causal Complexes", the author further suggests using nested granularity to describe causal complexes and applying rough sets and/or fuzzy sets to soften the need for preciseness. Various aspects of causality are discussed in these two chapters.

Seeing the needs for more fruitful exchanges between data mining practice and data mining research, the paper "Does Relevance Matter to Data Mining Research" by Mykola Pechenizkiy, Seppo Puuronen, and Alexcy Tsymbal addresses the balance issue between the rigor and relevance constituents of data mining research. The authors suggest the study of the foundation of data mining within a new proposed research framework that is similar to the ones applied in the IS discipline, which emphasizes the knowledge transfer from practice to research.

The ability to discover actionable knowledge is a significant topic in the field of data mining. The paper "E-Action Rules" by Li-Shiang Tsay and Zbigniew W. Ras proposes a new class of rules called "E-action rules" to enhance the traditional action rules by introducing its supporting class of objects in a more accurate way. Compared with traditional action rules or extended action rules, e-action rule is easier to interpret, understand, and apply by users. In their second paper "Mining e-Action Rules, System DEAR," a new algorithm for generating e-action rules, called Action-tree algorithm is presented in detail. The action tree algorithm, which is implemented in the system DEAR2.2, is simpler and more efficient than the action-forest algorithm presented in the previous paper.

In his first paper "Definability of Association Rules and Tables of Critical Frequencies," Jan Ranch presents a new intuitive criterion of definability of association rules based on tables of critical frequencies, which are introduced as a tool for avoiding complex computation related to the association rules corresponding to statistical hypotheses tests. In his second paper "Classes of Association Rules: An Overview," the author provides an overview of important classes of association rules and their properties, including logical aspects of calculi of association rules, evaluation of association rules in data with missing information, and association rules corresponding to statistical hypotheses tests.

In the paper "Knowledge Extraction from Microarray Datasets Using Combined Multiple Models to Predict Leukemia Types" by Gregor Stiglic, Nawaz Khan, and Peter Kokol, a new algorithm for feature extraction and classification on microarray datasets with the combination of the high accuracy of ensemble-based algorithms and the comprehensibility of a single decision tree is proposed. Experimental results show that this algorithm is able

to extract rules by describing gene expression differences among significantly expressed genes in leukemia.

In the paper "Using Association Rules for Classification from Databases Having Class Label Ambiguities: A Belief Theoretic Method" by S.P. Subasinghua, J. Zhang, K. Premaratae, M.L. Shyu, M. Kubat, and K.K.R.G.K. Hewawasam, a classification algorithm that combines belief theoretic technique and portioned association mining strategy is proposed, to address both the presence of class label ambiguities and unbalanced distribution of classes in the training data. Experimental results show that the proposed approach obtains better accuracy and efficiency when the above situations exist in the training data. The proposed classifier would be very useful in security monitoring and threat classification environments where conflicting expert opinions about the threat category are common and only a few training data instances available for a heightened threat category.

Privacy preserving data mining has received ever-increasing attention during the recent years. The paper "On the Complexity of the Privacy Problem" explores the foundations of the privacy problem in databases. With the ultimate goal to obtain a complete characterization of the privacy problem, this paper develops a theory of the privacy problem based on recursive functions and computability theory.

In the paper "Ensembles of Least Squares Classifiers with Randomized Kernels," the authors, Kari Torkkola and Eugene Tuv, demonstrate that stochastic ensembles of simple least square classifiers with randomized kernel widths and OOB-past-processing achieved at least the same accuracy as the best single RLSC or an ensemble of LSCs with fixed tuned kernel width, but require no parameter tuning. The proposed approach to create ensembles utilizes fast exploratory random forests for variable filtering as a preprocessing step; therefore, it can process various types of data even with missing values.

Shusahu Tsumoto contributes two papers that study contigency table from the perspective of information granularity. In the first paper "On Pseudo-statistical Independence in a Contingency Table," Shusuhu shows that a contingency table may be composed of statistical independent and dependent parts and its rank and the structure of linear dependence as Diophatine equations play very important roles in determining the nature of the table. The second paper "Role of Sample Size and Determinants in Granularity of Contingency Matrix" examines the nature of the dependence of a contingency matrix and the statistical nature of the determinant. The author shows that as the sample size N of a contingency table increases, the number of 2×2 matrix with statistical dependence will increase with the order of N^3, and the average of absolute value of the determinant will increase with the order of N^2.

The paper "Generating Concept Hierarchy from User Queries" by Bob Wall, Neal Richter, and Rafal Angryk develops a mechanism that builds concept hierarchy from phrases used in historical queries to facilitate users' navigation of the repository. First, a feature vector of each selected query is generated by extracting phrases from the repository documents matching the

query. Then the Hierarchical Agglomarative Clustering algorithm and subsequent portioning and feature selection and reduction processes are applied to generate a natural representation of the hierarchy of concepts inherent in the system. Although the proposed mechanism is applied to an FAQ system as proof of concept, it can be easily extended to any IR system.

Classification Association Rule Mining (CARM) is the technique that utilizes association mining to derive classification rules. A typical problem with CARM is the overwhelming number of classification association rules that may be generated. The paper "Mining Efficiently Significant Classification Associate Rules" by Yanbo J. Wang, Qin Xin, and Frans Coenen addresses the issues of how to efficiently identify significant classification association rules for each predefined class. Both theoretical and experimental results show that the proposed rule mining approach, which is based on a novel rule scoring and ranking strategy, is able to identify significant classification association rules in a time efficient manner.

Data mining is widely accepted as a process of information generalization. Nevertheless, the questions like what in fact is a generalization and how one kind of generalization differs from another remain open. In the paper "Data Preprocessing and Data Mining as Generalization" by Anita Wasilewska and Ernestina Menasalvas, an abstract generalization framework in which data preprocessing and data mining proper stages are formalized as two specific types of generalization is proposed. By using this framework, the authors show that only three data mining operators are needed to express all data mining algorithms; and the generalization that occurs in the preprocessing stage is different from the generalization inherent to the data mining proper stage.

Unbounded, ever-evolving and high-dimensional data streams, which are generated by various sources such as scientific experiments, real-time production systems, e-transactions, sensor networks, and online equipments, add further layers of complexity to the already challenging "drown in data, starving for knowledge" problem. To tackle this challenge, the paper "Capturing Concepts and Detecting Concept-Drift from Potential Unbounded, Ever-Evolving and High-Dimensional Data Streams" by Ying Xie, Ajay Ravichandran, Hisham Haddad, and Katukuri Jayasimha proposes a novel integrated architecture that encapsulates a suit of interrelated data structures and algorithms which support (1) real-time capturing and compressing dynamics of stream data into space-efficient synopses and (2) online mining and visualizing both dynamics and historical snapshots of multiple types of patterns from stored synopses. The proposed work lays a foundation for building a data stream warehousing system as a comprehensive platform for discovering and retrieving knowledge from ever-evolving data streams.

In the paper "A Conceptual Framework of Data Mining," the authors, Yiyu Yao, Ning Zhong, and Yan Zhao emphasize the need for studying the nature of data mining as a scientific field. Based on Chen's three-dimension view, a threelayered conceptual framework of data mining, consisting of the philosophy layer, the technique layer, and the application layer, is discussed

in their paper. The layered framework focuses on the data mining questions and issues at different abstract levels with the aim of understanding data mining as a field of study, instead of a collection of theories, algorithms, and software tools.

The papers "How to Prevent Private Data from Being Disclosed to a Malicious Attacker" and "Privacy-Preserving Naive Bayesian Classification over Horizontally Partitioned Data" by Justin Zhan, LiWu Chang, and Stan Matwin, address the issue of privacy preserved collaborative data mining. In these two papers, secure collaborative protocols based on the semantically secure homomorphic encryption scheme are developed for both learning Support Vector Machines and Nave Bayesian Classifier on horizontally partitioned private data. Analyses of both correctness and complexity of these two protocols are also given in these papers.

We thank all the contributors for their excellent work. We are also grateful to all the referees for their efforts in reviewing the papers and providing valuable comments and suggestions to the authors. It is our desire that this book will benefit both researchers and practitioners in the filed of data mining.

Tsau Young Lin
Ying Xie
Anita Wasilewska
Churn-Jung Liau

Contents

Compact Representations of Sequential Classification Rules

Elena Baralis, Silvia Chiusano, Riccardo Dutto, and Luigi Mantellini

Politecnico di Torino, Dipartimento di Automatica ed Informatica
Corso Duca degli Abruzzi 24, 10129 Torino, Italy
elena.baralis@polito.it, silvia.chiusano@polito.it,
riccardo.dutto@polito.it, luigi.mantellini@polito.it

Summary. In this chapter we address the problem of mining sequential classification rules. Unfortunately, while high support thresholds may yield an excessively small rule set, the solution set becomes rapidly huge for decreasing support thresholds. In this case, the extraction process becomes time consuming (or is unfeasible), and the generated model is too complex for human analysis.

We propose two compact forms to encode the knowledge available in a sequential classification rule set. These forms are based on the abstractions of general rule, specialistic rule, and complete compact rule. The compact forms are obtained by extending the concept of closed itemset and generator itemset to the context of sequential rules. Experimental results show that a significant compression ratio is achieved by means of both proposed forms.

1 Introduction

Association rules [3] describe the co-occurrence among data items in a large amount of collected data. They have been profitably exploited for classification purposes [8, 11, 19]. In this case, rules are called classification rules and their consequent contains the class label. Classification rule mining is the discovery of a rule set in the training dataset to form a model of data, also called classifier. The classifier is then used to classify new data for which the class label is unknown.

Data items in an association rule are unordered. However, in many application domains (e.g., web log mining, DNA and proteome analysis) the order among items is an important feature. Sequential patterns have been first introduced in [4] as a sequential generalization of the itemset concept. In [20,24,27,35] efficient algorithms to extract sequences from sequential datasets are proposed. When sequences are labeled by a class label, classes can be modeled by means of sequential classification rules. These rules are implications where the antecedent is a sequence and the consequent is a class label [17].

E. Baralis et al.: *Compact Representations of Sequential Classification Rules*, Studies in Computational Intelligence (SCI) **118**, 1–30 (2008)
www.springerlink.com

In large or highly correlated datasets, rule extraction algorithms have to deal with the combinatorial explosion of the solution space. To cope with this problem, pruning of the generated rule set based on some quality indexes (e.g., confidence, support, and χ^2) is usually performed. In this way rules which are redundant from a functional point of view [11, 19] are discarded. A different approach consists in generating equivalent representations [7] that are more compact, without information loss.

In this chapter we propose two compact forms to represent sets of sequential classification rules. The first compact form is based on the concept of generator sequence, which is an extension to sequential patterns of the concept of generator itemset [23]. Based on generator sequences, we define general sequential rules. The collection of all general sequential rules extracted from a dataset represents a sequential classification rule cover. A rule cover encodes all useful classification information in a sequential rule set (i.e., is equivalent to it for classification purposes). However, it does not allow the regeneration of the complete rule set.

The second proposed compact form exploits jointly the concepts of closed sequence and generator sequence. While the notion of generator sequence, to our knowledge, is new, closed sequences have been introduced in [29,31]. Based on closed sequences, we define closed sequential rules. A closed sequential rule is the most specialistic (i.e., characterized by the longest sequence) rule into a set of equivalent rules. To allow regeneration of the complete rule set, in the compact form each closed sequential rule is associated to the complete set of its generator sequences.

To characterize our compact representations, we first define a general framework for sequential rule mining under different types of constraints. Constrained sequence mining addresses the extraction of sequences which satisfy some user defined-constraints. Example of constraints are minimum or maximum gap between events [5,17,18,21,25], sequence length or regular expression constraints over a sequence [16, 25]. We characterize the two compact forms within this general framework.

We then define a specialization of the proposed framework which addresses the maximum gap constraint between consecutive events in a sequence. This constraint is particularly interesting in domains where there is high correlation between neighboring elements, but correlation rapidly decreases with distance. Examples are the biological application domain (e.g., the analysis of DNA sequences), text analysis, web mining. In this context, we present an algorithm for mining our compact representations.

The chapter is organized as follows. Section 2 introduces the basic concepts and notation for the sequential rule mining task, while Sect. 3 presents our framework for sequential rule mining. Sections 4 and 5 describe the compact forms for sequences and for sequential rules, respectively. In Sect. 6 the algorithm for mining our compact representations is presented, while Sect. 7 reports experimental result on the compression effectiveness of the proposed techniques. Section 8 discusses previous related work. Finally, Sect. 9 draws some conclusions and outlines future work.

2 Definitions and Notation

Let $\mathcal{I}$ be a set of items. A sequence S on $\mathcal{I}$ is an ordered list of events, denoted $S = (e_1, e_2, \ldots, e_n)$, where each event $e_i \in S$ is an item in $\mathcal{I}$. In a sequence, each item can appear multiple times, in different events. The overall number of items in S is the length of S, denoted $|S|$. A sequence of length n is called n-sequence.

A dataset $\mathcal{D}$ for sequence mining consists of a set of input-sequences. Each input-sequence in $\mathcal{D}$ is characterized by a unique identifier, named Sequence Identifier (SID). Each event within an input-sequence SID is characterized by its position within the sequence. This position, named event identifier (eid), is the number of events which precede the event itself in the input-sequence.

Our definition of input-sequence is a restriction of the definition proposed in [4, 35]. In [4, 35] each event in an input-sequence contains more items and the eid identifier associated to the event corresponds to a temporal timestamp. Our definition considers instead domains where each event is a single symbol and is characterized by its position within the input-sequence. Applicative examples are the biological domain for proteome or DNA analysis, or the text mining domain. In these contexts each event corresponds to either an aminoacid or a single word.

When dataset $\mathcal{D}$ is used for classification purposes, each input-sequence is labeled by a class label c. Hence, dataset $\mathcal{D}$ is a set of tuples (SID, S, c), where S is an input-sequence identified by the SID value and c is a class label belonging to the set $\mathcal{C}$ of class labels in $\mathcal{D}$. Table 1 reports a very simple sequence dataset, used as a running example in this chapter.

The notion of containment between two sequences is a key concept to characterize the sequential classification rule framework. In this section we introduce the general notion of sequence containment. In the next section, we explore the concept of containment between two sequences and we formalize the concept of sequence containment with constraints.

Given two arbitrary sequences X and Y, sequence Y "contains" X when it includes the events in X in the same order in which they appear in X [5, 35]. Hence, sequence X is a *subsequence* of sequence Y. For example for sequence $Y = ADCBA$, some possible subsequences are ADB, DBA, and CA.

An arbitrary sequence X is a sequence in dataset $\mathcal{D}$ when at least one input-sequence in $\mathcal{D}$ "contains" X (i.e., X is the subsequence of some input-sequences in $\mathcal{D}$).

Table 1. Example sequence dataset $\mathcal{D}$

SID	Sequence	Class
1	$ADCA$	c_1
2	$ADCBA$	c_2
3	ABE	c_1

A sequential rule [4] in $\mathcal{D}$ is an implication in the form $X \rightarrow Y$, where X and Y are sequences in $\mathcal{D}$ (i.e., both are subsequences of some input-sequences in $\mathcal{D}$). X and Y are respectively the antecedent and the consequent of the rule. Classification rules (i.e., rules in a classification model) are characterized by a consequent containing a class label. Hence, we define sequential classification rules as follows.

Definition 1 (Sequential Classification Rule). *A sequential classification rule $r : X \rightarrow c$ is a rule for $\mathcal{D}$ when there is at least one input-sequence S in $\mathcal{D}$ such that (i) X is a subsequence of S, (ii) and S is labeled by class label c.*

Differently from general sequential rules, the consequent of a sequential classification rule belongs to set $\mathcal{C}$, which is disjoint from $\mathcal{I}$. We say that a rule $r : X \rightarrow c$ *covers* (or *classifies*) a data object d if d "contains" X. In this case, r classifies d by assigning to it class label c.

3 Sequential Classification Rule Mining

In this section, we characterize our framework for sequential classification rule mining. Sequence containment is a key concept in our framework. It plays a fundamental role both in the rule extraction phase and in the classification phase. Containment can be defined between:

- *Two arbitrary sequences*. This containment relationship allows us to define generalization relationships between sequential classification rules. It is exploited to define the concepts of closed and generator sequence. These concepts are then used to define two concise representations of a classification rule set.
- *A sequence and an input-sequence*. This containment relationship allows us to define the concept of support for both a sequence and a sequential classification rule.

Various types of constraints, discussed later in the section, can be enforced to restrict the general notion of containment. In our framework, sequence mining is constrained by two sets of functions (Ψ, Φ). Set Ψ describes containment between two arbitrary sequences. Set Φ describes containment between a sequence and an input-sequence, and allows the computation of sequence (and rule) support. Sets Ψ and Φ are characterized in Sects. 3.1 and 3.2, respectively. The concise representations for sequential classification rules we propose in this work require pair (Ψ, Φ) to satisfy some properties, which are discussed in Sect. 3.3. Our definitions are a generalization of previous definitions [5, 17], which can be seen as particular instances of our framework. In Sect. 3.4 we discuss some specializations of our (Ψ, Φ)-constrained framework for sequential classification rule mining.

3.1 Sequence Containment

A sequence X is a *subsequence* of a sequence Y when Y contains the events in X in the same order in which they appear in X [5,35].

Sequence containment can be ruled by introducing constraints. Constraints define how to select events in Y that match events in X. For example, in [5] the concept of contiguity constraint was introduced. In this case, events in sequence Y should match events in sequence X without any other interleaved event. Hence, X is a *contiguous subsequence* of Y. In the example sequence $Y = ADCBA$, some possible contiguous subsequence are ADC, DCB, and BA.

Before formally introducing constraints, we define the concept of matching function between two arbitrary sequences. The matching function defines how to select events in Y that match events in X.

Definition 2 (Matching Function). *Let $X = (x_1, \ldots, x_m)$ and $Y = (y_1, \ldots, y_l)$ be two arbitrary sequences, with arbitrary length l and $m \leq l$. A function $\psi : \{1, \ldots, m\} \longrightarrow \{1, \ldots, l\}$ is a matching function between X and Y if ψ is strictly monotonically increasing and $\forall j \in \{1, \ldots, m\}$ it is $x_j = y_{\psi(j)}$.*

The definition of constrained subsequence is based on the concept of matching function. Consider for example sequences $Y = ADCBA$, $X = DCB$, and $Z = BA$. Sequence X matches Y with respect to function $\psi(j) = 1 + j$ (with $1 \leq j \leq 3$), and sequence Z matches Y according to function $\psi(j) = 3 + j$ (with $1 \leq j \leq 2$). Hence, sequences X and Z match Y with respect to the class of possible matching functions in the form $\psi(j) = \mathit{offset}{+}j$.

Definition 3 (Constrained Subsequence). *Let Ψ be a set of matching functions between two arbitrary sequences. Let $X = (x_1, \ldots, x_m)$ and $Y = (y_1, \ldots, y_l)$ be two arbitrary sequences, with arbitrary length l and $m \leq l$. X is a constrained subsequence of Y with respect to Ψ, written as $X \sqsubseteq_\Psi Y$, if there is a function $\psi \in \Psi$ such that X matches Y according to ψ.*

Definition 3 yields two particular cases of sequence containment based on the length of sequences X and Y. When X is shorter than Y (i.e., $m < l$), then X is a *strict* constrained subsequence of Y, written as $X \sqsubset_\Psi Y$. Instead, when X and Y have the same length (i.e., $m = l$), the subsequence relation corresponds to the identity relation between X and Y.

Definition 3 can support several different types of constraints on subsequence matching. Both unconstrained matching and contiguous subsequence are particular instances of Definition 3. In particular, in the case of contiguous subsequence, set Ψ includes the complete set of matching function in the form $\psi(j) = \mathit{offset} + j$. When set Ψ is the universe of all the possible matching functions, sequence X is an *unconstrained* subsequence (or simply a subsequence) of sequence Y, denoted as $X \sqsubseteq Y$. This case corresponds to the usual definition of subsequence [5,35].

3.2 Sequence Support

The concept of support is bound to dataset $\mathcal{D}$. In particular, for a sequence X the support in a dataset $\mathcal{D}$ is the number of input-sequences in $\mathcal{D}$ which contain X [4]. Hence, we need to define when an input-sequence contains a sequence. Analogously to the concept of sequence containment introduced in Definition 3, an input-sequence S contains a sequence X when the events in X match the events in S based on a given matching function. However, in an input-sequence S events are characterized by their position within S. This information can be exploited to constrain the occurrence of an arbitrary sequence X in the input-sequence S.

Commonly considered constraints are maximum and minimum gap constraints and windows constraints [17, 25]. Maximum and minimum gap constraints specify the maximum and minimum number of events in S which may occur between two consecutive events in X. The window constraint specifies the maximum number of events in S which may occur between the first and last event in X. For example sequence ADA occurs in the input-sequence $S = ADCBA$, and satisfies a minimum gap constraint equal to 1, a maximum gap constraint equal to 3 and a window constraint equal to 4.

In the following we formalize the concept of gap constrained occurrence of a sequence into an input-sequence. Similarly to Definition 3, we introduce a set of possible matching function to check when an input-sequence S in $\mathcal{D}$ contains an arbitrary sequence X. With respect to Definition 3, these matching functions may incorporate gap constraints. Formally, a gap constraint on a sequence X and an input-sequence S can be formalized as $Gap\ \theta\ K$, where Gap is the number of events in S between either two consecutive elements of X (i.e., maximum and minimum gap constraints), or the first and last elements of X (i.e., window constraint), θ is a relational operator (i.e., $\theta \in \{>, \geq, =, \leq, <\}$), and K is the maximum/minimum acceptable gap.

Definition 4 (Gap Constrained Subsequence). *Let $X = (x_1, \ldots, x_m)$ be an arbitrary sequence and $S = (s_1, \ldots, s_l)$ an arbitrary input-sequence in $\mathcal{D}$, with arbitrary length $m \leq l$. Let Φ be a set of matching functions between two arbitrary sequences, and Gap θ K be a gap constraint. Sequence X occurs in S under the constraint Gap θ K, written as $X \preceq_\Phi S$, if there is a function $\varphi \in \Phi$ such that (a) $X \sqsubseteq_\Phi S$ and (b) depending on the constraint type, φ satisfies one of the following conditions*

- $\forall j \in \{1, \ldots, m-1\}$, $(\varphi(j+1) - \varphi(j)) \leq K$, *for maximum gap constraint*
- $\forall j \in \{1, \ldots, m-1\}$, $(\varphi(j+1) - \varphi(j)) \geq K$, *for minimum gap constraint*
- $(\varphi(m) - \varphi(1)) \leq K$, *for window constraint*

When no gap constraint is enforced, the definition above corresponds to Definition 3. When consecutive events in X are adjacent in input-sequence S, then X is a *string sequence* in S [32]. This case is given when the maximum gap constraint is enforced with maximum gap $K = 1$. Finally, when set Φ is the

universe of all possible matching functions, relation $X \preceq_\Phi S$ can be formalized as (a) $X \sqsubseteq S$ and (b) X satisfies $Gap\ \theta\ K$ in S. This case corresponds to the usual definition of gap constrained sequence as introduced for example in [17, 25].

Based on the notion of containment between a sequence and an input-sequence, we can now formalize the definition of support of a sequence. In particular, $sup_\Phi(X) = |\{(SID, S, c) \in \mathcal{D} \mid X \preceq_\Phi S\}|$. A sequence X is frequent with respect to a given support threshold $minsup$ when $sup_\Phi(X) \geq minsup$.

The quality of a (sequential) classification rule $r : X \rightarrow c_i$ may be measured by means of two quality indexes [19], rule support and rule confidence. These indexes estimate the accuracy of r in predicting the correct class for a data object d. Rule support is the number of input-sequences in $\mathcal{D}$ which contain X and are labeled by class label c_i. Hence, $sup_\Phi(r) = |\{(SID, S, c) \in \mathcal{D} \mid X \preceq_\Phi S \wedge c = c_i\}|$. Rule confidence is given by the ratio $conf_\Phi(r) = sup_\Phi(r)/sup_\Phi(X)$. A sequential rule r is frequent if $sup_\Phi(r) \geq minsup$.

3.3 Framework Properties

The concise representations for sequential classification rules we propose in this work require the pair (Ψ, Φ) to satisfy the following two properties.

Property 1 (Transitivity). *Let (Ψ, Φ) define a constrained framework for mining sequential classification rules. Let X, Y, and Z be arbitrary sequences in $\mathcal{D}$. If $X \sqsubseteq_\Psi Y$ and $Y \sqsubseteq_\Psi Z$, then it follows that $X \sqsubseteq_\Psi Z$, i.e., the subsequence relation defined by Ψ satisfies the transitive property.*

Property 2 (Containment). *Let (Ψ, Φ) define a constrained framework for mining sequential classification rules. Let X,Y be two arbitrary sequences in $\mathcal{D}$. If $X \sqsubseteq_\Psi Y$, then it follows that $\{(SID, S, c) \in \mathcal{D} \mid X \preceq_\Phi S\} \supseteq \{(SID, S, c) \in \mathcal{D} \mid Y \preceq_\Phi S\}$.*

Property 2 states the anti-monotone property of support both for sequences and classification rules. In particular, for an arbitrary class label c it is $sup_\Phi(X \rightarrow c) \geq sup_\Phi(Y \rightarrow c)$.

Albeit in a different form, several specializations of the above framework have already been proposed previously [5, 17, 25]. In the remainder of the chapter, we assume a framework for sequential classification rule mining where Properties 1 and 2 hold.

The concepts proposed in the following sections rely on both properties of our framework. In particular, the concepts of closed and generator itemsets in the sequence domain are based on Property 2. These concepts are then exploited in Sect. 5 to define two concise forms for a sequential rule set. By means of Property 1 we define the equivalence between two classification rules. We exploit this property to define a compact form which allows the classification of unlabeled data without information loss with respect to the complete rule set. Both properties are exploited in the extraction algorithm described in Sect. 6.

3.4 Specializations of the Sequential Classification Framework

In the following we discuss some specializations of our (Ψ, Φ)-constrained framework for sequential classification rule mining. They correspond to particular cases of constrained framework for sequence mining proposed in previous works [5, 17, 25]. Each specialization is obtained from particular instances of function sets Ψ and Φ.

Containment between two arbitrary sequences is commonly defined by means of either the unconstrained subsequence relation or the contiguous subsequence relation. In the former, set Ψ is the complete set of all possible matching functions. In the latter, set Ψ includes all matching functions in the form $\psi(j) = \mathit{offset}+j$. It can be easily seen that both notions of sequence containment satisfy Property 1.

Commonly considered constraints to define the containment between an input-sequence S and a sequence X are maximum and minimum gap constraints and window constraint. The gap constrained occurrence of X within S is usually formalized as $X \sqsubseteq S$ and X satisfies the gap constraint in S. Hence, in relation $X \preceq_{\Phi} S$, set Φ is the universe of all possible matching functions and X satisfies $\mathit{Gap}\ \theta\ K$ in S.

- *Window constraint.* Between the first and last events in X the gap is lower than (or equal to) a given window-size. It can be easily seen that an arbitrary subsequence of X is contained in S within the same window-size. Thus, Property 2 is verified. In particular, Property 2 is verified both for unconstrained and contiguous subsequence relations.
- *Minimum gap constraint.* Between two consecutive events in X the gap is greater than (or equal to) a given size. It directly follows that any pair of non-consecutive events in X also satisfy the constraint. Hence, an arbitrary subsequence of X is contained in S within the minimum gap constraint. Thus, Property 2 is verified. In particular, Property 2 is verified both for unconstrained and contiguous subsequence relations.
- *Maximum gap constraint.* Between two consecutive events in X the gap is lower than (or equal to) a given gap-size. Differently from the two cases above, for an arbitrary pair of non-consecutive events in X the constraint may not hold. Hence, not all subsequences of X are contained in input-sequence S. Instead, Property 2 is verified when considering contiguous subsequences of X.

The above instances of our framework find application in different contexts. In the biological application domains, some works address finding DNA sequences where two consecutive DNA symbols are separated by gaps of more or less than a given size [36]. In the web mining area, approaches have been proposed to predict the next web page requested by the user. These works analyze web logs to find sequences of visited URLs where consecutive URLs are separated by gaps of less than a given size or are adjacent in the web log (i.e., maxgap = 1) [32]. In the context of text mining, gap constraints can be

used to analyze word sequences which occur within a given window size, or where the gap between two consecutive words is less than a certain size [6].

The concise forms presented in this chapter can be defined for any framework specialization satisfying Properties 1 and 2. Among the different gap constraints, the maximum gap constraint is particularly interesting, since it finds applications in different contexts. For this reason, in Sect. 6 we address this particular case, for which we present an algorithm to extract the proposed concise representations.

4 Compact Sequence Representations

To tackle with the generation of a large number of association rules, several alternative forms have been proposed for the compact representation of frequent itemsets. These forms include maximal itemsets [10], closed itemsets [23, 34], free sets [12], disjunction-free generators [13], and deduction rules [14]. Recently, in [29] the concept of closed itemset has been extended to represent frequent sequences.

Within the framework presented in Sect. 3, we define the concept of constrained closed sequence and constrained generator sequence. Properties of closed and generator itemsets in the itemset domain are based on the anti-monotone property of support, which is preserved in our framework by Property 2. The definition of closed sequence was previously proposed in the case of unconstrained matching in [29]. This definition corresponds to a special case of our constrained closed sequence. To completely characterize closed sequences, we also propose the concept of generator itemset [9,23] in the domain of sequences.

Definition 5 (Closed Sequence). *An arbitrary sequence X in $\mathcal{D}$ is a closed sequence iff there is not a sequence Y in $\mathcal{D}$ such that (i) $X \sqsubset_{\psi} Y$ and (ii) $sup_{\Phi}(X) = sup_{\Phi}(Y)$.*

Intuitively, a closed sequence is the maximal subsequence common to a set of input-sequences in $\mathcal{D}$. A closed sequence X is a concise representation of all sequences Y that are subsequences of it, and have its same support. Hence, an arbitrary sequence Y is represented in a closed sequence X when Y is a subsequence of X and X and Y have equal support.

Similarly to the frequent itemset context, we can define the concept of *closure* in the domain of sequences. A closed sequence X which represents a sequence Y is the *sequential closure* of Y and provides a concise representation of Y.

Definition 6 (Sequential Closure). *Let X, Y be two arbitrary sequences in $\mathcal{D}$, such that X is a closed sequence. X is a sequential closure of Y iff (i) $Y \sqsubseteq_{\Psi} X$ and (ii) $sup_{\Phi}(X) = sup_{\Phi}(Y)$.*

The next definition extends the concept of generator itemset to the domain of sequences. Different sequences can have the same sequential closure, i.e., they are represented in the same closed sequence. Among the sequences with the same sequential closure, the shortest sequences are called generator sequences.

Definition 7 (Generator Sequence). *An arbitrary sequence X in $\mathcal{D}$ is a generator sequence iff there is not a sequence Y in $\mathcal{D}$ such that (i) $Y \sqsubset_{\Psi} X$ and (ii)$sup_{\Phi}(X) = sup_{\Phi}(Y)$.*

Special cases of the above definitions are the *contiguous closed sequence* and the *contiguous generator sequence*, where the matching functions in set Ψ define a contiguous subsequence relation. Instead, we have an *unconstrained closed sequence* and an *unconstrained generator sequence* when Ψ defines an unconstrained subsequence relation.

Knowledge about generators associated to a closed sequence X allow generating all sequences having X as sequential closure. For example, let closed sequence X be associated to a generator sequence Z. Consider an arbitrary sequence Y with $Z \sqsubset_{\Psi} Y$ and $Y \sqsubset_{\Psi} X$. Then, X is the sequential closure of Y. From Property 2, it follows that $sup_{\Phi}(Z) \geq sup_{\Phi}(Y)$ and $sup_{\Phi}(Y) \geq sup_{\Phi}(X)$. Being X the sequential closure of Z, Z and X have equal support. Hence, Y has the same support as X. It follows that sequence X is the sequential closure of Y according to Definition 6.

In the example dataset, $ADBA$ is a contiguous closed sequence with support 33.33% under the maximum gap constraint 2. $ADBA$ represents contiguous sequences BA, DB, DBA, ADB, $ADBA$ which satisfy the same gap constraint. BA and DB are contiguous generator sequence for $ADBA$.

In the context of association rules, an arbitrary itemset has a unique closure. The property of uniqueness is lost in the sequential pattern domain. Hence, for an arbitrary sequence X the sequential closure can include several closed sequences. We call this set the *closure sequence set* of X, denoted $\mathcal{CS}(X)$. According to Definition 6, the sequential closure for a sequence X is defined based on the pair of matching functions (Ψ, Φ). Being a collection of sequential closures, the closure sequence set of X is defined with respect to the same pair (Ψ, Φ).

Property 3. *Let X be an arbitrary sequence in $\mathcal{D}$ and $\mathcal{CS}(X)$ the set of sequences in $\mathcal{D}$ which are the sequential closure of X. The following properties are verified. (i) If X is a closed sequence, then $\mathcal{CS}(X)$ includes only sequence X. (ii) Otherwise, $\mathcal{CS}(X)$ may include more than one sequence.*

In Property 3, case (i) trivially follows from Definition 5. We prove case (ii) by means of an example. Consider the contiguous closed sequences $ADCA$ and ACA, which satisfy maximum gap 2 in the example dataset. The generator sequence C is associated to both closed sequences. Instead, D is a generator only for $ADCA$. From Property 3 it follows that a generator sequence can generate different closed sequences.

5 Compact Representations of Sequential Classification Rules

We propose two compact representations to encode the knowledge available in a sequential classification rule set. These representations are based on the concepts of closed and generator sequence. One concise form is a lossless representation of the complete rule set and allows regenerating all encoded rules. This form is based on the concepts of both closed and generator sequences. Instead, the other representation captures the most general information in the rule set. This form is based on the concept of generator sequence and it does not allow the regeneration of the original rule set. Both representations provide a smaller and more easily understandable class model than traditional sequential rule representations.

In Sect. 5.1, we introduce the concepts of general and specialistic classification rule. These rules characterize the more general (shorter) and more specific (longer) classification rules in a given classification rule set. We then exploit the concepts of general and specialistic rule to define the two compact forms, which are presented in Sects. 5.2 and 5.3, respectively.

5.1 General and Specialistic Rules

In associative classification [11,19,30], a shorter rule (i.e., a rule with less elements in the antecedent) is often preferred to longer rules with same confidence and support with the intent of both avoiding the risk of overfitting, and reducing the size of the classifier. However, in some applications (e.g., modeling surfing paths in web log analysis [32]), longer sequences may be more accurate since they contain more detailed information. In these cases, longest-matching rules may be preferable to shorter ones. To characterize both kinds of rules, we propose the definition of specialization of a sequential classification rule.

Definition 8 (Classification Rule Specialization). *Let $r_i : X \rightarrow c_i$ and $r_j : Y \rightarrow c_j$ be two arbitrary sequential classification rules for $\mathcal{D}$. r_j is a specialization of r_i iff (i) $X \sqsubset_\Psi Y$, (ii) $c_i = c_j$, (iii) $sup_\Phi(X) = sup_\Phi(Y)$, and (iv) $sup_\Phi(r_i) = sup_\Phi(r_j)$.*

From Definition 8, a classification rule r_j is a specialization of a rule r_i if r_i is more general than r_j, i.e., r_i has fewer conditions than r_j in the antecedent. Both rules assign the same class label and have equal support and confidence.

The next lemma states that any new data object covered by r_j is also covered by r_i. The lemma trivially follows from Property 1, the transitive property of the set of matching functions Ψ.

Lemma 1. *Let r_i and r_j be two arbitrary sequential classification rules for $\mathcal{D}$, and d an arbitrary data object covered by r_j. If r_j is a specialization of r_i, then r_i covers d.*

With respect to the definition of specialistic rule proposed in [11, 19, 30], our definition is more restrictive. In particular, both rules are required to have the same confidence, support and class label, similarly to [7] in the context of associative classification.

Based on Definition 8, we now introduce the concept of general rule. This is the rule with the shortest antecedent, among all rules having same class label, support and confidence.

Definition 9 (General Rule). *Let $\mathcal{R}$ be the set of frequent sequential classification rules for $\mathcal{D}$, and $r_i \in \mathcal{R}$ an arbitrary rule. r_i is a general rule in $\mathcal{R}$ iff $\nexists r_j \in \mathcal{R}$, such that r_i is a specialization of r_j.*

In the example dataset, $BA \rightarrow c_2$ is a contiguous general rule with respect to the rules $DBA \rightarrow c_2$ and $ADBA \rightarrow c_2$. The next lemma formalizes the concept of general rule by means of the concept of generator sequence.

Lemma 2 (General Rule). *Let $\mathcal{R}$ be the set of frequent sequential classification rules for $\mathcal{D}$, and $r \in \mathcal{R}$, $r : X \rightarrow c$, an arbitrary rule. r is a general rule in $\mathcal{R}$ iff X is a generator sequence in $\mathcal{D}$.*

Proof. We first prove the sufficient condition. Let $r_i : X \rightarrow c$ be an arbitrary rule in $\mathcal{R}$, where X is a generator sequence. By Definition 7, if X is a generator sequence then $\forall r_j : Y \rightarrow c$ in $\mathcal{R}$ with $Y \sqsubset_\Psi X$ it is $sup_\Phi(Y) > sup_\Phi(X)$. Thus, r_i is a general rule according to Definition 9. We now prove the necessary condition. Let $r_i : X \rightarrow c$ be an arbitrary general rule in $\mathcal{R}$. For the sake of contradiction, let X not be a generator sequence. It follows that $\exists r_j : Y \rightarrow c$ in $\mathcal{R}$, with $Y \sqsubset_\Psi X$ and $sup_\Phi(X) = sup_\Phi(Y)$. Hence, from Property 2, $\{(SID, S, c) \in \mathcal{D} \mid Y \preceq_\Phi S\} = \{(SID, S, c) \in \mathcal{D} \mid X \preceq_\Phi S\}$, and thus $sup_\Phi(r_i) = sup_\Phi(r_j)$. It follows that r_i is not a general rule according to Definition 9, a contradiction. □

By applying iteratively Definition 8 in set $\mathcal{R}$, we can identify some particular rules which are not specializations of any other rules in $\mathcal{R}$. These are the rules with the longest antecedent, among all rules having same class label, support and confidence. We name these rules *specialistic rules*.

Definition 10 (Specialistic Rule). *Let $\mathcal{R}$ be an arbitrary set of frequent sequential classification rules for $\mathcal{D}$, and $r_i \in \mathcal{R}$ an arbitrary rule. r_i is a specialistic rule in $\mathcal{R}$ iff $\nexists r_j \in \mathcal{R}$ such that r_j is a specialization of r_i.*

For example, $B \rightarrow c_2$ is a contiguous specialistic rule in the example dataset, with support 33.33% and confidence 50%. The contiguous rules $ACBA \rightarrow c_2$ and $ADCBA \rightarrow c_2$ which include it have support equal to 33.33% and confidence 100%.

The next lemma formalizes the concept of specialistic rule by means of the concept of closed sequence.

Lemma 3 (Specialistic Rule). *Let $\mathcal{R}$ be the set of frequent sequential classification rules for $\mathcal{D}$, and $r \in \mathcal{R}$, $r : X \rightarrow c$, an arbitrary rule. r is a specialistic rule in $\mathcal{R}$ iff X is a closed sequence in $\mathcal{D}$.*

Proof. We first prove the sufficient condition. Let $r_i : X \rightarrow c$ be an arbitrary rule in $\mathcal{R}$, where X is a closed sequence. By Definition 5, if X is a closed sequence then $\forall r_j : Y \rightarrow c$ in $\mathcal{R}$, with $X \sqsubset_\Psi Y$ it is $sup_\Phi(X) > sup_\Phi(Y)$. Thus, r_i is a specialistic rule according to Definition 10. We now prove the necessary condition. Let $r_i : X \rightarrow c$ be an arbitrary specialistic rule in $\mathcal{R}$. For the sake of contradiction, let X not be a closed sequence. It follows that $\exists r_j : Y \rightarrow c$ in $\mathcal{R}$, with $X \sqsubset_\Psi Y$ and $sup_\Phi(X) = sup_\Phi(Y)$. Hence, from Property 2, $\{(SID, S, c) \in \mathcal{D} \mid Y \preceq_\Phi S\} = \{(SID, S, c) \in \mathcal{D} \mid X \preceq_\Phi S\}$, and thus $sup_\Phi(r_i) = sup_\Phi(r_j)$. It follows that r_i is not a specialistic rule according to Definition 10, a contradiction. □

5.2 Sequential Classification Rule Cover

In this section we present a compact form which is based on the general rules in a given set $\mathcal{R}$. This form allows the classification of unlabeled data without information loss with respect to the complete rule set $\mathcal{R}$. Hence, it is equivalent to $\mathcal{R}$ for classification purposes.

Intuitively, we say that two rule sets are equivalent if they contain the same knowledge. When referring to a classification rule set, its knowledge is represented by its capability in classifying an arbitrary data object d. Note that d can be matched by different rules in $\mathcal{R}$. Each rule r labels d with a class c. The estimated accuracy of r in predicting the correct class is usually given by r's support and confidence.

The equivalence between two rule sets can be formalized in terms of rule cover.

Definition 11 (Sequential Classification Rule Cover). *Let $\mathcal{R}_1$ and $\mathcal{R}_2 \subseteq \mathcal{R}_1$ be two arbitrary sequential classification rule sets extracted from $\mathcal{D}$. $\mathcal{R}_2$ is a sequential classification rule cover of $\mathcal{R}_1$ if (i) $\forall r_i \in \mathcal{R}_1$, $\exists r_j \in \mathcal{R}_2$, such that r_i is a specialization of r_j according to Definition 8 and (ii) $\mathcal{R}_2$ is minimal.*

When $\mathcal{R}_2 \subseteq \mathcal{R}_1$ is a classification cover of $\mathcal{R}_1$, the two sets classify in the same way an arbitrary data object d. If a rule $r_i \in \mathcal{R}_1$ labels d with class c, then in $\mathcal{R}_2$ there is a rule r_j, where r_i is a specialization of r_j, and r_j labels d with the same class c (see Lemma 1). r_i and r_j have the same support and confidence. It follows that $\mathcal{R}_1$ and $\mathcal{R}_2$ are equivalent for classification purposes.

We propose a compact representation of rule set $\mathcal{R}$ which includes all general rules in $\mathcal{R}$. This compact representation, named *classification rule cover*, encodes all necessary information to perform classification, but it does not allow the regeneration of the complete rule set $\mathcal{R}$.

Definition 12 (Classification Rule Cover). *Let $\mathcal{R}$ be the set of frequent sequential classification rules for $\mathcal{D}$. The classification rule cover of $\mathcal{R}$ is the set*

$$CRC = \{r \in \mathcal{R} | r : G \rightarrow c \wedge G \in \mathcal{G}\},\ \mathcal{G} \text{ is the set of generator sequences in } \mathcal{D}. \tag{1}$$

The next theorem proves that the CRC rule set is a sequential classification rule cover of $\mathcal{R}$. Hence, it is a compact representation of $\mathcal{R}$, equivalent to it for classification purposes.

Theorem 1. *Let $\mathcal{R}$ be the set of frequent sequential classification rules for $\mathcal{D}$. The rule set $CRC \subseteq \mathcal{R}$ is a sequential classification rule cover of $\mathcal{R}$.*

Proof. Consider an arbitrary rule $r_i \in \mathcal{R}$. By Definition 12 and Lemma 2, there exists at least a rule $r_j \in CRC$, r_j not necessarily identical to r_i, such that r_j is a general rule and r_i is a specialization of r_j according to Definition 8. Hence, it follows that the CRC rule set satisfies point (i) in Definition 11. Consider now an arbitrary rule $r_j \in CRC$. By removing r_j, (at least) r_j itself is no longer represented in CRC by Definition 9. Thus, CRC is a minimal representation of $\mathcal{R}$ (point (ii) in Definition 11). □

5.3 Compact Classification Rule Set

In this section we present a compact form to encode a classification rule set, which, differently from the classification rule cover presented in the previous section, allows the regeneration of the original rule set $\mathcal{R}$. The proposed representation relies on the notions of both closed and generator sequences.

In the compact form, both general and specialistic rules are explicitly represented. All the remaining rules are summarized by means of an appropriate encoding. The compact form consists of a set of elements named *compact rules*. Each compact rule includes a specialistic rule, a set of general rules, and encodes a set of rules that are specializations of them.

Definition 13 (Compact Rule). *Let M be an arbitrary closed sequence in $\mathcal{D}$, and $\mathcal{G}(M)$ the set of its generator sequences. Let $c \in \mathcal{C}$ be an arbitrary class label. $\mathcal{F} : (\mathcal{G}(M), M) \rightarrow c$ is a compact rule for $\mathcal{D}$. $\mathcal{F}$ represents all rules $r : X \rightarrow c_i$ for $\mathcal{D}$ with (i) $c_i = c$ and (ii) $M \in \mathcal{CS}(X)$, i.e., M belongs to the sequential closure set of X.*

By Definition 13, the rule set represented in a compact rule $\mathcal{F} : (\mathcal{G}(M), M) \rightarrow c$ includes (i) the rule $r : M \rightarrow c$, which is a specialistic rule since M is a closed sequence; (ii) the set of rules $r : G \rightarrow c$ that are general rules since G is a generator sequence for M (i.e., $G \in \mathcal{G}(M)$); and (iii) a set of rules $r : X \rightarrow c$ that are a specialization of rules in (ii). For rules in case (iii), the antecedent X is a subsequence of M (i.e., $X \sqsubseteq_{\Psi} M$) and it completely includes at least one of the generator sequences in $\mathcal{G}(M)$ (i.e., $\exists G \in \mathcal{G}(M) | G \sqsubseteq_{\Psi} X$).

In the example dataset, the contiguous classification rules $BA \rightarrow c_2$, $DB \rightarrow c_2$, $DBA \rightarrow c_2$, $ADB \rightarrow c_2$, and $ADBA \rightarrow c_2$ are represented in the compact rule $(\{BA, DB\}, ADBA) \rightarrow c_2$.

The next lemma proves that the rules represented in a compact rule are characterized by the same values of support and confidence.

Lemma 4. *Let $\mathcal{F} : (\mathcal{G}(M), M) \rightarrow c$ be an arbitrary compact rule for $\mathcal{D}$. For each rule $r : X \rightarrow c$ represented in $\mathcal{F}$ it is (i) $sup_\Phi(X) = sup_\Phi(M)$ and (ii) $sup_\Phi(r) = sup_\Phi(M \rightarrow c)$.*

Proof. Let $r : X \rightarrow c$ be an arbitrary rule, and $\mathcal{F} : (\mathcal{G}(M), M) \rightarrow c$ an arbitrary compact rule for $\mathcal{D}$. If r is represented in $\mathcal{F}$, then by Definition 13 it is $M \in \mathcal{CS}(X)$. Thus, by Definition 6, $X \sqsubseteq_\Psi M$ and $sup_\Phi(X) = sup_\Phi(M)$. Hence, from Property 2 (containment property) it follows $sup_\Phi(X \rightarrow c) = sup_\Phi(M \rightarrow c)$. □

We use the concept of compact rule to encode the set $\mathcal{R}$ of frequent sequential classification rules. We propose a compact representation of $\mathcal{R}$ named *compact classification rule set* ($CCRS$). This compact form includes one compact rule for each specialistic rule in $\mathcal{R}$. Each compact rule includes the specialistic rule itself and all general rules associated to it.

Definition 14 (Compact Classification Rule Set). *Let $\mathcal{R}$ be the set of frequent sequential classification rules for $\mathcal{D}$. Let $\mathcal{M}$ be the set of closed sequences, and $\mathcal{G}$ the set of generator sequences in $\mathcal{D}$. The compact classification rule set ($CCRS$) is defined as follows*

$$CCRS = \{\mathcal{F} : (\mathcal{G}(M), M) \rightarrow c\},\ \mathcal{G}(M) \subseteq \mathcal{G}\ \wedge\ M \in \mathcal{M} \tag{2}$$

where $\forall r : M \rightarrow c$ in $\mathcal{R}$ such that $M \in \mathcal{M}$, then $\exists \mathcal{F} : (\mathcal{G}(M), M) \rightarrow c$ and $\mathcal{G}(M)$ contains all generator sequences for M.

The following theorem proves that $CCRS$ is a minimal and complete representation of $\mathcal{R}$.

Theorem 2. *Let $\mathcal{R}$ be the set of frequent sequential classification rules for $\mathcal{D}$, and $CCRS$ the compact classification rule cover of $\mathcal{R}$. $CCRS$ is a complete and minimal representation of $\mathcal{R}$.*

Proof. We first prove that $CCRS$ is a complete representation of $\mathcal{R}$. By Definition 14, set $CCRS$ includes one compact rule for each specialistic rule in $\mathcal{R}$. Hence, $\forall r_i : X \rightarrow c$ in $\mathcal{R}$, there is a compact rule $\mathcal{F} : (\mathcal{G}(M), M) \rightarrow c$ in $CCRS$, with $M \in \mathcal{CS}(X)$. This compact rule encodes r_i. Hence $CCRS$ completely represents $\mathcal{R}$. We then prove that $CCRS$ is a minimal representation of $\mathcal{R}$. Consider an arbitrary compact rule $\mathcal{F} : (\mathcal{G}(M), M) \rightarrow c$ in $CCRS$. $\mathcal{F}$ (also) encodes specialistic rule $r_i : M \rightarrow c$ in $\mathcal{R}$. From Property 3 it follows that the sequential closure set of M includes only sequence M (i.e., $\mathcal{CS}(M) = M$). Hence, $\mathcal{F}$ is the unique compact rule in $CCRS$ encoding r_i. By removing this rule, r_i is no longer represented in $CCRS$. Thus, $CCRS$ is a minimal representation of $\mathcal{R}$. □

From the properties of closed itemsets, it follows that a rule set containing only specialistic rules is a compact and lossless representation of $\mathcal{R}$ only when anti-monotonic constraints (e.g., support constraint) are applied. This property is lost in case of non anti-monotonic constraints (e.g., confidence constraint). In the $CCRS$ representation, each compact rule contains all information needed to generate all the rules encoded in it independently from the other rules in the set. Hence, it is always possible to regenerate set $\mathcal{R}$ starting from the $CCRS$ rule set.

6 Mining Compact Representations

In this section we present an algorithm to extract the compact rule set and the classification rule cover representations from a sequence dataset. The algorithm works in a specific instance of our framework for sequential rule mining. Recall that in our framework sequence mining is constrained by the pair (Ψ, Φ). The set of matching functions Ψ defines the containment between a sequence and an input-sequence. In the considered framework instance, functions in Ψ yield a contiguous subsequence relation. Hence, the mined compact representations yield *contiguous closed* sequences and *contiguous generator* sequences. In this section, we will denote the mined sequences simply as generator or closed sequences since the contiguity constraint is assumed. Set Φ contains all matching functions which satisfy the maximum gap constraint. Hence, the gap constrained subsequence relation $X \preceq_\Phi S$ (where X is a sequence and S an input-sequence) can be formalized as $X \sqsubseteq S$ and X satisfies the maximum gap constraint in S. Furthermore, for an easier readability we denote sequence support, rule support, and rule confidence by omitting set Φ.

The proposed algorithm is levelwise [5] and computes the set of closed and generator sequences by increasing length. At each iteration, say iteration k, the algorithm performs the following operations. (1) Starting from set $\mathcal{M}^k$ of k-sequences, it generates set $\mathcal{M}^{k+1}$ of (k+1)-sequences. Then, (2) it prunes from $\mathcal{M}^{k+1}$ sequences encoding only unfrequent classification rules. This pruning method limits the number of iterations and avoids the generation of uninteresting (i.e., unfrequent) rules. (3) The algorithm checks $\mathcal{M}^{k+1}$ against $\mathcal{M}^k$ to identify the subset of closed sequences in $\mathcal{M}^k$ and the subset of generator sequences in $\mathcal{M}^{k+1}$. (4) Based on this knowledge, the algorithm updates the CRC and $CCRS$ sets.

Each sequence is provided of the necessary information to support the next iteration of the algorithm and to compute the compact representations potentially encoded by it. The following information is associated to a sequence X. (a) A sequence identifier list (denoted *id-list*) recording the input-sequences including X. The id-list is a set of triplets $(SID, eid, Class)$, where SID is the input-sequence identifier, eid is the event identifier for the first[1] item of

[1] As discussed afterwards, knowledge about the event identifiers for the other items in X is not necessary.

X within sequence SID, and $Class$ is the class label associated to sequence SID. (b) Two flags, $isClosed$ and $isGenerator$, stating when sequence X is a candidate closed or generator sequence, respectively. (c) The set $\mathcal{G}(X)$ including the sequences which are generators of X.

The proposed algorithm has a structure similar to GSP [5], where sequence mining is performed by means of a levelwise search. To increase the efficiency of our approach, we associate to each sequence an id-list similar to the one in [17].

A sequence X generates a set of classification rules having X as antecedent, and the class labels in the id-list of X as consequent. The support of X ($sup(X)$) is the number of different $SIDs$ in the id-list of X. For a rule $r : X \rightarrow c$, the support ($sup(r)$) is the number of different $SIDs$ in the id-list labeled by the class label c. The confidence is given by *conf(r)=sup(r)/sup(X)*.

The algorithm, whose pseudocode is shown in Fig. 1, is described in the following. As a preliminary step, we compute the set $\mathcal{M}^1$ of 1-sequences which encodes at least one frequent classification rule (line 3). All sequences in $\mathcal{M}^1$ are generator sequences by Definition 7. For each sequence $X \in \mathcal{M}^1$, the set $\mathcal{G}(X)$ of its generator sequences is initialized with the sequence itself. All sequences in $\mathcal{M}^1$ are also candidate closed sequences by Definition 5. Hence, both flags $isClosed$ and $isGenerator$ are set to true.

Generating $\mathcal{M}^{k+1}$. At iteration k+1 we generate set $\mathcal{M}^{k+1}$ by joining $\mathcal{M}^k$ with $\mathcal{M}^k$. Function *generate_cand_closed* (line 10) generates a new (k+1)-sequence $Z \in \mathcal{M}^{k+1}$ by combining two k-sequences $X, Y \in \mathcal{M}^k$.

Our generation method is based on the contiguous subsequence concept (similar to GSP [5]). Sequence $Z \in \mathcal{M}^{k+1}$ is generated from two sequences

```
1.  CompactForm_Miner(D,minsup,minconf,maxgap)
2.  {CRC = CCRS = ∅;
3.    k=1;
4.    M^1 = compute_M^1(D, minsup);
5.    for all X ∈ M^1
6.      CRC = CRC ∪ {extract_general_rules(X,minsup,minconf)};
7.    while(M^k ≠ ∅)
8.      {M^{k+1} = ∅;
9.        for all (X, Y) ∈ M^k ⋈ M^k
10.         {Z=generate_cand_closed(X,Y,maxgap);
11.           if (support_pruning(Z,minsup)==false) then
12.             {M^{k+1} = M^{k+1} ∪ {Z};
13.               evaluate_closure(Z,X,Y);}}
14.       for all X ∈ M^k with X.isClosed == true
15.         CCRS = CCRS ∪ {extract_compact_rules(X,minsup,minconf)};
16.       for all X ∈ M^{k+1} with X.isGenerator == true
17.         CRC = CRC ∪ {extract_general_rules(X,minsup,minconf)};
18.       k= k+1;}
```

Fig. 1. Compact form mining algorithm

$X, Y \in \mathcal{M}^k$ which are contiguous subsequences of Z, i.e., they share with Z either the k-prefix or the k-suffix. In particular, sequences X and Y generate a new sequence Z if (k-1)suffix(X)=(k-1)prefix(Y). Sequence Z thus contains the first item in X, the $k-1$ items common to both X and Y, and the last item in Y. Z should also satisfy the maximum gap constraint.

Based on Property 2, we compute the id-list for sequence Z. Since X and Y are subsequences of Z, sequence Z is contained in the input-sequences common to both X and Y, where Z satisfies the maximum gap constraint. Function *generate_cand_closed* computes the id-list for sequence Z by joining the id-lists of X and Y. This operation corresponds to a temporal join operation [17]. We observe that sequence Z is obtained by extending Y on the left, with the first item of X (or equivalently by extending X on the right, with the last item of Y). By construction, Y (and X) satisfies the maximum gap constraint. Hence, the new sequence Z satisfies the constraint if the gap between the first items of X and Y is lower or equal to *maxgap*. It follows that the only information needed to perform the temporal join operation between X and Y are the *SIDs* of the input-sequences which include X and Y, and the event identifiers associated to the first items of X and Y.

Pruning $\mathcal{M}^{k+1}$ based on support. Function *support_pruning* (line 11) evaluates the support for the sequential classification rules with Z as antecedent and the class labels in the id-list of Z as consequent. Sequence Z is discarded when none of its associated classification rules has support above *minsup*. Otherwise Z is added to $\mathcal{M}^{k+1}$. This pruning criterion exploits the well known anti-monotone property of support [3], which is guaranteed by Property 2 in our framework. If a classification rule $Z \rightarrow c_i$ does not satisfy the support constraint, then no classification rule $K \rightarrow c_j$, with Z subsequence of K and $c_i = c_j$ can satisfy the support constraint.

Checking closed sequences in $\mathcal{M}^k$ and generator sequences in $\mathcal{M}^{k+1}$. Consider an arbitrary sequence $Z \in \mathcal{M}^{k+1}$, generated from sequences $X, Y \in \mathcal{M}^k$ as described above. Function *evaluate_closure* (line 13) checks if Z is a candidate sequential closure according to Definition 6 for either X or Y, or both of them. Function *evaluate_closure* compares the support of Z with the supports of X and Y. Three cases are given:

1. $\text{sup(Z)} < \text{sup(X)}$ and $\text{sup(Z)} < \text{sup(Y)}$, i.e., Z is not a candidate sequential closure for either X or Y.
2. $\text{sup}(Z) = \text{sup}(X)$, i.e., Z is a candidate sequential closure for X.
3. $\text{sup}(Z) = \text{sup}(Y)$, i.e., Z is a candidate sequential closure for Y.

In case (1), sequence Z is a generator sequence according to Definition 7, since it has lower support than any of its contiguous subsequences. The only two contiguous subsequences of Z in $\mathcal{M}^k$ are X and Y. By Property 1, any subsequence of X or Y is also a subsequence of Z. Hence, all possible contiguous subsequences of Z are X, Y, and the contiguous subsequences of X

and Y. Both X and Y have support higher than Z. By Property 2, any subsequence of X (or Y) has support higher than or equal to X (or Y). Hence, Z is a generator sequence by Definition 7. At this step, sequence Z is also a candidate closed itemset. The set of its generator sequences is initialized with the sequence Z itself ($\mathcal{G}(Z) = Z$).

In case (2), sequence X is not a closed sequence according to Definition 5. Instead, Z is a candidate sequential closure for X. Furthermore, Z is a candidate sequential closure for all sequences represented in X. In fact, sequences represented in X are contiguous subsequences of X that have its same support. They are generated from X by means of the sequences in $\mathcal{G}(X)$. By Property 1, all subsequences of X are also subsequences of Z. Hence, all generator sequences associated to X are inherited by Z. Analogously to case (2), in case (3) Y is not a closed sequence. All generator sequences associated to Y are inherited by Z.

Function *evaluate_closure* updates the flag *isClosed* for sequences X, Y, and Z, and flag *isGenerator* for sequence Z. The flag *isGenerator* is *true* for a generator sequence, and is $false$ otherwise. The flag *isClosed* is *true* for a candidate closed sequence and is $false$ for a non closed sequence.

Updating sets CRC and CCRS. Once the generation of set $\mathcal{M}^{k+1}$ is completed, all sequences in $\mathcal{M}^{k+1}$ have been marked as actual generator or non generator sequences. In addition, all candidate closed sequences in $\mathcal{M}^k$ have been marked as actual closed or non closed sequences.

For each closed sequence $X \in \mathcal{M}^k$, function *extract_compact_rules* (line 15) extracts the compact rules with $\{\mathcal{G}(X), X\}$ as antecedent and that satisfy both support and confidence constraints. These rules are included in the $CCRS$ rule set.

For each generator sequence $Z \in \mathcal{M}^{k+1}$, function *extract_general_rules* (line 17) extracts the general rules with Z as antecedent that satisfy both support and confidence constraints. These rules are added to the CRC rule set.

6.1 Example

By means of the example dataset in Table 1, we describe how the proposed algorithm performs the extraction of the CRC and $CCRS$ rule sets. Due to the small size of the example, we do not enforce any support and confidence constraint, and as gap constraint we consider $maxgap = 1$.

The first step is the generation of set $\mathcal{M}^1$ (function *compute_*$\mathcal{M}^1$ in line 4). Since no support constraint is enforced, $\mathcal{M}^1$ includes all sequences with length equal to 1. Set $\mathcal{M}^1$ is shown in Fig. 2a. By Definition 7, all sequences in $\mathcal{M}^1$ are contiguous generator sequences. For each of them, the set $\mathcal{G}$ of its generator sequences is initialized with the sequence itself. Furthermore, all sequences in $\mathcal{M}^1$ contribute to the CRC set. This set is shown in Fig. 2b.

By joining $\mathcal{M}^1$ with itself, we generate set $\mathcal{M}^2$ which includes all sequences with length equal to 2 (function *generate_cand_closed* in line 10) and is reported in Fig. 3a. For example, sequence DA is obtained from sequences D

$\mathcal{M}^1$			
Sequence	sup		$\mathcal{G}$
	c_1	c_2	
A	2	1	$\{A\}$
B	1	1	$\{B\}$
C	1	1	$\{C\}$
D	1	1	$\{D\}$
E	1	0	$\{E\}$

(a)

General rules in $\mathcal{M}^1$		
rule	sup%	conf%
$A \rightarrow c_1$	66.66	66.66
$A \rightarrow c_2$	33.33	33.33
$B \rightarrow c_1$	33.33	50.00
$B \rightarrow c_2$	33.33	50.00
$C \rightarrow c_1$	33.33	50.00
$C \rightarrow c_2$	33.33	50.00
$D \rightarrow c_1$	33.33	50.00
$D \rightarrow c_2$	33.33	50.00
$E \rightarrow c_1$	33.33	100.00

(b)

Fig. 2. $\mathcal{M}^1$ set and general rules in $\mathcal{M}^1$

$\mathcal{M}^2$			
Sequence	sup		$\mathcal{G}$
	c_1	c_2	
AB	1	0	$\{AB\}$
AC	1	1	$\{C\}$
AD	1	1	$\{D\}$
AE	1	0	$\{E\}$
BA	0	1	$\{BA\}$
BE	1	0	$\{E\}$
CA	1	1	$\{C\}$
CB	0	1	$\{CB\}$
DA	1	0	$\{DA\}$
DB	0	1	$\{DB\}$
DC	1	1	$\{C, D\}$

(a)

General rules in $\mathcal{M}^2$		
rule	sup%	conf%
$AB \rightarrow c_1$	33.33	100.00
$BA \rightarrow c_2$	33.33	100.00
$CB \rightarrow c_2$	33.33	100.00
$DA \rightarrow c_1$	33.33	100.00
$DB \rightarrow c_2$	33.33	100.00

(b)

Compact rules in $\mathcal{M}^1$		
rule	sup%	conf%
$(\{A\}, A) \rightarrow c_1$	66.66	66.66
$(\{A\}, A) \rightarrow c_2$	33.33	33.33
$(\{B\}, B) \rightarrow c_1$	50.00	50.00
$(\{B\}, B) \rightarrow c_2$	50.00	50.00

(c)

Fig. 3. $\mathcal{M}^2$ set, general rules in $\mathcal{M}^2$, and compact rules in $\mathcal{M}^1$

and A by joining their id-lists. The id-list of DA contains the input-sequences where the gap between D and A is lower than *maxgap*. In particular it contains only the input-sequence with SID = 1.

By checking $\mathcal{M}^1$ against $\mathcal{M}^2$, we identify the subset of closed sequences in $\mathcal{M}^1$ and the subset of generator sequences in $\mathcal{M}^2$ (function *evaluate_closure* in line 13). In set $\mathcal{M}^1$, sequences A and B are closed sequences. For example, sequence B is a closed sequence since both sequences in $\mathcal{M}^2$ including B (i.e., AB and BE) have lower support than it. Hence, we generate the compact rules for sequences A and B (see Fig. 3c). In set $\mathcal{M}^2$, five sequences are generators (i.e., AB, BA, CB, DA and DB). For example, sequence AB is a generator sequence since all its subsequences in $\mathcal{M}^1$ (i.e., A and B) have higher support than it. The set of its generators $\mathcal{G}(AB)$ is initialized with the sequence itself. Figure 3b shows the general rules in $\mathcal{M}^2$.

Compact rules		
rule	sup%	conf%
$(\{A\}, A) \rightarrow c_1$	66.66	66.66
$(\{A\}, A) \rightarrow c_2$	33.33	33.33
$(\{B\}, B) \rightarrow c_1$	33.33	50.00
$(\{B\}, B) \rightarrow c_2$	33.33	50.00
$(\{E\}, AE) \rightarrow c_1$	33.33	100.00
$(\{AB, E\}, ABE) \rightarrow c_1$	33.33	100.00
$(\{C\}, ACA) \rightarrow c_1$	33.33	50.00
$(\{C\}, ACA) \rightarrow c_2$	33.33	50.00
$(\{DA\}, ADA) \rightarrow c_1$	33.33	100.00
$(\{CB, BA\}, ACBA) \rightarrow c_2$	33.33	100.00
$(\{DB, BA\}, ADBA) \rightarrow c_2$	33.33	100.00
$(\{D, C\}, ADCA) \rightarrow c_1$	33.33	50.00
$(\{D, C\}, ADCA) \rightarrow c_2$	33.33	50.00
$(\{CB\}, ADCBA) \rightarrow c_2$	33.33	100.00

(a) $CCRS$ set

General rules		
rule	sup%	conf%
$A \rightarrow c_1$	66.66	66.66
$A \rightarrow c_2$	33.33	33.33
$B \rightarrow c_1$	33.33	50.00
$B \rightarrow c_2$	33.33	50.00
$C \rightarrow c_1$	33.33	50.00
$C \rightarrow c_2$	33.33	50.00
$D \rightarrow c_1$	33.33	50.00
$D \rightarrow c_2$	33.33	50.00
$E \rightarrow c_1$	33.33	100.00
$AB \rightarrow c_1$	33.33	100.00
$BA \rightarrow c_2$	33.33	100.00
$CB \rightarrow c_2$	33.33	100.00
$DA \rightarrow c_1$	33.33	100.00
$DB \rightarrow c_2$	33.33	100.00

(b) CRC set

Fig. 4. Compact representations

Sequences in set $\mathcal{M}^2$ which are not generators inherit generators from their subsequences with the same support. For example, sequence BE contains sequence E, and BE and E have equal support. Hence, we add to $\mathcal{G}(BE)$ all sequences in set $\mathcal{G}(E)$ (i.e., E).

By iteratively applying the algorithm, we generate set $\mathcal{M}^3$, which includes all sequences with length=3, by joining $\mathcal{M}^2$ with itself . For instance, we generate sequence DCA from sequences DC and CA. DCA has the same support as both CA and DC. Hence, DCA is not a generator sequence. Instead, it inherits generators from both CA and DC. Hence $\mathcal{G}(DCA) = \{D, C\}$.

Set $\mathcal{M}^3$ does not contribute to the CRC set, since none of its elements is a generator sequence. For set $\mathcal{M}^2$, only sequence AE is a closed sequence. Hence, it generates the compact rule $(\{E\}, AE) \rightarrow c_1$.

Figure 4 reports the CRC and $CCRS$ sets for our example dataset.

7 Experimental Results

Experiments have been run to evaluate both the compression achievable by means of the proposed compact representations and the performance of the proposed algorithm. To run experiments we considered three datasets. Reuters-21578 news and NewsGroups datasets [2] include textual data. DNA dataset includes collections of DNA sequences [2]. Table 2 reports the number of items, sequences, and class labels for each dataset. For Reuters and NewsGrousp datasets items correspond to words in a text. For DNA dataset items correspond to four aminoacid symbols. Table 2 also shows the maximum, minimum and average length of sequences in the datasets.

Table 2. Datasets

Dataset	Sequences	Sequence length			Items	Classes
		Min	Max	Avg		
Reuters-21578	6,454	4	371	52.03	27,600	10
NewsGroup	2,000	83	21,691	303.97	41,420	20
DNA	2,000	60	60	60	4	3

We ran experiments with different support threshold values (denoted *minsup*) and for different maximum gap values (denoted *maxgap*). Experiments were run on an Intel P4 with 2.8 GHz CPU clock rate and 2 GB RAM. The *CompactForm_Miner* algorithm has been implemented in ANSI C.

7.1 Compression Factor

Let $\mathcal{R}$ be the set of all rules which satisfy both *minsup* and *maxgap* constraints and CRC and $CCRS$ the set of general rules and compact rules satisfying the same constraints. To measure the compression factor achieved by our compact representations, we compare their size with the size of the complete rule set. The compression factor (CF%) for the two representations is respectively $(1 - \frac{|CRC|}{|\mathcal{R}|})\%$ and $(1 - \frac{|CCRS|}{|\mathcal{R}|})\%$.

For the CRC representation, a high compression factor indicates that rules whose antecedent is a generator sequence are a small fraction of $\mathcal{R}$. Instead, for the $CCRS$ representation, a high compression factor indicates that rules whose antecedent is a closed sequence are a small fraction of $\mathcal{R}$. In both cases, a small subset of $\mathcal{R}$ encodes all useful information to model classes.

Different data distributions yield a different behavior when varying *minsup* and *maxgap* values. In the following we summarize some common behaviors. Then, we analyze each dataset separately and discuss it in detail.

For moderately high *minsup* values, the two representations have a very close size (or even exactly the same size). In this case, the subsets of rules in $\mathcal{R}$ having as antecedent a closed sequence or a generator sequence are almost the same.

When lowering the support threshold or increasing the *maxgap* value, the number of rules in set $\mathcal{R}$ and in sets $CCRS$ and CRC increases significantly. In this case, the CRC representation often achieves a higher compression than the $CCRS$ representation. This effect occurs for $maxgap > 1$ and low *minsup* values. In this case, the set of rules with a generator sequence as antecedent is smaller than the set of rules with a closed sequence as antecedent. This occurs because when increasing *maxgap* or decreasing *minsup*, mined sequences are characterized by increasing length. Hence, the number of closed sequences, which are the sequences with the longest antecedent, increases significantly. Instead, the increase in the number of generator sequences, which have shorter

length, is less remarkable. Few generator sequences (in most cases only one) are associated to each closed sequence. In addition, as stated by Property 3, each generator sequence can be common to different closed sequences.[2]

In some cases, the CRC representation achieves a slightly lower compression than the $CCRS$ representation. It occurs for $maxgap = 1$ and low $minsup$ values. With respect to the case above, for this $minsup$ and $maxgap$ values there are a few more generator sequences than closed sequences. On the average more than one generator sequence is associated to each closed sequence (about 2 in the DNA dataset, and 1.2 in the Reuters and Newsgroup datasets). Generator sequences are still common to more closed sequence as stated in Property 3.

Reuters Dataset

Figure 5 reports the total number of rules in set $\mathcal{R}$ for different $minsup$ and $maxgap$ values. Results show that the rule set becomes very large for $minsup = 0.1\%$ and $maxgap \geq 3$ (e.g., 1,306,929 rules for $maxgap = 5$).

Figure 6a, b show the compression achieved by the two compact representations. For both of them, for a given $maxgap$ value, the compression factor increases when $minsup$ decreases. Furthermore, for a given $minsup$ value, the compression factor increases when the $maxgap$ value increases. For both representations, the compression factor is significant when set $\mathcal{R}$ includes many rules. When $minsup = 0.1\%$ and $3 \leq maxgap \leq 5$, $\mathcal{R}$ includes from 184,715 to 1,291,696 rules. Compression ranges from 52.57 to 58.61% for the $CCRS$ representation and from 60.18 to 80.54% for the CRC representation. A lower compression (less than 10%) is obtained when $maxgap = 1$. However, in this case the complete rule set is rather small, since it only includes about 12,000 rules when $minsup = 0.1\%$ and less than 2,000 rules for higher support thresholds.

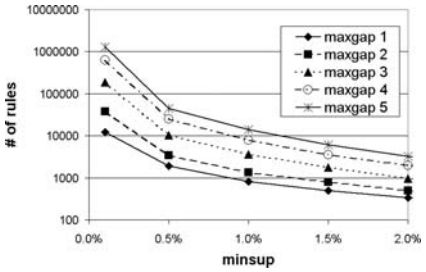
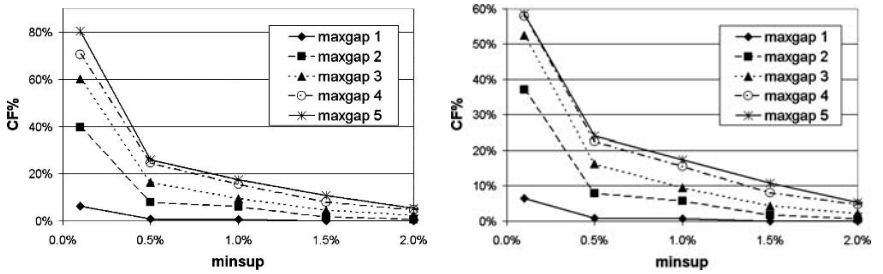


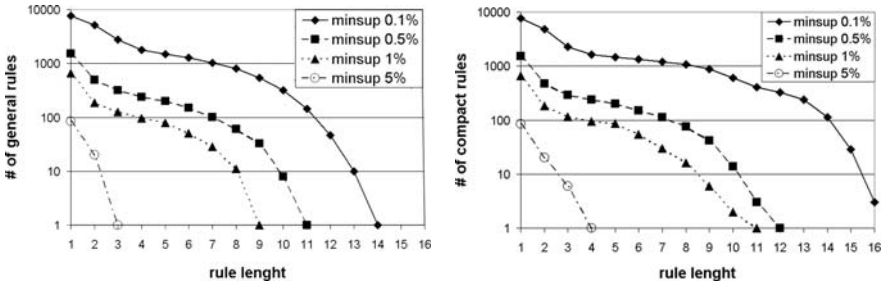
Fig. 5. Number of rules for Reuters dataset

[2]Recall that this behavior is peculiar of the sequential pattern domain. In the context of itemset mining, the number of generator itemsets is always greater than or equal to the number of closed itemsets. Furthermore, the sets of generator itemsets associated to different closed itemsets are disjoint.



(a) *CRC* Set **(b)** *CCRS* Set

Fig. 6. Compression factor for Reuters dataset



(a) *CRC* Set **(b)** *CCRS* Set

Fig. 7. Rule length for *CRC* and *CCRS* sets for Reuters dataset ($maxgap = 2$)

For low support thresholds and high *maxgap* values, the *CRC* representation always achieves a higher compression. In particular, when $minsup = 0.1\%$ and $3 \leq maxgap \leq 5$, the compression factor is more than 10% higher than in the *CCRS* representation (about 20% when $maxgap = 5$). The two representations provide a comparable compression for higher *minsup* and lower *maxgap* values. To analyze this behavior, Fig. 7 plots the number of general and compact rules for different rule lengths, for $maxgap = 2$ and different *minsup* values. As discussed above, when decreasing *minsup*, the number of compact rules increases more significantly. Figure 7 shows that this is due to an increment in the number of compact rules with longer size.

As showed in Fig. 7a, b, for a given *minsup* value compression increases for increasing *maxgap* values. Figure 8 focuses on this issue and plots the compression factor for both compact forms for a large set of *maxgap* values and for thresholds $minsup = 0.5\%$ and $minsup = 1\%$. For both forms the compression factor increases until $maxgap = 5$ and then decreases again. The compression factors are very close until $maxgap = 5$ and then the difference between the two representations becomes more significant. This difference is more relevant when $minsup = 0.5\%$. The *CRC* form always achieves higher compression. An analogous behavior has been obtained for other *minsup* values.

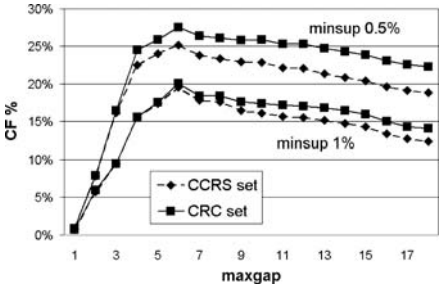
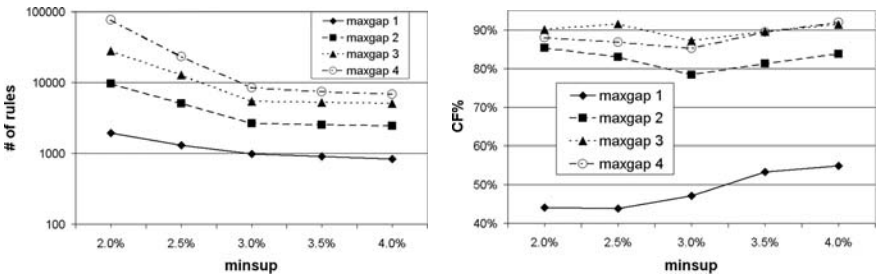


Fig. 8. Compression factor when varying *maxgap* for Reuters dataset



(a) Number of rules **(b)** Compression factor for *CRC* set

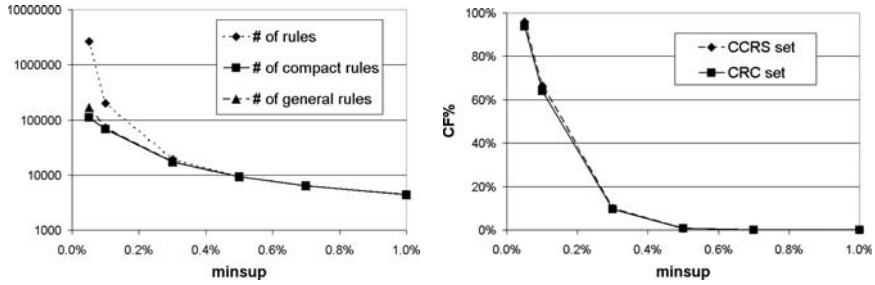
Fig. 9. Newsgroup dataset

Newsgroup Dataset

Figure 9a reports the total number of rules in set $\mathcal{R}$ for different *minsup* and *maxgap* values. The compression factor shows a similar behavior for the two compact forms. In the following we discuss the compression factor for the *CRC* set, taken as a representative example (see Fig. 9b). When $maxgap \neq 1$, the compression factor is only slightly sensitive to the variation of the support threshold. Hence, the fraction of rules with a closed or a generator sequence as antecedent does not vary significantly when vaying support. Similarly to the case of the Reuters dataset, the *CRC* representation always achieves a higher compression than the *CCRS* representation, with an improvement of about 20%.

The case $maxgap = 1$ yields a different behavior. For both representations, the compression factor increases for increasing support thresholds. From Fig. 9b, the cardinality of the complete rule set is rather stable for growing support values. Instead, both the number of closed and generator sequences decreases. This effect yields growing compression when increasing the support threshold.

When varying *maxgap*, both compact forms show a compression factor behavior similar to the Reuters dataset. For a given a *minsup* value, the



(a) Number of rules **(b)** Compression factor

Fig. 10. DNA dataset

compression factor first increases when increasing *maxgap*. After a given *maxgap* value, it decreases again. This behavior is less evident than in the Reuters dataset. Furthermore, the *maxgap* value where the maximum compression is achieved varies with the support threshold.

DNA Dataset

For the DNA dataset, we only consider the case $maxgap = 1$. This constraint is particularly interesting in the biological application domain since sequences of adjacent items in the DNA input sequences are mined. Figure 10a reports the number of rules in sets $\mathcal{R}$, $CCRS$, and CRC for different *minsup* values. Even if the alphabet only includes four symbols, a large number of rules is generated when decreasing the support threshold.

Figure 10b shows the compression factor for the two compact representations. Both compact forms yield significant benefits for low support thresholds. In this case $\mathcal{R}$ contains a large number of rules (2,672,408 rules when minsup=0.05%), while both compact forms have a significantly smaller size (CF=95.85% for the CRC representation and CF=93.74% for the $CCRS$ representation). The CRC representation provides a slightly lower compression than the $CCRS$ representation for low support thresholds. Instead, the compression factor is comparable for high *minsup* values.

7.2 Running Time

For high support thresholds and low *mingap* values, rule mining is performed in less than 60 s for all considered datasets. The CPU time increases when low *minsup* and high *mingap* values are considered. For these values, a larger solution space has to be explored and thus the amount of required memory is large. Our algorithm adopts a levelwise approach which requires a large memory space because of its nature. On the other hand, this approach allows us to explore the solution set and identify both closed and generator sequences, in

order to associate each generator sequence to the appropriate closed sequence. For example, in the Reuters dataset when $minsup = 0.1\%$ and $maxgap$ goes from 3 to 5, CPU times range from 22 min up to 1.80 h. In the Newsgroup dataset, CPU time is always lower than 1 min. However, when we consider $maxgap > 4$ and supports lower than 2.5% more than 10 h are required.

We compared our time performance with the algorithm proposed in [17], an efficient state-of-the-art algorithm for constrained sequence mining. Zaki [17] does not address the extraction of closed and generator sequences. The code was downloaded from [1]. To optimize memory usage, [17] partitions the search space and analyzes each partition independently. The same approach can not be applied in our context due to the type of search we want to perform. When the required memory is not very large, the two algorithms provide comparable performance. Otherwise, [17] yields better performance. For example, when large amounts of memory are required, [17] runs up to five times faster for the Reuters dataset.

8 Related Work

Sequential pattern mining is a relevant research area with applications in a variety of different contexts. Examples of sequential data include text data, DNA sequence, web log files, and customer purchase transactions. The problem of mining sequential patterns has been introduced in [4]. It has been further studied in [20, 24, 27, 35], where different sequence extraction algorithms have been proposed.

Several types of user-defined constraints on sequential pattern have also been considered. For example, minimum or maximum gap constraint between consecutive events or between the first and the last event in the sequence have been addressed in [5, 17, 18, 21, 25]. Alternative constraints for sequence mining are relative to sequence length, occurrence of a set of items within the sequence, or regular expression constraints over a sequence [16, 25].

For classification datasets, where each input sequence is characterized by a class label, sequential classification rules have been introduced in [17]. These rules allow modeling class properties and have been exploited for instance in the web context to predict users' next requests and behavior (e.g., to predict the next request for a web document) [32], or in the biological domain for example to predict protein properties [26].

When low support thresholds are considered or the dataset is highly correlated, a huge set of sequences may be generated, until the problem becomes intractable. To deal with the generation of a large solution set, in the context of association rule mining a significant effort has been devoted to define concise representations for frequent itemsets and association rules. For frequent itemsets these forms are based on the concepts of maximal itemsets [10], closed itemsets [23, 34], free sets [12], disjunction-free generators [13], and deduction

rules [14]. For association rules, concise representations have been proposed based on closed and generator itemsets [22, 23, 33]. In the context of associative classification, compact representations for associative classification rules have been proposed based on generator itemsets [7] and free-sets [15].

In the sequence domain less effort has been so far devoted to mine concise representations. Recently in [29, 31] the concept of closed itemset has been extended to represent frequent sequences, and in [28] an algorithm to mine top-k closed sequential patterns has been presented. As far as we know, no concise representations have been proposed for sequential classification rules.

Our work addresses the definition of concise representations for a sequential classification rule set. We define a general framework for sequential classification rule mining. In the framework, the notions of containment between two arbitrary sequences, and a sequence and an input sequence are a generalization of previous definitions of constrained containment [5, 17, 25]. In this general context, we define the concept of sequence generator, which, to our knowledge, has never been proposed before in the sequence domain. Furthermore, we introduce the concepts of constrained closed sequence and constrained generator sequence. We exploit both concepts to define two compact representations of a classification rule set.

9 Conclusions and Future Work

In this chapter we propose two compact representations to encode the knowledge available in a sequential classification rule set. The classification rule cover (CRC) is defined by means of the concept of generator sequence and yields a simple rule set, which is equivalent to the complete rule set for classification purposes. Compact rules, which are the building blocks of the compact classification rule set ($CCRS$), are characterized by a more complex structure, based on closed sequences and their associated generator sequences. Compact rules allow us to regenerate the entire set of frequent sequential classification rules from the compact form.

Experiments on textual and biological datasets show that the compression ratio is significant for low support thresholds and correlated datasets. In this case, traditional techniques would generate a huge amount of classification rules.

As future work, we plan to exploit our compact representations to design an effective classifier. A promising direction is the integration of both sequential and associative classification rules, to exploit both the specific characterization provided by sequential rules and the general representation given by associative classification rules.

References

1. http://www.cs.rpi.edu/ zaki/software/
2. UCI repository of machine learning databases, http://www.ics.uci.edu/∼ mlearn/mlrepository.html
3. R. Agrawal and R. Srikant. Fast algorithm for mining association rules. In *VLDB'94*, pages 487–499, 1994
4. R. Agrawal and R. Srikant. Mining sequential patterns. In *IEEE ICDE'95*, pages 3–14, 1995
5. R. Agrawal and R. Srikant. Mining sequential patterns: Generalizations and performance improvements. In *EDBT 1996*, pages 3–17
6. H. Ahonen-Myka. Discovery of frequent word sequences in text. In *ESF Exploratory Workshop on Pattern Detection and Discovery 2002*
7. E. Baralis and S. Chiusano. Essential classification rule sets. In *ACM Transactions on Database Systems (TODS), vol. 29, no. 4, December 2004*, pages 635–674
8. E. Baralis and P. Garza. A lazy approach to pruning classification rules. In *IEEE ICDM'02*, pages 35–42, 2002
9. Y. Bastide, R. Taouil, N. Pasquier, and L. Lakhal. Discovering frequent closed itemsets for association rules. In *ICDT'99*, pages 398–416, 1999
10. R.J. Bayardo. Efficiently mining long patterns from databases. In *ACM SIGMOD'98*, pages 85–93, 1998
11. B. Liu, W. Hsu, and Y. Ma. Integrating classification and association rule mining. In *ACM KDD'98*, pages 80–86, 1998
12. J.-F. Boulicaut, A. Bykowski, and C. Rigotti. Free-sets: A condensed representation of boolean data for the approximation of frequency queries. *In Data Mining and Knowledge Discovery journal*, vol. 7, pages 5–22, 2003
13. A. Bykowski and C. Rigotti. A condensed representation to find frequent patterns. In *APODS 2001*
14. T. Calders and B. Goethals. Mining all non-derivable frequent itemsets. *In the 6th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'02), pp. 74–85, Springer*, Berlin Heidelberg New York, 2002
15. B. Cremilleux and J.-F. Boulicaut. Simplest rules characterizing classes generated by delta-free sets. In *Proceedings of the 22 Annual International Conference Knowledge Based Systems and Applied Artificial Intelligence (ES'02)*, pages 33–46, Springer, Berlin Heidelberg New York, December 2002
16. M. Garofalakis, R. Rastogi, and K. Shim. Spirit: Sequential pattern mining with regular expression constraints. In *VLDB 1999*, pages 223–234
17. M.J. Zaki. Sequence mining in categorical domains: Incorporating constraints. In *CIKM 2004*, pages 442–429
18. M. Leleu, C. Rigotti, J.-F. Boulicaut, and G. Euvrard. Constraint-based mining of sequential pattern over datasets with consecutive repetitions. In *PKDD 1997*, pages 303–314
19. W. Li, J. Han, and J. Pei. CMAR: Accurate and efficient classification based on multiple class-association rules. In *IEEE ICDM'01*, pages 369–376, 2001
20. H. Mannila, H. Toivonen, and I. Verkamo. Discovery of frequent episodes in event sequence. In *DMKD 1997*, pages 259–289
21. S. Orlando, R. Perego, and C. Silvestri. A new algorithm for gap constrained sequence mining. In *ACM SAC 2004*

22. N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Closed set based discovery of small covers for association rules. In *BDA 1999*, pages 361–381
23. N. Pasquier, Y. Bastide, R. Taouil, G. Stumme, and L. Lakhal. Mining minimal non-redundant association rules using frequent closed itemsets. In *CL 2000*, pages 972–986
24. J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M.-C. Hsu. PrefixSpan mining sequential patterns efficiently by prefix projected pattern growth. In *IEEE ICDE'01*, pages 215–226, 2001
25. J. Pei, J. Han, and W. Wang. Mining sequential patterns with constraints in large databases. In *CIKM 2002*
26. R. She, F. Chen, K. Wang, M. Ester, J.L. Gardy, and F.S.L. Brinkman. Frequent-subsequence-based prediction of outer membrane proteins. In *SIGKDD 2003*, pages 436–445
27. P. Tzvetkov, X. Yan, and J. Han. Sequential pattern mining using bitmap representation. In *SIGKDD 2002*, pages 429–435
28. P. Tzvetkov, X. Yan, and J. Han. Tsp: Mining top-k closed sequential patterns. In *IEEE ICDM 2003*, pages 347–354
29. J. Wang and J. Han. Bide: Efficient mining of frequent closed sequences. In *IEEE ICDE'04*, pages 79–90, 2004
30. K. Wang, S. Zhou, and Y. He. Growing decision trees on support-less association rules. *In KDD'00, Boston, MA*, pages 265–269, 2000
31. X. Yan, J. Han, and R. Afshar. Clospan: Mining closed sequential patterns in large datasets. In *SIAM 2003*
32. Q. Yang, I.T.Y. Li, and K. Wang. Building association-rule based sequential classifiers for web-document prediction. In *DMKD 2004*, pages 253–273
33. M. Zaki. Generating non-redundant association rules. In *KDD 2000*, pages 34–43
34. M. Zaki and C.-J. Hsiao. Charm: An efficient algorithm for closed itemset mining. In *SIAM 2002*
35. M.J. Zaki. Spade: An efficient algorithm for mining frequent sequences. *Machine Learning (42)*, pages 31–60, 2001
36. M. Zhang, B. Kao, D.W. Cheung, and K.Y. Yip. Mining periodic patterns with gap requirement from sequences. In *ACM SIGMOD'05*, pages 623–633, 2005

An Algorithm for Mining Weighted Dense Maximal 1-Complete Regions

Haiyun Bian and Raj Bhatnagar

Department of ECECS, University of Cincinnati, Cincinnati, OH 45221, USA
bianh@ececs.uc.edu, raj.bhatnagar@uc.edu

Summary. We propose a new search algorithm for a special type of subspace clusters, called maximal 1-complete regions, from high dimensional binary valued datasets. Our algorithm is suitable for dense datasets, where the number of maximal 1-complete regions is much larger than the number of objects in the datasets. Unlike other algorithms that find clusters only in relatively dense subspaces, our algorithm finds clusters in all subspaces. We introduce the concept of weighted density in order to find interesting clusters in relatively sparse subspaces. Experimental results show that our algorithm is very efficient, and uses much less memory than other algorithms.

1 Introduction

Frequency has been used for finding interesting patterns in various data mining problems, such as the minimum support threshold used in mining frequent itemsets [2,3] and the minimum density defined in mining subspace clusters [1]. A priori-like algorithms [1] perform levelwise searches for all patterns having enough frequencies (either support or density) starting from single dimensions, and prune the search space based on the rationale that in order for a $k-$dimensional pattern to be frequent, all its $(k-1)-$dimensional sub-patterns must also be frequent. A large frequency threshold is usually set in most of the algorithms to control the exponential growth of the search space as a function of the highest dimensionality of the frequent patterns.

Closed patterns was proposed [7] to reduce the number of frequent patterns being returned by the algorithm without losing any information. Mining closed patterns is lossless in the sense that all frequent patterns can be inferred from the set of closed patterns. Most algorithms proposed for mining closed patterns require all candidates found so far to be kept in memory to avoid duplicates [9, 11, 12]. These algorithms also require the minimum frequency threshold value to be specified before the algorithms are run, and the same value is used to prune off candidates for patterns in all subspaces.

H. Bian and R. Bhatnagar: *An Algorithm for Mining Weighted Dense Maximal* 1*-Complete Regions*, Studies in Computational Intelligence (SCI) **118**, 31–48 (2008)
www.springerlink.com

Table 1. Subspaces with varied density

	a	b	c	d	e	f
1	0	0	0	0	0	1
2	0	0	0	0	0	1
3	0	0	0	0	1	1
4	0	0	0	1	1	1
5	0	0	0	1	1	1
6	0	0	0	1	1	1
7	0	0	0	1	1	0
8	0	0	0	1	0	0
9	1	1	1	0	0	0
10	1	1	1	0	0	0

However, patterns with higher dimensionality tend to have less frequencies, so using the same threshold value for all patterns risks losing patterns in higher dimensional spaces. Furthermore, patterns with the same dimensionality may need different frequency threshold values for various reasons. For example, a pattern with higher frequency in very dense dimensions may not be as informative and interesting as a pattern with lower frequency in very sparse dimensions. Setting a relatively high frequency threshold tends to bias the search algorithm to favor patterns in dense subspaces only, while patterns in less dense subspaces are neglected. Consider the example shown in Table 1. Each column denotes one of the six attributes (a, b, c, d, e, f), and each row denotes one object (data point). An entry '1' in row i and column j denotes that object i has attribute j. There is a pattern in subspace $\{abc\}$ that contains two instances $\{9, 10\}$, and subspace $\{def\}$ has another pattern containing three instances $\{4, 5, 6\}$. If we set the minimum frequency threshold to be 3, we lose the pattern in $\{abc\}$. However, this pattern in $\{abc\}$ maybe more interesting than the one in $\{def\}$, considering the fact that the number of '1's in attributes a, b, c is much smaller than in attributes d, e, f. Actually, all instances that have entry '1' in a also have entry '1' in b and c, and this may suggest a strong correlation between a, b, c, and also a strong correlation between instances 9 and 10. On the other hand, although the pattern in $\{def\}$ has a larger frequency, it does not suggest such strong correlations either between attributes d, e, f or between instances 4–6. So we suggest that smaller frequency threshold should be chosen for subspaces with lower densities, that is, subspaces with less number of '1' entries.

We propose a weighted density measure in this chapter, which captures the requirement to use a smaller density threshold for less dense subspaces. And we present an efficient search algorithm to find all patterns satisfying a minimum weighted density threshold.

Most algorithms for finding closed patters report only the dimensions in which the patterns occur, without explicitly listing all the objects that are contained in the patterns. However, the object space of the patterns is crucial

in interpreting the relationships between two possibly overlapping patterns. Our algorithm treats the objects and dimensions (attributes) equally, and all patterns are reported with their associated dimensions and subsets of objects.

Another advantage of our algorithm lies in its step-wise characteristic, that is, the computation of the next pattern depends only on the current pattern. Our algorithm is memory efficient due to this property, since there is no need to keep all previously generated patterns in the memory.

In the rest of the chapter, we present our algorithm in the context of the subspace clustering problem, but the algorithm and the theorem can also be applied to other closed set mining problems such as frequent closed itemsets [7] and maximal biclique [8]. We first present in Sect. 2 the definition of maximal 1-complete region, where we also introduce the terms and notations used in this chapter. Section 3 presents our algorithm. Section 4 presents the experimental results. Finally, we make the conclusion.

2 Problem Statement

A data space $\mathcal{DS}$ is characterized by a set of attributes $\mathcal{A}$ (attribute space) and a population of objects $\mathcal{O}$ (object space). Each object $o_i \in \mathcal{O}$ has a value assigned for each attributes $a_j \in \mathcal{A}$, denoted as d_{ij}. We consider only binary valued datasets in this chapter, that is, $d_{i,j} \in [0, 1]$. However, real valued datasets can be quantized into binary values, and different quantization methods lead to clusters of different semantics [6]. A subspace S is a subset of $\mathcal{A}$. A subspace cluster C is defined as $<O, A>$, where $O \subseteq \mathcal{O}$ and $A \subseteq \mathcal{A}$. We call O and A the *object set* and the *attribute set* of the subspace cluster respectively. Subspace clustering is a search for subsets of $\mathcal{P}(\mathcal{A})$ (the power set of $\mathcal{A}$) where interesting clusters exist.

2.1 The Prefix Tree of Subspaces

Let "$<_L$" be a lexicographic order on the attributes in $\mathcal{A}$, and we use $a_i <_L a_j$ to indicate that attribute a_i is lexicographically smaller than attribute a_j. Each subspace is represented as the set of attributes contained in it in the lexicographically increasing order. For example, a subspace containing attribute a_1, a_2, a_3 ($a_1 <_L a_2 <_L a_3$) is labeled as $\{a_1a_2a_3\}$. And we arrange all subspaces into a prefix-based tree structure $\mathcal{T}_{\mathcal{DS}}$ as follows:

1. Each node in the tree corresponds to one subspace, and the tree is rooted at the node corresponding to the empty subspace that contains no attributes.
2. For a node with label $S = \{a_1, \ldots, a_{k-1}, a_k\}$, its parent is the node whose label is $S' = \{a_1, \ldots, a_{k-1}\}$.

Table 2 shows an example dataset, and Fig. 1 shows its prefix tree of subspaces.

Table 2. An example data table

	a	b	c	d
1	0	0	1	1
2	1	0	1	1
3	1	1	1	0
4	0	0	1	1
5	1	1	0	0
6	0	0	1	1
7	0	0	1	1
8	0	1	0	0

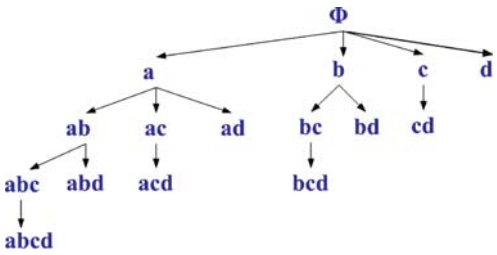


Fig. 1. Prefix-based subspaces search tree

2.2 Maximal 1-complete Regions and Closed Subspaces

We are interested in finding subspace clusters that contain largest regions of '1' entries, formally defined as follows:

Definition 1. *A subspace cluster* $C = <O, A>$ *of binary valued data space* $\mathcal{DS}$ *is a* 1-complete region *if it contains only '1' entries.*

Definition 2. *A complete dense region* $C = <O, A>$ *is a* maximal 1-complete region *if all regions that are proper super-regions of* C *are not* 1*-complete.*

For the example shown in Table 2, $<\{1, 2, 4, 6, 7\}, \{d\}>$ is a 1-complete region but it is not maximal, because its super-region $<\{1, 2, 4, 6, 7\}, \{cd\}>$ is 1-complete. $<\{1, 2, 4, 6, 7\}, \{cd\}>$ is a maximal 1-complete region, while $<\{1, 2, 3, 4, 6, 7\}, \{cd\}>$ is not 1-complete since it contains zero entries. If we consider each attribute (column) in Table 2 as a bit vector, all 1-complete regions can be found by intersecting all possible subsets of attributes. However, not all of them are maximal, so the problem is to find those subsets of attributes whose intersection produce maximal 1-complete regions.

Definition 3. *If a subspace is the attribute set of a maximal* 1*-complete region, we call this subspace a* closed subspace*.*

According on Definition 3, each maximal 1-complete region corresponds to one unique closed subspace. In order to find all maximal 1-complete regions, we can traverse the prefix tree of subspaces and check each node to see whether it is a closed subspace. In the following, we present several methods to test whether a subspace is closed. We first introduce two functions that perform mapping between the object space and the attribute space.

We define $\psi(S)$ to be $\{o_i | \forall a_j \in S, d_{ij} = 1\}$, that is, $\psi(S)$ returns the set of objects that have entry '1' for all the attributes in S. Similarly, $\varphi(O)$ is defined to be $\{a_j | \forall o_i \in O, d_{ij} = 1\}$, that is, $\varphi(O)$ returns the set of attributes that are shared by all objects in O. Then $\varphi \circ \psi$ is a closure operator, and we have the following lemma.

Lemma 1. *The following statements are equivalent:*

1. $C = <\psi(S), S>$ *is maximal* 1*-complete*
2. S *is a closed subspace*
3. $\nexists a \in \mathcal{A}/S$*, for which* $\psi(a) \supseteq \psi(S)$
4. $\varphi \circ \psi(S) = S$

Proof. $1 \leftrightarrow 2$: True by Definition 3.

$1 \rightarrow 3$: $C = <\psi(S), S>$ is maximal 1-complete means that we cannot add any attribute a to S to get an enlarged region, and at the same time maintain the 1-complete property. If there exists a for which $\psi(a) \supseteq \psi(S)$, then adding a to S will produce a region that has 1-complete property, which contradicts the fact that C is maximal 1-complete.

$3 \rightarrow 4$ and $4 \rightarrow 1$ can be proved similarly. □

From Lemma 1, we can see that $\varphi \circ \psi(S)$ is a superset of S if S is not closed, or equal to S if S is closed. Figure 2 shows a modified prefix tree from Fig. 1, where each node in Fig. 2 has two labels, including the corresponding subspace S and the object set $\psi(S)$. For example, node "$b, 358$" (Fig. 2) represents that this node corresponds to subspace $\{b\}$, for which $\psi(\{b\}) = \{3, 5, 8\}$. Underlined nodes are those 1-complete regions that are not maximal. Furthermore, nodes corresponding to subspaces with equal closure are grouped together into one equivalence class in Fig. 2. For example, $\varphi \circ \psi\{bc\} = \{abc\}$, so nodes "$bc$" and "$abc$" are grouped together. Notice that all equivalent subspaces have the same object set, so each equivalence class generates only one maximal 1-complete region. Therefore, we need only find one subspace for every such equivalence class in order to find all 1-complete regions.

2.3 The Lectical Order Between Subspaces

From Fig. 2, we can see that within each equivalence class, the closed subspace is always to the left of those non-closed ones. Based on this observation, we define a total order, called the *lectical* order, on the set of all subspaces. A similar definition can be found in [5]. A subspace S_1 is called *lectically smaller*

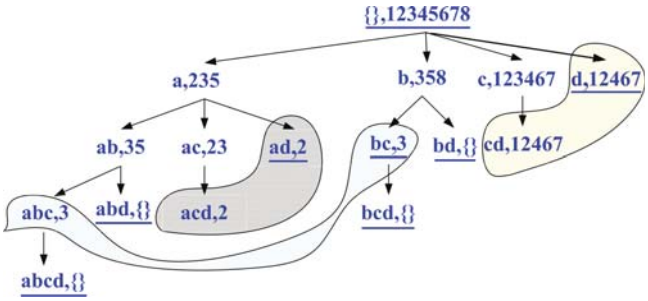


Fig. 2. Prefix tree of equivalence classes

than subspace S_2, denoted as $S_1 \ll S_2$, if the lexicographically smallest attribute a_i that distinguishes S_1 from S_2 belongs to S_2. That is, there exists $a_i \in S_2 \wedge a_i \notin S_1$, and all attributes lexicographically smaller than a_i are shared by S_1 and S_2. Formally,

$S_1 \ll S_2 :\Leftrightarrow \exists_{a_i \in S_2 \setminus S_1}\ S_1 \cap \{a_1, a_2, \ldots, a_{i-1}\} = S_2 \cap \{a_1, a_2, \ldots, a_{i-1}\}$.

If we know the attribute a_i that distinguishes S_1 and S_2, we say S_1 is i-smaller than S_2, denoted as $S_1 \ll_i S_2$.

For example, $\{ad\} \ll_c \{acd\}$ because the lexicographically smallest attribute that distinguishes them is c, and it belong to $\{acd\}$.

We define S^i to be a subset of S which includes all the attributes in S that are lexicographically smaller than a_i, that is, $S^i := S \cap \{a_1, \ldots, a_{i-1}\}$. Starting from an arbitrary subspace S, the next lectically smallest subspace that is larger than S can be computed based on Lemma 2.

Lemma 2. *The lectically smallest subspace that is lectically larger than S is $S^i \cup \{a_i\}$, where a_i is the lexicographically largest attribute that is not contained in S.*

Proof. Let $S_1 = S^i \cup \{a_i\}$, with a_i being the lexicographically largest attribute that is not contained in S. Suppose the lemma is not true, then there must exist S_2, such that $S \ll S_2 \ll S_1$. Since $S \ll S_2$, there must exist an attribute $a_j (i \neq j)$, which satisfies $a_j \in S_2$, $a_j \notin S$ and $S^{j-1} = S_2^{j-1}$. We also know that a_i is the smallest attribute that differentiates S and S_1, so $S^{i-1} = S_1^{i-1}$. We consider the following two possible relationships between a_i and a_j.

$a_i <_L a_j$: Since $i<j$, $S \ll_j S_2$ implies a_j is not contained in S, which contradicts the fact that a_i is the largest attribute not contained in S.
$a_i >_L a_j$: Since $i>j$, $S^{i-1} = S_1^{i-1}$ implies $S^{j-1} = S_1^{j-1}$. And we also have $S^{j-1} = S_2^{j-1}$, so $S_1^{j-1} = S_2^{j-1}$. Since the smallest attribute that differentiates S and S_1 is a_i, which is larger than a_j, so $a_j \notin S_1$. Since $S_1^{j-1} = S_2^{j-1}$, $a_j \in S_2$ and $a_j \notin S_1$, we have $S_1 \ll S_2$, which contradicts the assumption $S \ll S_2 \ll S_1$. □

Starting from the empty subspace, if we keep looking for the next lectically smallest subspace, we actually perform a right-to-left pre-order depth-first

traversal of the prefix tree. For the example shown in Fig. 1, the total lectical order is: $\{\phi\} \ll_d \{d\} \ll_c \{c\} \ll_d \{cd\} \ll_b \{b\} \ll_d \{bd\} \ll_c \{bc\} \ll_d \{bcd\} \ll_a \{a\} \ll_d \{ad\} \ll_c \{ac\} \ll_d \{acd\} \ll_b \{ab\} \ll_d \{abd\} \ll_c \{abc\} \ll_d \{abcd\}$.

The next question is how to find the next closed subspace after S. Let a_i be the lexicographically largest attribute that is not contained in S. If $S \cup \{a_i\}$ is a closed subspace, then it is trivial that $S \cup \{a_i\}$ is the next closed subspace. If $S \cup \{a_i\}$ is not closed, then its closure $\varphi \circ \psi(S^i \cup \{a_i\})$ must contain an attribute $a_j <_L a_i$ and $a_j \notin S$. To simplify the notation, we define $\mathbf{S} \oplus \mathbf{a_i} := \varphi \circ \psi(\mathbf{S^i} \cup \{\mathbf{a_i}\})$. Lemma 3 shows the method to find the next closed subspace after S.

Lemma 3. *The lectically smallest closed subspace that is lectically larger than S is $\varphi \circ \psi(S^i \cup \{a_i\})$, where a_i is the lexicographically largest attribute that is not contained in S for which $S \ll_i S \oplus a_i$ holds.*

A detailed proof for Lemma 3 can be found in [5]. Let a_i be the lexicographically largest attribute that is not contained in S for which $S \ll_i S \oplus a_i$ holds. Let a_k be an attribute $a_k \notin S$ and $a_k >_L a_i$. Since $S \not\ll_k S \oplus a_k$, $S \oplus a_k$ must contains at least one attribute that is lexicographically smaller than a_k. Let $S \ll_j S \oplus a_k$, that is, a_j is the lexicographically smallest attribute that differentiates S and $S \oplus a_k$. If $a_j <_L a_i$, then $S \oplus a_k$ is lectically larger than $S \oplus a_i$. If $a_j = a_i$, then $S \oplus a_i = S \oplus a_k$. If $a_j >_L a_i$, this contradicts the assumption that a_i is the lexicographically largest attribute that is not contained in S for which $S \ll_i S \oplus a_i$ holds. So in conclusion, Lemma 3 is true.

2.4 Density and Weighted Density

Notice that many nodes in Fig. 2 contain empty object set, which do not contribute to the clustering process. Furthermore, simply enumerating all maximal 1-complete regions is very time consuming. So we focus on finding those maximal 1-complete regions that contain at least a certain number of objects. Formally, we define the density of a single attribute a_i to be the ratio between the number of '1' entries in a_i and the total number of objects in the data, denoted as $dens(a_i)$. For the example shown in Table 2, $dens(d)$ is $\frac{5}{7}$ and $dens(a)$ is $\frac{3}{7}$. Similarly, the density of a subspace cluster is the ratio between the number of objects contained in it and the total number of objects in the data space. For example, the density of $<\{1,2,4,6,7\},\{cd\}>$ is $\frac{5}{7}$.

The weighted density of a subspace cluster $C = <O, A>$, denoted as $dens_w(C)$, is defined as the ratio between $dens(C)$ and the average density over all attributes contained in A, that is, $dens_w(C) = \frac{dens(C)}{\frac{1}{|A|}(\sum_{a_i \in A} dens(a_i))}$, where $|A|$ is the number of attributes contained in S. We call the denominator, $\frac{1}{|A|}(\sum_{a_i \in A} dens(a_i))$, the weight.

Since each subspace S has a unique closure $\varphi \circ \psi(S)$, which corresponds to exactly one maximal 1-complete region $C = <\psi(S), \varphi \circ \psi(S)>$, we define the density of subspace S ($dens(S)$) to be $dens(C)$, where C is the cluster having

the closure of S ($\varphi \circ \psi(S)$) as its attribute set. Similarly, $dens_w(S)$ is equal to $dens_w(C)$.

The next section presents the algorithm for finding all maximal 1-complete regions that have a weighted density larger than δ, where δ is a real number between 0 and 1.

3 Mining Weighted Dense Maximal 1-complete Regions

In this section, we present the underlying idea of our algorithm and the proof of correctness. Then we present some methods to speed up the algorithm.

3.1 Non-Decreasing Property

As shown in Fig. 2, density is non-increasing along any branches in the tree. This is because that the set of objects that are contained in a child node $S \cup \{a_i\}$ is the intersection of $\psi(\{a_i\})$ and the object set of its parent node S. Consequently, $\psi(S \cup \{a_i\})$ must be a subset of $\psi(S)$.

However, weighted density does not have this property. Although the density is non-increasing (numerator), the weights (denominator) may decrease when less dense attributes are added. If the decrease of the weights is faster than the decrease of density, weighted density of a child node may become larger than its parent node. One way to guarantee that weighted density is non-increasing along any branches is to enforce a constraint on the lexicographical order. More specifically, we sort all the attributes into the increasing density order, such that the lexicographically largest attribute is the one that has the largest density. By doing this, we can make sure that when we go deeper into the tree, the weights never decrease. Therefore, weighted density along any branches of the tree must also be non-increasing. This property facilitates the search algorithm that is introduced later.

In the remaining of the chapter, we assume that the data has been sorted this way. For the data shown in Table 2, the sorted dataset is shown in Table 3.

Table 3. Sorted example

	a	b	c	d
1	0	0	1	1
2	1	0	1	1
3	1	1	0	1
4	0	0	1	1
5	1	1	0	0
6	0	0	1	1
7	0	0	1	1
8	0	1	0	0

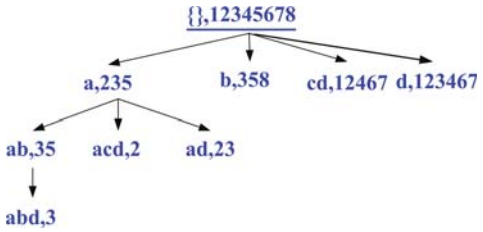


Fig. 3. 1-complete regions for sorted data

Figure 3 is a tree containing all and only the maximal 1-complete regions in this sorted dataset. Ideally, we only need to check all and only the nodes in Fig. 3, which is much smaller than the number of nodes contained in the complete tree as shown in Fig. 1.

3.2 Mining Weighted Dense 1-complete Regions

To better explain the algorithm, we first show the underlying idea and the correctness proof of our approach. Lemma 4 states that under certain condition, applying the "$\oplus a_i$" operator multiple times has the same effect as applying only once.

Lemma 4. $S \ll_i S \oplus a_i \rightarrow S \oplus a_i \oplus a_i = S \oplus a_i$.

Rationale. By definition, $S \ll_i S \oplus i$ means that $S \cap \{a_1, a_2, \ldots, a_{i-1}\} \cup \{a_i\} = S \oplus a_i \cap \{a_1, a_2, \ldots, a_{i-1}\} \cup \{a_i\}$. Since $S \oplus a_i \oplus a_i = \varphi \circ \psi(S \oplus a_i \cap \{a_1, a_2, \ldots, a_{i-1}\} \cup \{a_i\})$, and $\varphi \circ \psi(S \cap \{a_1, a_2, \ldots, a_{i-1}\} \cup \{a_i\}) = S \oplus a_i$, we have $S \oplus a_i \oplus a_i = S \oplus a_i$.

Lemma 5. $S \ll_i S \oplus a_i$ *and* $a_j >_L a_i \rightarrow S \oplus a_i \oplus a_j \supset S \oplus a_i$.

Rationale. $S \oplus a_i \oplus a_j = \varphi \circ \psi(S \oplus a_i \cap \{a_1, a_2, \ldots, a_{j-1}\} \cup \{a_j\})$. Since $a_i \in S \oplus a_i$ and $a_j >_L a_i$, $S \oplus a_i \cap \{a_1, a_2, \ldots, a_{j-1}\} \cup \{a_j\} \supset S \oplus a_i \cap \{a_1, a_2, \ldots, a_{i-1}\} \cup \{a_i\}$. This implies $\varphi \circ \psi(S \oplus a_i \cap \{a_1, a_2, \ldots, a_{j-1}\} \cup \{a_j\}) \supset \varphi \circ \psi(S \oplus a_i \cap \{a_1, a_2, \ldots, a_{i-1}\} \cup \{a_i\})$, which is equivalent to $S \oplus a_i \oplus a_j \supset S \oplus a_i$.

The implication of Lemma 5 is that if a 1-complete region C_1 in subspace $S \oplus a_i$ does not have enough density, then there is no need to check any attribute $a_j >_L a_i$. This is because Lemma 5 proves that $S \oplus a_i \oplus a_j$ is a superset of $S \oplus a_i$, thus the cluster C_2 in $S \oplus a_i \oplus a_j$ must have a density less than $dens(C_1)$. Furthermore, since the weights is non-decreasing along any branches after we sort the attributes into increasing density, $dens_w(C_2)$ must also be less than $dens_w(C_1)$. Thus if we know that $dens_w(C_1) < \delta$, $S \oplus a_i \oplus a_j$ can be safely pruned. Similarly, we can prove the following Lemma 6 by induction.

Lemma 6. $S \ll_i S \oplus a_i$ *and* $a_{k_m} >_L \ldots >_L a_{k_2} >_L a_{k_1} >_L a_i \rightarrow S \oplus a_i \oplus a_{k_1} \ldots \oplus a_{k_m} \supset S \oplus a_i$.

Lemma 6 tells us that if a 1-complete region C_1 in subspace $S \oplus a_i$ does not have enough weighted density, we can directly jump to test $S \oplus a_j$ for $a_j <_L a_i$ because anything in between must not meet the minimum weighted density threshold, which leads to Theorem 1.

Theorem 1. *The lectical smallest closed subspace larger than a given subspace* $S \subset \mathcal{A}$ *and having weighted density larger than* δ *is* $S \oplus a_i$*, where* a_i *is the lexicographically largest attribute which satisfies* $dens_w(S \oplus a_i) > \delta$ *and* $S \ll_i S \oplus a_i$*.*

Rationale. Let $S \oplus a_j$ be the lectically smallest closed subspace that is larger than S. If $dens_w(S \oplus a_j) > \delta$, the theorem is true since it is the same case as in Lemma 3. If $dens_w(S \oplus a_j) < \delta$, let a_i be the largest attribute for which $a_i <_L a_j$ and $S \ll_i S \oplus a_i$ hold. So we need to show that $S \oplus a_j \oplus a_i$ is the lectically smallest closed subspace that is larger than $S \oplus a_j$, and potentially could have enough weighted density. Since $dens_w(S \oplus a_j) < \delta$, Lemma 6 guarantees the search to start with a_{j-1} for the smallest weighted dense cluster. Since $S \ll_j S \oplus a_j$, $S \cap \{a_1, \ldots, a_{j-1}\} = S \oplus a_j \cap \{a_1, \ldots, a_{j-1}\}$. So the search for the next a_i performs the same on S and $S \oplus a_j$, that is, $S \oplus a_i = S \oplus a_j \oplus a_i$. So $S \oplus a_i$ is the lectically smallest closed subspace that is larger than S and could have enough weighted density. If $dens(S \oplus a_i) > \delta$, this theorem is true. Otherwise, find the next $a_k <_L a_i$ for which $S \ll_k S \oplus a_k$, and the proof can be completed inductively.

3.3 Lectical Weighted Dense Region Mining Algorithm

Theorem 1 states that if we find that a subspace $S \oplus a_i$ is not weighted dense, we can prune the search space by skipping all $a_j >_L a_i$, and check directly on a_{i-1} in the next iteration of the algorithm. Algorithm 1 is a straightforward implementation of this idea. Based on the correctness of Theorem 1, we can conclude the correctness of Theorem 2.

Algorithm 1 Lectical weighted dense region mining algorithm

```
1. C = <O, S> ← <ψ(φ), φ ∘ ψ(φ)>
2. IF (dens_w(C) > δ)
3.    Add C = <O, S> to Tree
4. found ← true
5. END IF
6. REPEAT
7.    (C, found) ← findnext(C)
8. UNTIL found = false
```

Theorem 2. *Algorithm 1 finds all maximal* 1*-complete regions that satisfy the minimum weighted density threshold* δ*.*

FUNCTION: findnext(C)

1. $found \leftarrow FALSE$
2. $o \leftarrow lexicographlly - largest(\{i | a_i \notin S\})$
3. **WHILE** $(!found)$ **AND** $(o \geq 0)$
4. **IF** $(a_o \notin S)$
5. $\overline{C} = <\overline{O}, \overline{S}> \leftarrow <\psi(S \oplus a_o), S \oplus a_o>$
6. **IF** $(dens_w(\overline{C}) > \delta)$ **AND** $(S \ll_o \overline{S})$
7. $found \leftarrow TRUE$
8. Add $\overline{C} = <(\overline{O}), \overline{S}>$ to Tree
9. **END IF**
10. **END IF**
11. $o \leftarrow o - 1$
12. **END WHILE**
13. **RETURN** ($\overline{C}$, found)

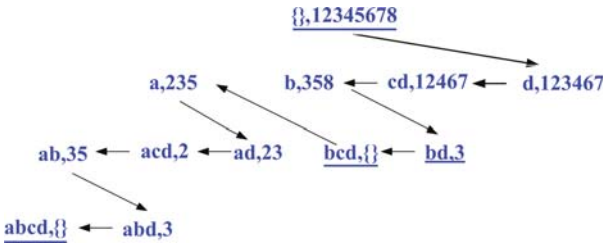


Fig. 4. Search tree of sorted data

The search starts out by finding the closure of the empty subspace (line 1), and adding that to the tree of closed subspace if it has enough weighted density (line 2–3). Then the algorithm keeps looking for the next lectically larger closed subspace satisfying the weighted density constraint until no more such subspaces can be found (line 6–8).

Function $findnextbasic$ accepts a 1-complete region C as parameter, and returns the next lectically smallest closed and weighted dense subspace and its corresponding maximal 1-complete region. First the flag $found$ is set to be false. Starting from the lexicographically largest attribute not contained in the current subspace S, it looks for an attribute a_o that meets the two conditions at line 6. The loop terminates either with a successful candidate or when all the possibilities have been tried (line 3).

Figure 4 traces the algorithm on the dataset shown in Table 3 with $\delta = 0$ (no weighted density pruning). Nodes in the tree are those being visited. Underlined nodes are non-maximal ones. The arrows indicate the sequence of visiting. Suppose we start from node $S = \phi$. Since the current subspace is empty, the largest attribute not contained in S is d. Then we compute the $S \oplus \{d\} = \{d\}$. Since $S \ll_d S \oplus \{d\}$, we output cluster $<\{123467\}, \{d\}>$ and keep looking for the next one.

3.4 Optimizing Techniques

In this section, we present several methods to optimize the time complexity of the basic algorithm. The data is stored as bit strings, that is, each attribute is represented as a string of 0 and 1. The major operation of our algorithm is bit intersection. When the percentage of 1 entries in the dataset is larger than 10%, using bit strings not only saves memory space, it also makes the computations more efficient.

Reuse Previous Results in Computing $\overline{O}$

The most expensive operation in Function $findnext$ is at line 5, where we need to compute the $\overline{S} = S \oplus a_o$ and its object set $\overline{O} = \psi(S \oplus a_o)$. Notice that for any node in the prefix tree as shown in Fig. 2, its object set can be computed incrementally from the object set of its parent. That is, the object of the child node is the intersection of the object set of the parent node and $\psi(a_o)$, where a_o is the newly added attribute. For example, the object set of cd can be computed by taking the intersection of the object set of its parent node $c(\{123467\})$ and $\psi(d)$ $(\{12467\})$. So we can maintain the object sets of all the nodes on the current branch of the search tree on a stack called $curPath$ to avoid duplicated intersection operations.

However, when the search moves from one branch to the other, the stack $curpath$ needs to be updated to maintain the correctness of the object set computation. For example, after we visited node ad, the next node to be visited is ac. But the object set of ac can not be incrementally computed based on the object set of ad, while it can be computed incrementally based on the object set of a. So we maintain another stack of attribute id called $istack$, which keeps track of all the attribute id for which $S \ll_o S \oplus a_o$ is true. For example, after we find that the next closed subspace after $\varphi \circ \psi(\phi)$ is $<\{123467\}, c>$, we push the object set into $curPath$ and we push c into stack $istack$. When we try to find the next closed subspace after c, we check if o is larger than the top of $istack$. If yes, that means that we are still on the same branch of the search tree, so there is no need to change the stack; if no, that means that we are jumping to a different branch, so pop up all the elements in $iStack$ that is larger than o. When popping out the elements in $iStack$, $curPath$ is also updated in the similar fashion. That is, whenever pop out an element from $iStack$, we also pop out an element from $curPath$.

Stack of Unpromising Nodes

Observe the search tree in Fig. 2. Starting from node ϕ, we first check if $\phi \ll_d \phi \oplus d$. Since $\phi \oplus d = \{cd\}$, we know that any closed subspaces that contain d must also contain c. So, after we reach node $\{a\}$, there is no need to check $\{ad\}$, since we know for sure that it can not be closed. For this type of pruning, we maintain a stack called $prelistStack$. This stack contains elements called

prelist, and for each attribute i, $prelist[i]$ is the id of the lexicographically smallest attribute j for which $\psi(j) \supseteq \psi(i)$. Initially set all $prelist[i] = i$. During the search algorithm, set the elements accordingly. Similar to $curPath$ and *iStack*, *prelist* needs to be updated when we jump between branches.

4 Experimental Results

We tested our algorithm on three datasets as listed in Table 4, which includes the name of the dataset, number of objects, number of attributes, minimum density of the attributes, and maximal density of the attributes. Mushroom and Chess are from [4], and Cog is from [10].[1] The objective of the experiments is to show that our algorithm can indeed find clusters both from dense subspaces and relatively sparse subspaces. All our experiments were performed on 2.4 GHz Pentium PC with 512 MB memory running Windows 2000.

All test data are very dense in the sense that the number of maximal 1-complete regions contained in the datasets is much larger than the number of objects in the datasets. Another feature of these data is that their attributes have quite different densities. Mushroom contains 129 attributes and 8,124 objects, while the most dense attribute contains all '1's and the least dense attribute contains only four '1's. The other two datasets have similar characteristics. Figure 5 shows the density distribution of the attributes for all the three datasets. For the Chess dataset, around 30% of the attributes have density less than 20%. If we set the minimum density to be 20%, we will not be able to find any patterns in almost one thirds of the subspaces. One possible solution to find patterns in these less dense subspaces is to reduce the minimum density threshold to less than 20%. However, reducing the minimum density threshold leads to an exponential growth in the total number of clusters being found, most of which belong to the more dense subspaces. So we perform the following experiments to show that our algorithm can find weighted dense 1-complete regions in both dense subspaces and sparse subspaces.

We compared our algorithm with $CLOSET+$ [11], which is an enhanced version of $CLOSET$ [9]. For $CLOSET+$, a very small minimum density threshold value is needed in order to find those weighted dense clusters in the less dense subspaces. We set the minimum density threshold for $CLOSET+$ to be a value such that it can find all weighted dense regions larger than a

Table 4. Datasets characteristics

	# Of objs	# Of attrs	Minimum density	Maximum density
Mushroom	8,124	129	0.01	1
Chess	3,196	75	0.03	1
COG	3,307	43	0.11	0.60

[1] Cog stands for clusters of orthologous genes.

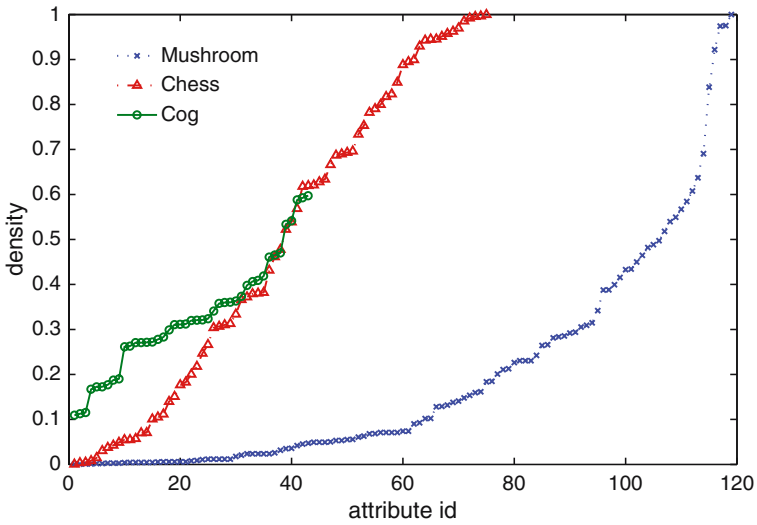


Fig. 5. Density distribution for all attributes

certain threshold value. For example, the least dense attribute in COG has density 0.11. If we want to find weighted dense clusters that have this least dense attribute, the minimum density threshold must be set to be no larger than 0.11. However, our tests show that for Chess and COG, $CLOSET+$ runs out of memory for these low threshold values. For Mushroom, $CLOSET+$ can finish the mining task for all threshold values.

Our algorithm uses almost the same amount of memory for all weighted density threshold values, since the computation of the next cluster depends only on the current cluster and not on any other previously found ones. As shown in Fig. 6, our algorithm uses almost the same amount of memory for all weighted density threshold values for all datasets. Compared with $CLOSET+$, our algorithms uses much less memory on Mushroom. For Chess and COG, the difference is more significant as $CLOSET+$ cannot finish the task due to insufficient memory.

We also compared the running time of our algorithm with $CLOSET+$ on the Mushroom data. Since $CLOSET+$ runs out of memory on Chess and Cog, we only report the running time for our algorithm. In order to find weighted dense clusters in the least dense subspaces, $CLOSET+$ needs to find almost all dense regions, which explains why its running time is almost constant for all threshold values. Even if we want to find all the maximal 1-complete regions in the data, our algorithm is still faster than $CLOSET+$.

Figure 9 shows the total number of clusters being found for various weighted density threshold values. For all three datasets, the running time curves as shown in Figs. 7 and 8 fit very well with the curves in Fig. 9. This suggests that our algorithm has a linear time complexity with the number of clusters being found.

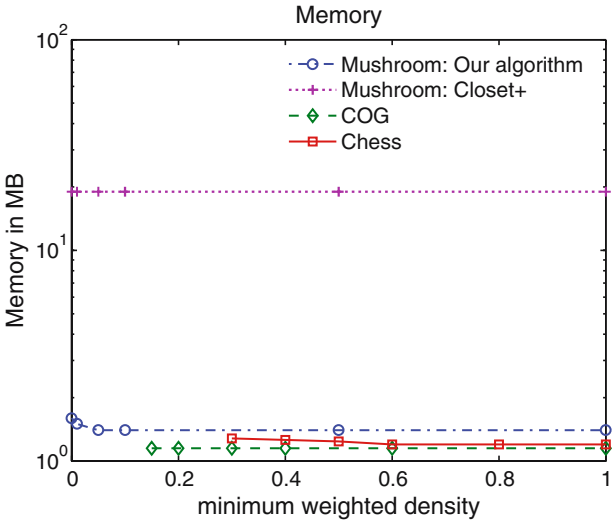


Fig. 6. Memory comparison

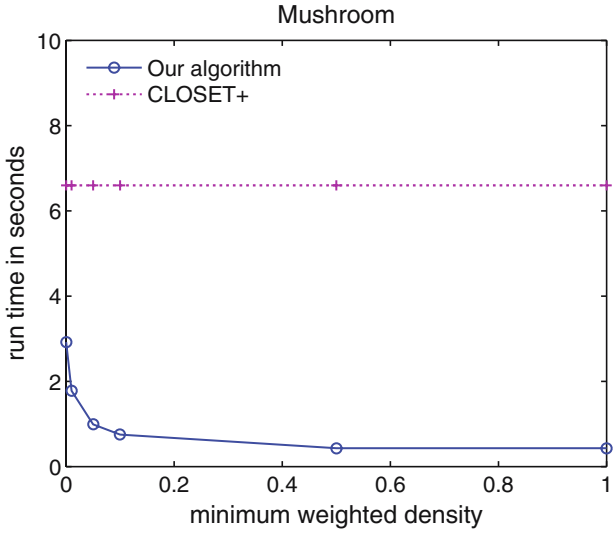


Fig. 7. Mushroom time comparison

We also want to show through experiments that using weighted density can find more clusters in less dense subspaces. So we compared the results from density pruning with the results from weighted density pruning. For fair comparison, we only compare when the minimum density threshold and the minimum weighted density threshold are equally selective, that is, there are equal number of clusters that satisfy each of the constraint. Figure 10 shows the percentage of the clusters being found after each attribute id on COG

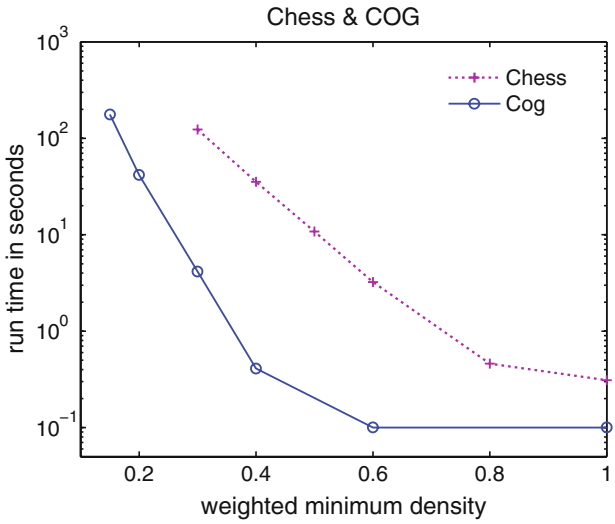


Fig. 8. Chess and COG run time comparison

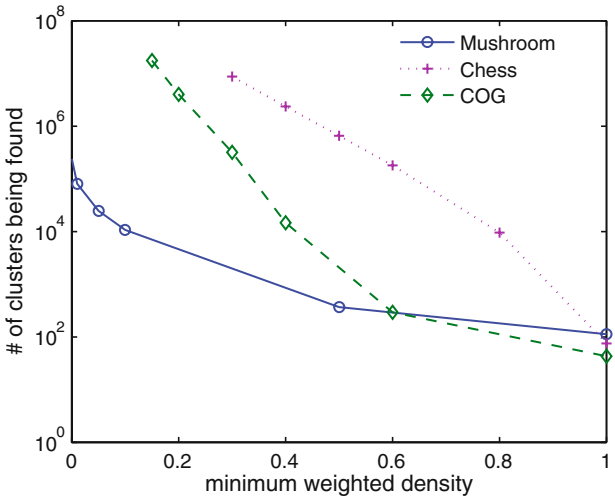


Fig. 9. Total number of clusters being found

when there are 10,000 clusters being found. Attributes are numbered such that more dense larger attributes have larger ids. The search starts from the attribute that has the largest id (45 in this case), and ends when it finishes attribute 0. From the figure we can see that when using weighted density, more clusters in the less dense subspaces are returned. Close examination reveals that using minimum density threshold, seven attributes are not included in any clusters. On the other hand, using weighted density, all attributes belong

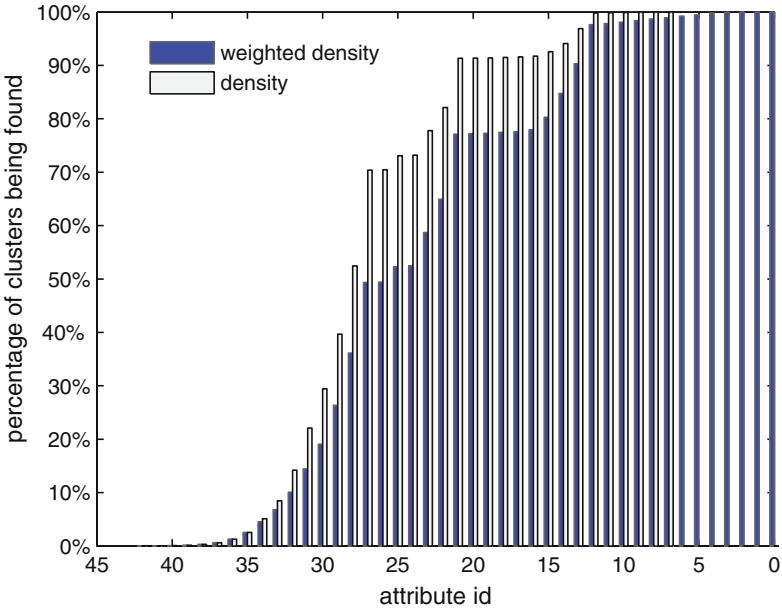


Fig. 10. Comparison between density and weighted density

to at least one cluster. We tested a set of different selective threshold values on all three datasets, and all of them confirms that using the weighted density constraint finds more clusters in less dense subspaces.

5 Conclusion

We have presented a new subspace clustering mining algorithm to find weighted dense maximal 1-complete regions in high dimensional datasets. Our algorithm is very memory efficient, since it does not need to keep all the clusters found so far in the memory. Unlike other density mining algorithms which tend to find only patterns in the dense subspaces while ignore patterns in less dense subspaces, our algorithm finds clusters in subspaces of all densities. Our experiments showed that our algorithm is more efficient than $CLOSET+$ from both time complexity and memory consumption perspectives.

References

1. Agrawal, R., Gehrke, J., Gunopulos, D., Raghavan, P.: Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications, *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data (SIGMOD'98)*, ACM, New York, June 1998, 94–105

2. Agrawal, R., Imielinski, T., Swami, A.: Mining Association Rules Between Sets of Items in Large Databases, *Proceedings of the 1993 ACM SIGMOD International Conference on Management of data (SIGMOD'93)*, ACM, New York, May 1993, 207–216
3. Agrawal, R., Srikant, R.: *Fast Algorithms for Mining Association Rules*, Morgan Kaufmann, Los Altos, CA, 1998, 580–592
4. Blake, C., Merz, C.: UCI Repository of machine learning databases, http://www.ics.uci.edu/∼mlearn/MLRepository.html, 1998
5. Ganter, B., Kuznetsov, S.O.: Stepwise Construction of the Dedekind–MacNeille Completion, *Proceedings of Sixth International Conference on Conceptual Structures (ICCS'98)*, August 1998, 295–302
6. Ganter, B., Wille, R.: *Formal Concept Analysis: Mathematical Foundations*, Springer, Berlin Heidelberg New York, 1999
7. Pasquier, N., Bastide, Y., Taouil, R., Lakhal, L.: Discovering Frequent Closed Itemsets for Association Rules, *Proceeding of the Seventh International Conference on Database Theory (ICDT'99)*, Springer, Berlin Heidelberg New York, January 1999, 398–416
8. Peeters, R.: The maximum edge biclique problem is NP-complete, *Discrete Applied Mathematics*, **131**, 2003, 651–654
9. Pei, J., Han, J., Mao, R.: CLOSET: An Efficient Algorithm for Mining Frequent Closed Itemsets, *ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, May 2000, 21–30
10. Roman, T., Natalie, F., et al.: The COG database: an updated version includes eukaryotes, *BMC Bioinformatics*, **4**, September 2003
11. Wang, J., Han, J., Pei, J.: CLOSET+: Searching for the Best Strategies for Mining Frequent Closed Itemsets, *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'03)*, ACM, New York, 2003, 236–245
12. Zaki, M.J., Hsiao, C.J.: Charm: an Efficient Algorithm for Closed Itemset Mining, *Proceedings of the Second SIAM International Conference on Data Mining (SDM'04)*, April 2002

Mining Linguistic Trends from Time Series

Chun-Hao Chen[1], Tzung-Pei Hong[2], and Vincent S. Tseng[3]

[1] Department of Computer Science and Information Engineering, National Cheng-Kung University, Tainan, Taiwan, ROC
chchen@idb.csie.edu.tw
[2] Department of Electrical Engineering, National University of Kaohsiung
tphong@nuk.edu.tw
[3] Department of Computer Science and Information Engineering, National Cheng-Kung University, Tainan, Taiwan, ROC
vincent@idb.csie.ncku.edu.tw

Summary. In this chapter, we propose a mining algorithm based on angles of adjacent points in a time series to find linguistic trends. The proposed approach first transforms data values into angles, and then uses a sliding window to generate continues subsequences from angular series. Several fuzzy sets for angles are predefined to represent semantic concepts understandable to human being. The a priori-like fuzzy mining algorithm is then used to generate linguistic trends. Appropriate post-processing is also performed to remove redundant patterns. Finally, experiments are made for different parameter settings, with experimental results showing that the proposed algorithm can actually work.

1 Introduction

Time-series data are commonly seen around our daily life. They are the data recorded at each time interval. For example, the stock prices evolving over a period of time are an example of a time series. Many sets of data in the fields like telecommunication, bioinformatics and medical treatment, are time series data.

Finding useful patterns from time-series data has recently become an important issue for researchers in the data-mining fields. Indyk et al. focused on the problem of identifying representative trends, such as relaxed periods and average trends over a period of observations in time series [8]. They first generated a template set of sketches by using polynomial convolution, where each sketch is a low dimensional vector. The sketches were then used to replace each interval to find representative trends. Patel et al. proposed a method based on Euclidean distance to find k-motifs, which mean frequently occurring patterns in time series [11]. They first normalized time series data, and then used approximated piecewise aggregation to reduce data dimension [9, 15]. After the

C.-H. Chen et al.: *Mining Linguistic Trends from Time Series*, Studies in Computational Intelligence (SCI) **118**, 49–60 (2008)
www.springerlink.com

data dimension was reduced, they further transformed the data into a discrete representation and mined k-motifs from the transformed time series. Agrawal et al. proposed an algorithm to capture the shapes from historical time-series database by using a simple translation [2]. They first transformed the difference value of every two adjacent data points into a predefined category, such as increase, steep increase, steep decrease, decrease, no-change, and zero. The same time series may be labeled more than one category. In other words, the intervals among these categories have overlapped a little. The transformed symbolic series were then used for querying desired results.

Most of the above approaches, however, usually require predefined crisp intervals for each category. It thus needs domain knowledge and depends on applications. Udechukwu et al. thus proposed a domain-independent trend-encoding method to mine frequent trends [13]. They transformed the difference value between two adjacent data points into an angle, instead of the difference value itself. The angles lay within the range -90^0 to 90^0, and were partitioned into 52 predefined angular categories, represented by letters. They then used the data structure of suffix trees to find the maximally repeated patterns as frequent trends. In this way, the effect of the domain knowledge could be reduced. Their approach, however, had too many angular categories, which might cause users hard to understand the meaning of the patterns easily.

As to fuzzy data mining, Hong et al. proposed several fuzzy mining algorithms to mine linguistic association rules from quantitative data [6, 7, 10]. They transformed each quantitative item into a fuzzy set and used fuzzy operations to find fuzzy rules. Their approaches, however, focused on transaction data. For time-series data, Song et al. proposed a fuzzy stochastic time series and built a model by assuming the values are fuzzy sets [12]. Chen et al. proposed a two-factor time-variant fuzzy time-series model to deal with forecasting problems [4]. Au and Chan proposed a fuzzy mining approach to find fuzzy rules for classifying time-series [1]. Watanabe exploited the Takagi–Sugeno model to build a time-series model [14].

In this chapter, we thus propose a mining algorithm based on angles of adjacent points in a time series to find linguistic trends. Several fuzzy sets for angles are predefined to represent semantic concepts understandable to human being. The a priori-like fuzzy mining algorithm is then used to generate linguistic trends. Appropriate post-processing is also performed to remove redundant patterns. Since the final results are represented by linguistic terms, they will be friendlier to human than quantitative representation.

2 Mining Linguistic Trends for Time Series

The proposed fuzzy mining algorithm integrates the fuzzy sets, the a priori mining algorithm and the time-series concepts to find out appropriate linguistic trends from a time series. The proposed approach first transforms data values into angles, and then uses a sliding window to generate continuous

subsequences from the transformed angular series. Several fuzzy sets for angles are predefined to represent semantic concepts understandable to human being. Finally, an a priori-like fuzzy mining algorithm is proposed to generate linguistic trends. Appropriate post-processing is also performed to remove redundant patterns. Details of the proposed mining algorithm are described below.

The proposed mining algorithm for linguistic trends:

INPUT: A time series S with k data points, a set of h membership functions for angles, a predefined minimum support α, and a sliding-window size w.

OUTPUT: A set of linguistic trends.

STEP 1: Transform every two adjacent data points in the time series S into an angle. Assume $S = (d_1,\ d_2,\ d_3, \ldots,\ d_k)$. Then the resulting angular series S' is formed as:

$$S' = (a_1, a_2, a_3, \ \ldots\ , a_{k-1}),$$

where a_i is the angle from data point d_i to d_{i+1}.

STEP 2: Transform S' into a set of subsequences $W(S)$ according to the sliding-window size w. That is,

$$W(S') = \{s_p | s_p = (a_p, a_{p+1}, \ \ldots\ , a_{p+w-1}), p = 1\ to\ k - w\},$$

where a_p is the value of the p-th angle in S'.

STEP 3: Transform the j-th ($j = 1$ to w) quantitative value (angle) v_{pj} in each subsequence s_p ($p = 1$ to k-w) into a fuzzy set f_{pj}, represented as:

$$\left(\frac{f_{pj1}}{R_{j1}}, \frac{f_{pj2}}{R_{j2}}, \ \cdots\ , \frac{f_{pjh}}{R_{pjh}}\right),$$

using the given membership functions, where R_{jl} is the l-th fuzzy region of the j-th data point in each subsequence, h is the number of fuzzy memberships, and f_{pjl} is v_{pj}'s fuzzy membership value in region R_{jl}. Each R_{jl} is called a fuzzy term.

STEP 4: Calculate the scalar cardinality of each fuzzy term R_{jl} as:

$$count_{jl} = \sum_{p=1}^{k-w} f_{pjl}.$$

STEP 5: Collect each fuzzy term to form the candidate 1-patternsets C_1.

STEP 6: Check whether the support ($=count_{jl}/(k - w)$) of each R_{jl} in C_1 is larger than or equal to the predefined minimum support value. If R_{jl} satisfies the above condition, put it in the set of large 1-pattern-sets (L_1). That is:

$$L_1 = \{R_{jl} | count_{jl} \geq \alpha, 1 \leq j \leq p + w - 1\ and\ 1 \leq l \leq h\}.$$

STEP 7: If L_1 is not null, then do the next step; otherwise, exit the algorithm.

STEP 8: Set $r = 1$, where r is used to represent the number of fuzzy terms in the current pattern-sets to be processed.

STEP 9: Join the large r-pattern-sets L_r to generate the candidate $(r+1)$-pattern-set C_{r+1} in a way similar to that in the a priori algorithm [3] except that two items generated from the same order of data points in subsequences cannot simultaneously exist in a pattern in C_{r+1}. Besides, the first $(r\text{-}1)$-subpattern in a large r-pattern must be the same with the last $(r\text{-}1)$-subpattern in another r-pattern to form a candidate $(r+1)$-pattern in C_{r+1}.

STEP 10: Do the following substeps for each newly formed $(r+1)$-pattern I with fuzzy terms $(I_1, I_2, \ldots, I_{r+1})$ in C_{r+1}:

STEP 10.1 Calculate the fuzzy value of I in each subsequence s_p as $f_I^{(s_p)} = f_{I_1}^{(s_p)} \wedge f_{I_2}^{(s_p)} \wedge, \ldots, \wedge f_{I_{r+1}}^{(s_p)}$, where $f_{I_j}^{(s_p)}$ is the membership value of fuzzy pattern I_j in s_p. If the minimum operator is used for the intersection, then:

$$f_I^{(s_p)} = Min_{j=1}^{r+1} f_{I_j}^{(s_p)}$$

STEP 10.2 Calculate the count of I in all subsequences as:

$$count_I = \sum_{p=1}^{k-w} f_I^{(S_p)}.$$

STEP 10.3 If the support of I is larger than or equal to the predefined minimum support value α, put it in L_{r+1}.

STEP 11: IF L_{r+1} is null, then do the next step; otherwise, set $r = r + 1$ and repeat Steps 8–10.

STEP 12: Shift each large pattern $(I_1, I_2, \ldots, I_q)$, $q \geq 2$, into $(I'_1, I'_2, \ldots, I'_q)$, such that the fuzzy region R_{jl} in I_1 will become R_{1l} in I'_1 and a fuzzy region R_{it} in the other items will become $R_{(i-j+1)t}$, where R_{jl} is the l-th fuzzy region of the j-th data point in each subsequence.

STEP 13: Remove redundant large patterns from the results after Step 12.

STEP 14: Output the maximally large patterns generated from Step 13 as the linguistic trends.

3 An Example

In this section, a simple example is given to show how the proposed algorithm can generate fuzzy linguistic trends from the given time series. Assume the data points in a time series are shown in Table 1. The time series in Table 1 contains 13 data points. Each data point represents a value at a certain time. For example, the second data point in the time series means the value obtained at time 2 is 4.

Table 1. The time series used in this example

Time series
3, 4, 7, 9, 8, 3, 2, 4, 8, 10, 8, 4, 2

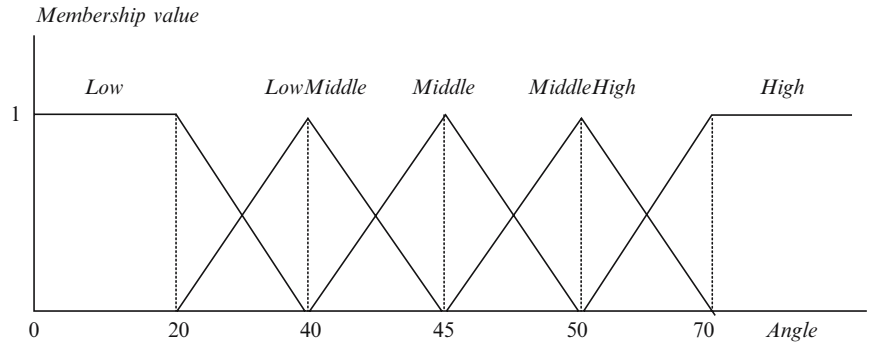


Fig. 1. The membership functions for angles

Table 2. The transformed angular series from Table 1

Angular sequence
45, 71.56, 63.43, −45, −78.96, −45, 63.43, 75.96, 63.43, −63.43, −75.96, −63.43

The range of the angles is between -90^0 and 90^0. Assume the membership functions for the angular values are defined as shown in Fig. 1. There are five fuzzy membership functions, represented as linguistic terms, *Low*, *LowMiddle*, *Middle*, *MiddleHigh* and *High*, for positive angles. There are another five membership functions, represented as *low*, *lowmiddle*, *middle*, *middlehigh* and *high*, for negative angles. Thus, the uppercase initial letter means the angle is positive, and the lowercase initial letter means the angle is negative.

For the time series given in Table 1, the proposed fuzzy mining algorithm proceeds as follows.

STEP 1: Every two adjacent data points in Table 1 are transformed into an angle. The results are shown in Table 2.

STEP 2: The angular series is then used to generate a set of subsequences according to the predefined window size. Assume the given window size is 6. There are totally 7 $(=13-6)$ subsequences to be obtained. The results are shown in Table 3.

STEP 3: The angular values in each subsequence are then transformed into fuzzy sets according to the membership functions given in Fig. 1. Take the first value v_{11} (=45) in the subsequence s_1 as an example. The value "45" is converted into the fuzzy set $(1.0/P_1.m)$, where $P_i.term$ is a fuzzy region of the i-th data in the subsequences and is called a fuzzy term. For example, $P_1.m$ represents the fuzzy region *middle* of the first data point in each subsequence.

Table 3. The transformed angular series from Table 1

S_p	Subsequence
S_1	(45, 71.56, 63.43, −45, −78.96, −45)
S_2	(71.56, 63.43, −45, −78.96, −45, 63.43)
S_3	(63.43, −45, −78.96, −45, 63.43, 75.96)
S_4	(−45, −78.96, −45, 63.43, 75.96, 63.43)
S_5	(−78.96, −45, 63.43, 75.96, 63.43, −63.43)
S_6	(−45, 63.43, 75.96, 63.43, −63.43, −75.96)
S_7	(63.43, 75.96, 63.43, −63.43, −75.96, −63.43)

Table 4. The fuzzy sets transformed from the data in Table 3

S_p	P_1	P_2	P_3	P_4	P_5	P_6
S_1	$P_1.m(1)$	$P_2.h(.92)$	$P_3.mh(.32)$ $P_3.h(.67)$	$P_4.M(1)$	$P_5.H(.56)$	$P_6.M(1)$
S_2	$P_1.h(.92)$	$P_2.mh(.32)$ $P_2.h(.67)$	$P_3.M(1)$	$P_4.H(.56)$	$P_5.M(1)$	$P_6.mh(.32)$ $P_6.h(.67)$
S_3	$P_1.mh(.32)$ $P_1.h(.67)$	$P_2.M(1)$	$P_3.H(.56)$	$P_4.M(1)$	$P_5.mh(.32)$ $P_5.h(.67)$	$P_6.h(.70)$
S_4	$P_1.M(1)$	$P_2.H(.56)$	$P_3.M(1)$	$P_4.mh(.32)$ $P_4.h(.67)$	$P_5.h(.70)$	$P_6.mh(.32)$ $P_6.h(.67)$
S_5	$P_1.H(.56)$	$P_2.M(1)$	$P_3.mh(.32)$ $P_3.h(.67)$	$P_4.h(.70)$	$P_5.mh(.32)$ $P_5.h(.67)$	$P_6.MH(.32)$ $P_6.H(.67)$
S_6	$P_1.M(1)$	$P_2.mh(.32)$ $P_2.h(.67)$	$P_3.h(.70)$	$P_4.mh(.32)$ $P_4.h(.67)$	$P_5.MH(.32)$ $P_5.H(.67)$	$P_6.H(.70)$
S_7	$P_1.mh(.32)$ $P_1.h(.67)$	$P_2.h(.70)$	$P_3.mh(.32)$ $P_3.h(.67)$	$P_4.MH(.32)$ $P_4.H(.67)$	$P_5.H(.70)$	$P_6.MH(.32)$ $P_6.H(.67)$

This step is repeated for the other angles and subsequences, with the results shown in Table 4.

STEP 4: The scalar cardinality of each fuzzy term is calculated as its count value. Take the fuzzy term $P_1.h$ as an example. Its scalar cardinality = (0 + 0.92 + 0.67 + 0 + 0 + 0 + 0.67) = 2.26. This step is repeated for the other fuzzy terms.

STEP 5: All the fuzzy items are collected as the candidate 1-pattern-sets.

STEP 6: For each fuzzy candidate 1-pattern, its support is checked against the predefined minimum support value α. Assume in this example, α is set at 0.075. Since the support values of all the candidate 1-patterns are larger than 0.075, these patterns are thus put in L_1 (Table 5).

STEP 7: Since L_1 is not null, the next step is done.

STEP 8: Set $r = 1$, where r is the number of fuzzy terms in the current pattern-sets to be processed.

STEP 9: In this step, the candidate set C_{r+1} is generated from L_r. C_2 is thus first generated from L_1. In this example, totally 19 candidate

Table 5. The set of large 1-pattern-sets L_1 for this example

Itemset	Count	Itemset	Count	Itemset	Count
$P_1.h$	2.26	$P_3.M$	2.00	$P_5.h$	2.04
$P_1.M$	2.00	$P_3.mh$	0.96	$P_5.M$	1.00
$P_1.m$	1.00	$P_4.M$	2.00	$P_6.H$	2.04
$P_2.h$	2.96	$P_4.H$	1.23	$P_6.h$	2.04
$P_2.M$	2.00	$P_4.h$	2.04	$P_6.M$	1.00
$P_3.h$	2.71	$P_5.H$	1.93		

Table 6. The membership values for $P_1.H \cap P_2.H$

s_p	$P_1.h$	$P_2.h$	$P_1.h \cap P_2.h$
1	0	0.92	0.0
2	0.92	0.67	0.67
3	0.67	0	0.0
4	0	0	0.0
5	0	0	0.0
6	0	0.67	0.0
7	0.67	0.70	0.67

2-pattern-sets are generated. Note that no two fuzzy terms with the same P_i are put in a candidate 2-pattern-set.

STEP 10: The following substeps are done for each newly formed candidate pattern-set.

STEP 10.1: The fuzzy membership value of each candidate pattern-set in each subsequence is calculated. Here, assume the minimum operator is used for the intersection. Take ($P_1.h$, $P_2.h$) as an example. The derived membership value for this candidate 2-pattern-set in s_2 is calculated as: $\min(0.92, 0.67) = 0.58$. The results for the other subsequences are shown in Table 6.

STEP 10.2: The scalar cardinality (count) of each candidate 2-pattern-set in the subsequences is then calculated.

STEP 10.3: The supports of the above candidate pattern-sets are then calculated and compared with the predefined minimum support 0.075. In this example, 18 pattern-sets satisfy this condition. They are thus kept in L_2 (Table 7).

STEP 11: Since L_2 is not null in the example, $r = r + 1 = 2$. Steps 8–10 are then repeated to find L_3 and others. In this example, the other fuzzy large pattern-sets found are shown in Table 8.

STEP 12: The large patterns are shifted to the ones with the first data-point subscript. For example, the three patterns ($P_2.h$, $P_3.h$, $P_4.h$), ($P_3.h$, $P_4.h$, $P_5.h$) and ($P_4.h$, $P_5.h$, $P_6.h$) in L_3 are shifted into ($P_1.h$, $P_2.h$, $P_3.h$). The other patterns are also checked for shifting in the same way.

STEP 13: Redundant patterns are removed. For example, the four large patterns, ($P_1.h$, $P_2.h$, $P_3.h$), ($P_1.h$, $P_2.h$, $P_3.h$), ($P_1.h$, $P_2.h$, $P_3.h$) and ($P_1.h$, $P_2.h$, $P_3.h$), are the same and only one of them is kept. The final results are shown in Table 9.

Table 7. The pattern-sets and their counts in L_2

Itemset	Count	Itemset	Count
$(P_1.h, P_2.h)$	1.34	$(P_3.M, P_4.h)$	0.67
$(P_1.h, P_2.M)$	0.67	$(P_4.M, P_5.h)$	0.67
$(P_1.M, P_2.h)$	0.67	$(P_4.H, P_5.H)$	0.67
$(P_1.m, P_2.h)$	0.92	$(P_4.h, P_5.H)$	0.67
$(P_2.h, P_3.h)$	2.01	$(P_4.h, P_5.h)$	1.34
$(P_2.h, P_3.M)$	0.67	$(P_5.H, P_6.H)$	1.34
$(P_2.M, P_3.h)$	0.67	$(P_5.h, P_6.H)$	0.67
$(P_3.h, P_4.M)$	0.67	$(P_5.h, P_6.h)$	1.34
$(P_3.h, P_4.H)$	0.67	$(P_5.M, P_6.h)$	0.67
$(P_3.h, P_4.h)$	1.34		
$(P_3.h, P_4.h)$	1.34		

Table 8. The other fuzzy large pattern-sets

L_i	Pattern-set
L_3	$(P_1.h, P_2.h, P_3.h)$, $(P_1.h, P_2.h, P_3.M)$, $(P_1.M, P_2.h, P_3.h)$, $(P_1.m, P_2.h, P_3.h)$, $(P_2.h, P_3.h, P_4.M)$, $(P_2.h, P_3.h, P_4.H)$, $(P_2.h, P_3.h, P_4.h)$, $(P_2.M, P_3.h, P_4.h)$, $(P_3.h, P_4.H, P_5.H)$, $(P_3.h, P_4.h, P_5.H)$, $(P_3.h, P_4.h, P_5.h)$, $(P_3.M, P_4.h, P_5.h)$, $(P_4.M, P_5.h, P_6.h)$, $(P_4.H, P_5.H, P_6.H)$, $(P_4.h, P_5.H, P_6.H)$, $(P_4.h, P_5.h, P_6.H)$, $(P_4.h, P_5.h, P_6.h)$
L_4	$(P_1.h, P_2.h, P_3.h, P_4.H)$, $(P_1.M, P_2.h, P_3.h, P_4.h)$, $(P_1.m, P_2.h, P_3.h, P_4.M)$, $(P_2.h, P_3.h, P_4.H, P_5.H)$, $(P_2.h, P_3.h, P_4.h, P_5.H)$, $(P_2.M, P_3.h, P_4.h, P_5.h)$, $(P_3.h, P_4.H, P_5.H, P_6.H)$, $(P_3.h, P_4.h, P_5.H, P_6.H)$, $(P_3.h, P_4.h, P_5.h, P_6.H)$, $(P_3.M, P_4.h, P_5.h, P_6.h)$
L_5	$(P_1.h, P_2.h, P_3.h, P_4.H, P_5.H)$, $(P_1.M, P_2.h, P_3.h, P_4.h, P_5.H)$, $(P_2.h, P_3.h, P_4.H, P_5.H, P_6.H)$, $(P_2.h, P_3.h, P_4.h, P_5.H, P_6.H)$, $(P_2.M, P_3.h, P_4.h, P_5.h, P_6.H)$
L_6	$(P_1.h, P_2.h, P_3.h, P_4.H, P_5.H, P_6.H)$, $(P_1.M, P_2.h, P_3.h, P_4.h, P_5.H, P_6.H)$

Table 9. The final large patterns after Step 13

L_i	Pattern-set
L_1	(H), (h), (mh), (M), (m)
L_2	(h, H), (h, h), (H, H), (h, M), (m, h), (M, h)
L_3	(H, H, H), (m, h, h), (h, h, H), (h, h, h), (h, H, H), (h, h, M), (M, h, h)
L_4	(M, h, h, h), (m, h, h, M), (h, H, H, H) (h, h, h, H), (h, h, H, H)
L_5	(h, h, h, H, H), (M, h, h, h, H), (h, h, H, H, H)
L_6	(h, h, h, H, H, H), (M, h, h, h, H, H)

STEP 14: The maximally fuzzy patterns generated from Step 13 are output as fuzzy linguistic trends that are (mh), (m, h, h, M), (h, h, h, H, H, H) and (M, h, h, h, H, H).

4 Experimental Results

In this section, the experiments made to show the performance of the proposed method are described. They were implemented in Java at a personal computer with Intel Pentium IV 3.20 GHz and 512 MB RAM. The dataset used in the experiments is a set of synthetic control-chart time series from *The UCI KDD Archive* [5]. The dataset contains 600 examples of control charts synthetically generated. The six classes are *normal*, *cyclic*, *increasing trend*, *decreasing trend*, *upward shift*, and *downward shift*. Each time series has sixty data points. One time series of each class was selected to make the following experiments.

Experiments were first made to show the relationship between numbers of linguistic trends and minimum support values. The sliding-window size was set at ten and the number of membership functions for angles is ten, with five for both positive and negative angles. Results for the class of *decreasing trend* are shown in Fig. 2.

From Fig. 2, it is easily seen that the number of linguistic trends decreased along with the increase of the length of large patterns except for L_2 and L_3. Finding L_2 and L_3 was the main effort in the mining process, which was consistent with previous study [6, 7, 10].

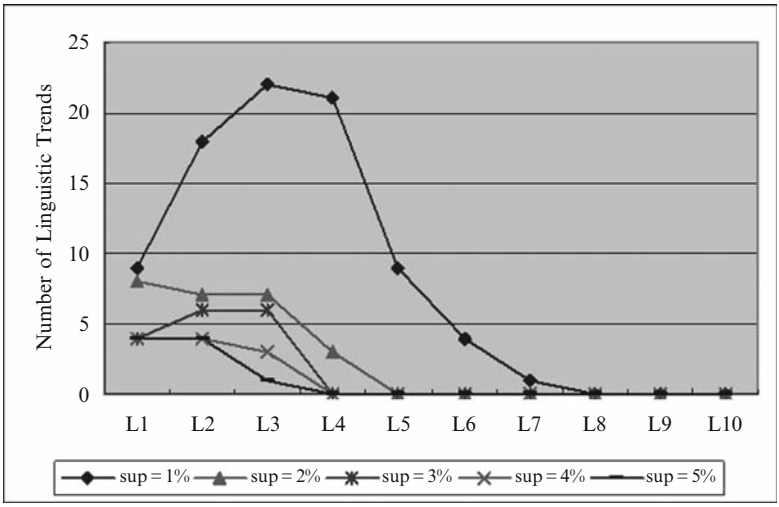


Fig. 2. The relationship between numbers of linguistic trends and min. sup

Table 10. The final large patterns aster Step 13

L_i	Fuzzy linguistic trends
L_1	(MH), (M), (LM), (h), (mh), (m), (lm), (a), (H)
L_2	(H, MH), (MH, H), (h, h), (lm, h), (H, h), (H, mh), (H, lm), (h, H), (H, H)
L_3	(MH, h, h), (H, h, h), (H, lm, h), (h, H, h), (H, H, h), (h, h, H), (H, h, H), (h, H, H), (h, H, MH)
L_4	(h, H, h, H), (MH, H, h, H), (h, H, h, h), (h, H, H, h), (H, h, H, H), (H, h, H, h)
L_5	(H, h, H, H, h)

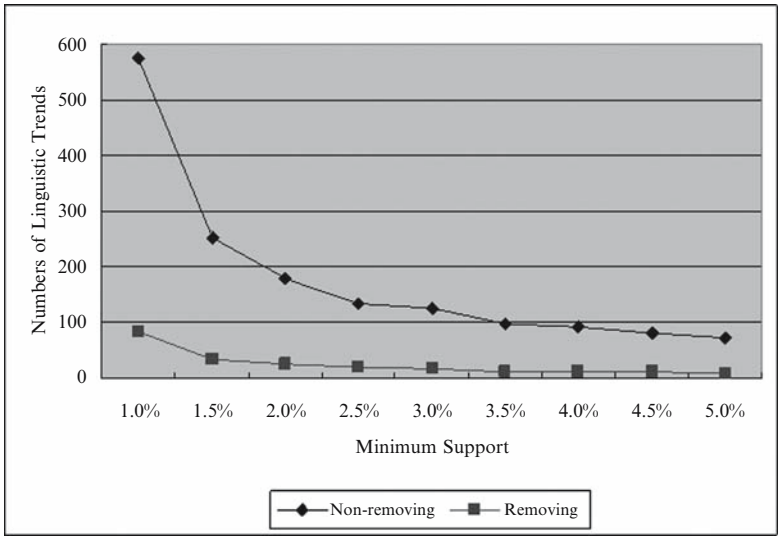


Fig. 3. The numbers of linguistic trends with and without Step 13

All the linguistic trends found for the class of decreasing trend with sliding-window size set at 10 and the minimum support set at 0.015 are listed in Table 10.

In Table 10, the pattern (H, h, H, H, h) is a linguistic trend with five fuzzy items in L_5. Lowercase and uppercase letters represent the positive and negative angular degrees upward and downward directions respectively. Most of the derived linguistic trends in Table 10 have the decreasing property, which is consistent with the class of *decreasing trend*.

The experiments were then made to compare the numbers of linguistic trends generated with and without Step 13 of removing redundant large patterns. The results are shown in Fig. 3.

From Fig. 3, it can be easily observed that removing redundant fuzzy large patterns during the mining process has its efficacy. Without this step, too many redundant linguistic trends may be generated and may make users confused.

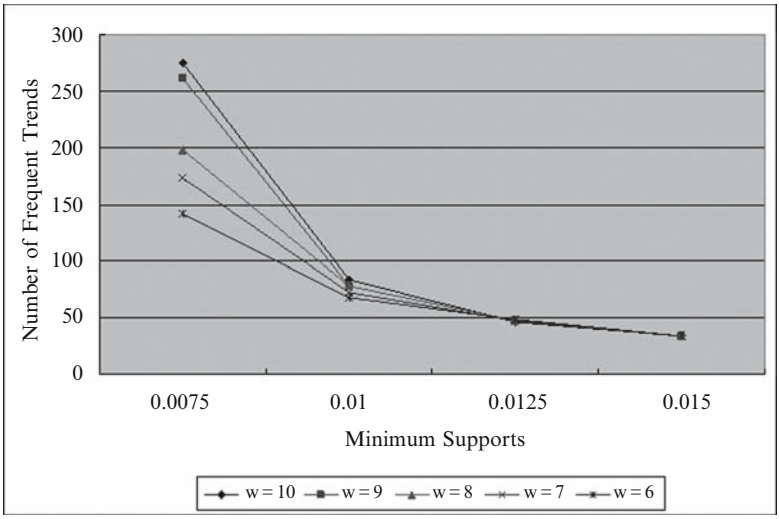


Fig. 4. The relationship between number of frequent trends and minimum supports along with different sliding-window sizes

At last, experiments were made to show the relationship between numbers of frequent trends and minimum supports along with different sliding-window sizes. The results were shown in Fig. 4.

As expected, when the sliding-window size increased, the numbers of frequent trends also increased.

5 Conclusion and Future Works

In this chapter, we have proposed a mining algorithm based on angles of adjacent points in a time series to find linguistic trends. The proposed approach first transforms data values into angles, and then uses a sliding window to generate continues subsequences from angular series. Several fuzzy sets for angles are predefined to represent semantic concepts understandable to human being. The a priori-like fuzzy mining algorithm is then used to generate linguistic trends. Appropriate post-processing is also performed to remove redundant patterns. Finally, experiments have been made for different parameter settings and experimental results shows that the proposed algorithm actually works.

Although the proposed method works for time series, it is just a beginning. There is still much work to be done in this field. In the future, we will continuously attempt to enhance the proposed algorithm for other applications, like telecommunication, bioinformatics, medical treatment and mobile computing. Besides, we will also continuously enhance the proposed algorithm for mining different kinds of knowledge.

Acknowledgement

This research was supported by the National Science Council of the Republic of China under contract NSC94-2213-E-390-005.

References

1. Au W H, Chan K C C (2004) Mining fuzzy rules for time series classification. The 2004 IEEE International Conference on Fuzzy Systems, Vol. 1, pp. 239–244
2. Agrawal R, Psaila G, Wimmers E L, Zait M (1995) Querying shapes of histories. The 21st International Conference on Very Large Databases, pp. 502–514
3. Agrawal R, Srikant R (1994) Fast algorithm for mining association rules. The International Conference on Very Large Databases, pp. 487–499
4. Chen S M, Hwang J R (2000) Temperature prediction using fuzzy time series. IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics, Vol. 30, No. 2, pp. 263–275
5. Hettich S, Bay S D (1999) The UCI KDD Archive, Department of Information and Computer Science, University of California, Irvine, CA
6. Hong T P, Kuo C S, Chi S C (1999) Mining association rules from quantitative data. Intelligent Data Analysis, Vol. 3, No. 5, pp. 363–376
7. Hong T P, Kuo C S, Chi S C (2001) Trade-off between time complexity and number of rules for fuzzy mining from quantitative data. International Journal of Uncertainty, Fuzziness and Knowledge-based Systems, Vol. 9, No. 5, pp. 587–604
8. Indyk P, Koudas N, Muthukrishnan S (2001) Identifying representative trends in massive time series data sets using sketches. The 26th International Conference on Very Large Data Bases, pp. 363–372
9. Keogh E, Chakrabarti K, Pazzani M, Mehrotra S (2001) Dimensionality reduction for fast similarity search in large time series databases. Journal of Knowledge and Information Systems, Vol. 3, No. 3, pp. 263–286
10. Lee Y C, Hong T P, Lin W Y (2004) Mining fuzzy association rules with multiple minimum supports using maximum constraints. Lecture Notes in Computer Science, Vol. 3214, pp. 1283–1290
11. Patel P, Keogh E, Lin J, Lonardi S (2002) Mining motifs in massive time series databases. The IEEE International Conference on Data Mining, pp. 370–377
12. Song Q, Chissom B S (1993) Fuzzy time series and its models. Fuzzy Sets System, Vol. 54, No. 3, pp. 269–277
13. Udechukwu A, Barker K, Alhajj R (2004) Discovering all frequent trends in time teries. The 2004 Winter International Symposium on Information and Communication Technologies, pp. 1–6
14. Watanabe N (2004) A fuzzy rule based time series model. The IEEE Annual Meeting on Fuzzy Information, Vol. 2, pp. 936–940
15. Yi B K, Faloutsos C (2000) Fast time sequence indexing for arbitrary Lp norms. The 26th International Conference on Very Large Databases, pp. 385–394

Latent Semantic Space for Web Clustering

I-Jen Chiang[1,3], Tsau Young ('T. Y.') Lin[2], Hsiang-Chun Tsai[3]
Jau-Min Wong[3], and Xiaohua Hu[4]

[1] Graduate Institute of Medical Informatics, Taipei Medical University, 205, Wu-Hsien Street, Taipei, Taiwan, ROC
ijchiang@tmu.edu.tw
[2] Department of Computer Science, San Jose State University, One Washington Square, San Jose, CA, USA
95192-0249 tylin@cs.sjsu.edu
[3] Graduate Institute of Biomedical Engineering, National Taiwan University, No.1, Sec. 1, Jen-Ai Road, Taipei, Taiwan, ROC
[4] College of Information Science and Technology, Drexel University, Philadelphia, PA 19104, USA
thu@cis.drexel.edu

Summary. To organize a huge amount of Web pages into topics, according to their relevance, is the efficient and effective method for information retrieval. Latent Semantic Space (LSS) naturally in the form on some geometric structure in *Combinatorial Topology* has been proposed for unstructured document clustering. Given a set of Web pages, the set of associations among frequently co-occurring terms in them forms naturally a CONCEPT, which is represented as a set of *connected components* of the simplicial complexes. Based on these concepts, Web pages can be clustered into meaningful categories.

1 Introduction

To adequately handle documents, a methodology to represent or to reveal their latent semantics are needed. To date, no universally accepted effective methodology has been discovered. In previous paper [15], we have pictured the latent semantics geometrically and call it the Latent Semantic Space (LSS) of the given set of documents. We take the key terms as vertices and visualize the term-associations(frequent co-occurring terms) as simplicial complex in LSS. Our thesis has been: a maximal connected component represents a CONCEPT in LSS of a collection of documents. However, in [15], we have not explored the full thesis, we consider only the PIMITIVE COMCEPTs of the highest dimension. Technically, we consider only the maximal connect components of the skeleton of the highest layer. In this paper, we explore the full notion

I-J. Chiang et al.: *Latent Semantic Space for Web Clustering*, Studies in Computational Intelligence (SCI) **118**, 61–77 (2008)
www.springerlink.com

of *PRINITIVE CONCEPT*s and the results are very encouraging.[1] These results can directly obtained from search engines. All the returned results are automatically clustered into different topics. The authoritative web pages in each topic are ranked based on how similar web pages belong to the topic. The experimental results indicate that we have an effective way to organize the large amount of return from a web query.

Internet is an information ocean. How to marshal large amount of returned web pages, paragraphs or sentences is the key issue. Roughly speaking, we decompose (triangulate, partition, granulate) LSS of documents (e.g., returned web pages or sentences) into *simplicial complex* in combinatorial topology [23], which could be viewed a special form of hypergraphs. However, we should note that the notion of simplicial complexes is actually predated that of hypergraphs about half a century, even though the latter notion is more familiar to modern computer scientists.

Let us recall some examples to illustrate the main intuition. The association that consists of "wall" and "street" denotes some financial notions that have meaning beyond the two nodes, "wall" and "street". This is similar to the notion of open segment (v_0, v_1)) that represents one dimensional geometric object, 1-simplex, that carries information beyond the two end points. In general, an r-association represents some semantic generated by a set of r keywords, may have more semantics or even have nothing to do with the individual keywords. A mathematical structure that reflects such phenomena is the notion of simplicial complex in combinatorial topology; see Sect. 3.

The thesis of this paper is that the simplicial complex of term-associations reflects the structure of the concepts in LSS of the documents. Based on such conceptual structure, the documents (returned pages, paragraph, or sentences) can be effectively clustered.

2 Key Terms and TDITF

The notion of association rules was introduced by Agrawal et al. [1] and has been demonstrated to be useful in several domains [4, 5], such as retail sales transaction database. In the theory two standard measures, called *support* and *confidence*, are often used. For documents the orders of keywords or directions of rules are not essential. Our focus will be on the support; a set of items that meets the support is often referred to as frequent itemsets; we will call them *associations* (undirected association rules) as to indicate the emphasis on their meaning more than the phenomena of frequency.

The frequency distribution of a word or phrase in a document collection is quite different from the item frequency distribution in a retail sales transaction database. In [14], we have shown that isomorphic relations have isomorphic

[1] The search engine's web site is at "http://ginni.bme.ntu.edu.tw/", which is a pentium IV personal computer.

associations. Documents are amorphous. An isomorphism essentially implies identity. So finding associations in a collection of textual documents is an important information and a challenging problem.

Traditional text mining generally consists of the analysis of a text document by extracting key words, phrases, concepts, etc. and representing in an intermediate form refined from the original text in that manner for further analysis with data mining techniques (e.g., to determine associations of concepts, key phrases, names, addresses, product names, etc.). Feldman and his colleagues [6,7,9] proposed the *KDT* and *FACT* system to discover association rules based on keywords labeling the documents, the background knowledge of keywords and relationships between them. This is not an effective approach, because a substantially large amount of background knowledge is required. Therefore, an automated approach that documents are labeled by the rules learned from labeled documents are adopted [13]. However, several association rules are constructed by a compound word (such as "Wall" and "Street" often co-occur) [19]. Feldman et al. [6, 8] further proposed term extraction modules to generate association rules by selected key words. Nevertheless, a system without the needs of human labeling is desirable. Holt and Chung [11] addressed Multipass-a priori and Multipass-DHP algorithms to efficiently find association rules in text by modified the a priori algorithm [2] and the DHP algorithm [18] respectively. However, these methods did not consider about the word distribution in a document, that is, identify the importance of a word in a document.

According to the trivial definition of distance measure in this space, no matter what kind of a method is, some common words are more frequent in a document than other words. Simple frequency of the occurrence of words is not adequate, as some documents are larger than others. Furthermore, some words may occur frequently across documents. In most cases, words appeared in a few documents tend to most "important." Techniques such as TFIDF [21] have been proposed directly to deal with some of these problems. The TFIDF value is the weight of term in each document. While considering relevant documents to a search query, if the TFIDF value of a term is large, then it will pull more weight than terms with lesser TFIDF values.

A general framework for text mining consists of two phases. The first phase, *feature extraction*, is to extract key terms from a collection of "indexed" documents; as a second step various methods such as association rules algorithms may be applied to determine relations between features.

While performing association analyses on a collection of documents, all documents should be indexed and stored in an intermediate form. Document indexing is originated from the task of assigning terms to documents for retrieval or extraction purposes. In early approach, an indexing model was developed based on the assumption that a document should be assigned those terms that are used by queries to retrieve the relevant document [10, 16]. The weighted indexing is the weighting of the index terms with respect to the document with this model given a theoretical justification in terms

of probabilities. The most simple and sophisticated weighted schema which is most common used in information retrieval or information extraction is TFIDF indexing, i.e., tf × idf indexing [20, 21], where tf denotes term frequency that appears in the document and idf denotes inverse document frequency where document frequency is the number of documents which contain the term. It takes effect on the commonly used word a relatively small tf × idf value. Moffat and Zobel [17] pointed out that tf × idf function demonstrates: (1) rare terms are no less important than frequent terms in according to their idf values; (2) multiple appearances of a term in a document are no less important than single appearances in according to their tf values. The tf × idf implies the significance of a term in a document, which can be defined as follows.

We observed that the direction of key terms (including compound words) is irrelevant information for the purpose of document clustering. So we ignore the *confidence* and consider only the *support*. In other words, we consider the structure of the *undirected* associations of key terms; we believe the set of key terms that co-occur reflects the essential information, the rule directions of the key terms are inessential, at least in the present stage of investigation. Let t_A and t_B be two terms. The *support* is defined for a collection of documents as follows.

Definition 1. *The significance of undirected associations of term t_A and term t_B in a collection is:*

$$\text{significance}(\mathrm{t_A}, \mathrm{t_B}, \mathrm{T_r}) = \frac{1}{|\mathrm{T_r}|} \sum_{\mathrm{i}=0}^{|\mathrm{T_r}|} \text{significance}(\mathrm{t_A}, \mathrm{t_B}, \mathrm{d_i})$$

where

$$\text{significance}(\mathrm{t_A}, \mathrm{t_B}, \mathrm{d_i}) = \mathrm{tf}(\mathrm{t_A}, \mathrm{t_B}, \mathrm{d_i}) \log \frac{|\mathrm{T_r}|}{|\mathrm{T_r}(\mathrm{t_A}, \mathrm{t_B})|},$$

$|T_r(t_A, t_B)|$ defines number of documents contained both term t_A and term t_B, and $|T_r|$ denotes the number of Web pages in a collection.

The term frequency $\mathrm{tf}(\mathrm{t_A}, \mathrm{t_B}, \mathrm{d_i})$ of both term t_A and t_B can be calculated as follows.

Definition 2.

$$\mathrm{tf}(\mathrm{t_A}, \mathrm{t_B}, \mathrm{d_j}) = \begin{cases} 1 + \log(\min\{N(t_A, d_j), N(t_B, d_j)\}) \\ \textit{if } N(t_A, d_j) > 0 \textit{ and } N(t_B, d_j) > 0 \\ \\ 0 \\ \textit{otherwise.} \end{cases}$$

A minimal threshold θ is imposed to filter out the terms that their significance values are small. It helps us to eliminate the most common terms in a collection and the nonspecific terms in a document.

Let t_A and t_B be two terms. The *support* defined in the document collection is as follows.

Definition 3. Support *denotes to be the significance of associations of term* t_A *and term* t_B *in a collection, that is,*

$$\mathrm{Support}(\mathrm{t_A}, \mathrm{t_B}) = \mathrm{significance}(\mathrm{t_A}, \mathrm{t_B}, \mathrm{T_r})$$

It is obvious that the support evaluated by *tfidf* satisfies the a priori condition.

3 Geometric Theory of Latten Semantic Space

The goal of this section is to model the internal semantic of a collection of documents using a set of geometric and topologic notions, called simplicial complex that is a special form of hypergraphs [15].

3.1 Simplicial Complex

Let us introduce and define some basic notions in combinatorial topology. The central notion is n-simplex.

Definition 4. *A* n*-simplex is a set of independent abstract vertices* $[v_0, \ldots, v_{n+1}]$. *A* r*-face of a* n*-simplex* $[v_0, \ldots, v_{n+1}]$ *is a* r*-simplex* $[v_{j_0}, \ldots, v_{j_{r+1}}]$ *whose vertices are a subset of* $\{ v_0, \ldots, v_{n+1} \}$ *with cardinality* $r+1$.

Geometrically 0-simplex is a vertex; 1-simplex is an open segment (v_0, v_1) that does not include its end points; 2-simplex is an open triangle (v_0, v_1, v_2) that does not include its edges and vertices; 3-simplex is an open tetrahedron (v_0, v_1, v_2, v_3) that does not includes all the boundaries. For each simplex, all its proper faces (boundaries) are not included. An n-simplex is the high dimensional analogy of those low dimensional simplexes (segment, triangle, and tetrahedron)in n-space. Geometrically, an n-simplex uniquely determines a set of $n+1$ linearly independent vertices, and vice versa. An n-simplex is the smallest convex set in a Euclidean space R^n that contains $n+1$ points v_0 …, v_n that do not lie in a hyperplane of dimension less than n. For example, there is the standard n-simplex

$$\delta^n = \{(t_0, t_1, \ldots, t_{n+1}) \in R^{n+1} \mid \sum_i t_i = 1, t_i \geq 0\}$$

The convex hull of any m vertices of the n-simplex is called an m-face. The 0-faces are the vertices, the 1-faces are the edges, 2-faces are the triangles, and the single n-face is the whole n-simplex itself. Formally,

Definition 5. *A simplicial complex C is a finite set of simplexes that satisfies the following two conditions:*

- *Any set consisting of one vertex is a simplex.*
- *Any face of a simplex from a complex is also in this complex.*

The vertices of the complex $v_0, v_1, \ldots, v_n$ is the union of all vertices of those simplexes ([23], pp. 108).

If the maximal dimension of the constituting simplexes is n then the complex is called n-complex.

Note that, any set of $n+1$ objects can be viewed as a set of abstract vertices, to stress this abstractness, some times we refer to such a simplex a combinatorial n-simplex. The corresponding notion of combinatorial n-complex can be defined by (combinatorial) r-simplexes. Now, by regarding the key terms, as defined by high TFIDT values, as abstract vertices, an association of $n+1$ key terms, called $n+1$-association, is a combinatorial n-simplex: A 2-association is an open 1-simplex. An open 1-simplex ("wall", "street") represents a financial notion that includes some semantics that is well beyond the two vertices, "wall" and "street." A $(n+1)$-association is a combinatorial n-simplex of keywords that often carries some deep semantics that are well beyond the "union" of its vertices, or faces individually.

We need much more precise notions. A (n, r)-skeleton (denoted by S_r^n) of n-complex is a n-complex, in which all k-simplexes($k \leq r$) have been removed. Two simplexes in a complex are said to be *directly connected* if the intersection of them is a nonempty face. Two simplexes in a complex are said to be *connected* if there is a finite sequence of directly connected simplexes connecting them. For any nonempty two simplexes A, B are said to be *r-connected* if there exits a sequence of k-simplexes $A = S_0, S_1, \ldots, S_m = B$ such that S_j and S_{j+1} has an h-common face for $j = 0, 1, 2, \ldots, m-1$; where $r \leq h \leq k \leq n$.

The maximal r-connected subcomplex is called a *r-connected component*. Note that a r-connected component implies there does not exist any r-connected component that is the superset of it. A maximal r-connected sub-complexes of n-complex is called r-connected component. A maximal r-connected component of n-complex is called connected component, if $r = 0$.

3.2 The Geometric Structure of Latent Semantic Space

From a collection of documents, a complex of term-associations can be generated. In this section, we will first examine the intuitive meaning of such a complex. First let us recall the notion of hypergraph:

Definition 6. *A* hypergraph $G = (V, E)$ *contains two distinct sets where V is a finite set of abstract vertices, and $E = \{e_1, e_2, \ldots, e_m\}$ is a nonempty family of subsets from V, in which each subset is called a* hyperedge.

It is obvious that a simplicial complex is a hypergraph: the set of vertices is V, and the set of simplexes is E. From the definition a simplicial complex is a very special kind of hypergraphs. However, there intrinsic and fundamental differences; Hypergraph is a combinatorial subject, while simplicial complex is a geometric concept. We will use the geometry to represent the concepts or thoughts in the collection of documents.

Note that the a priori conditions on term-associations meet the conditions of the simplicial complex: an 1-association is the 0-simplex, and a "subset" of an association is an association of shorter lengths. So the notion of simplicial complex is a natural view of term-associations. We will *take this view.*

In our application each simplex represents a certain concept. The 0-simplex (Network) might represents a CONCEPT. But it can be combined into many different concepts. For example, the following 1-simplexes (Computer, Network), (Traffic, Network), (Neural, Network), (Communication, Network), and etc., express further and richer semantic than their individual 0-simplexes. Of course, the 1-simplex (Neural, Network) is not conspicuous than the 2-simplexes (Artificial Neural Network) and (Biology, Neural, Network).

A collection of documents may carry a set of distinct CONCEPTS. Each concept, we believe, is carried by a connected component of the complex of term-associations. Here is our belief and our thesis:

- An IDEA (in the forms of complex of term-associations) may consist many CONCEPTS (in the form of connected components) that consists of PRIMITIVE CONCEPTS (in the form of simplexes). The maximal simplexes of highest dimension is called MAXMIAL PRIMITIVE CONCEPT. A simplex is said to be a maximal if no other simplex in the complex is a superset of it. Intuitively the geometric dimension represents the degree of preciseness or depth of the latent semantics that are represented by term-associations.

Example 1 *In Fig. 1, we have an idea that consist of 12 terms that organized in the forms of* 3*-complex, denoted by* S^3*. Simplex*(a, b, c, d) *and Simplex*(w, x, y, z) *are two maximal simplexes of* 3*, the highest dimension. Let us consider* S_1^3*. It is the leftover from the removal of all* 0*-simplexes from* S^3*:*

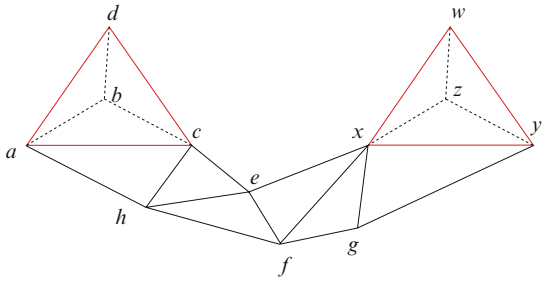


Fig. 1. A complex with 12 vertexes

- Simplex(a, b, c, d) *and its ten faces:*
  - Simplex(a, b, c)
  - Simplex(a, b, d)
  - Simplex(a, c, d)
  - Simplex(b, c, d)
  - Simplex(a, b)
  - Simplex(a, c)
  - Simplex(b, c)
  - Simplex(a, d)
  - Simplex(b, d)
  - Simplex(c, d)
- Simplex(a, c, h) *and its three faces:*
  - Simplex(a, c)
  - Simplex(a, h)
  - Simplex(c, h)
- Simplex(c, h, e) *and its three faces:*
  - Simplex(c, h)
  - Simplex(h, e)
  - Simplex(c, e)
- Simplex(e, h, f) *and its three faces:*
  - Simplex(e, h)
  - Simplex(h, f)
  - Simplex(e, f)
- Simplex(e, f, x) *and its three faces:*
  - Simplex(e, f)
  - Simplex(e, x)
  - Simplex(f, x)
- Simplex(f, g, x) *and its three faces:*
  - Simplex(f, g)
  - Simplex(g, x)
  - Simplex(f, x)
- Simplex(g, x, y) *and its three faces:*
  - Simplex(g, x)
  - Simplex(g, y)
  - Simplex(x, y)
- Simplex(w, x, y, z) *and its ten faces:*
  - Simplex(w, x, y)
  - Simplex(w, x, z)
  - Simplex(w, y, z)
  - Simplex(x, y, z)
  - Simplex(w, x)
  - Simplex(w, y)
  - Simplex(w, z)
  - Simplex(x, y)
  - Simplex(x, z)
  - Simplex(y, z)

Note that Simplex(a, c), Simplex(c, h), Simplex(h, e), Simplex(e, f), Simplex (f, x), Simplex(g, x), *and* Simplex(x, y) *all have common faces. So they generate a connected path from* Simplex(a, b, c, d) *to* Simplex(w, x, y, z), *and subpaths. Therefore the* S_1^3 *complex is connected. This assertion also implies that* S^3 *is connected. Hence the IDEA consists of a single CONCEPT(please note the technical meaning of the IDEA and CONCEPT given above). Next, let us consider the* $(3, 2)$*-skeleton* S_2^3*, by removing all* 0*-simplexes and* 1*-simplexes from* S^3*:*

- Simplex(a, b, c, d) *and its four faces:*
  - Simplex(a, b, c)
  - Simplex(a, b, d)
  - Simplex(a, c, d)
  - Simplex(b, c, d)
- Simplex(a, c, h)
- Simplex(c, h, e)
- Simplex(e, h, f)
- Simplex(e, f, x)
- Simplex(f, g, x)
- Simplex(g, x, y)
- Simplex(w, x, y, z) *and its four faces:*
  - Simplex(w, x, y)
  - Simplex(w, x, z)
  - Simplex(w, y, z)
  - Simplex(x, y, z)

There are no common faces between any two simplexes, so S_2^3 *has eight connected components, or eight CONCEPTS. For* S_3^3*, it consists of two nonconnected* 3*-simplexes or two MAXIMAL PRIMITIVE CONCEPTS.*

A complex, connected component or simplex of a skeleton represent a more technically refined IDEA, CONCEPT or PRIMITVE CONCEPT. If a maximal connected component of a skeleton contains only one simplex, this component is said to organize a primitive concept.

3.3 Layered Views

Based on the dimension hierarchies of primitive CONCEPTS, we can define the notion of layered clustering As seen in Example 1, connected components in S_k^n are contained in that of S_r^n, where $k \geq r$.

Example 2 *Figure 2 is* 2*-complex composed of the term set* $V = \{t_A, t_B, t_C\}$ *in a collection of documents. It is a close* 2*-simplex; we recall here that a closed simplex is a complex that consists of one simplex and all its faces. In the skeleton* S_1^2*, all* 0*-simplexes are ignored, i.e., the terms depicted in dash lines. The simplex set* $S = \{\text{Simplex}_1^2, \text{Simplex}_2^1, \text{Simplex}_3^1, \text{Simplex}_4^1\}$ *is the*

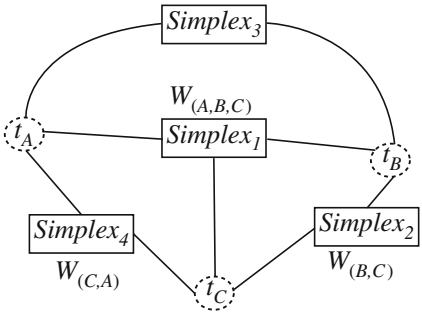


Fig. 2. This figure illustrates the skeleton S_1^3 of Example 2. It is composed from three key terms $\{t_A, t_B, t_C\}$ of a collection of documents, where each simplex is identified by its tfidf value and all 0-simplexes have been removed (the nodes are drawn by using dash circles). Note that Simplex_1 has dimension 2, we draw its incidences with three vertices, but skip the incidences with three 1-simplexes

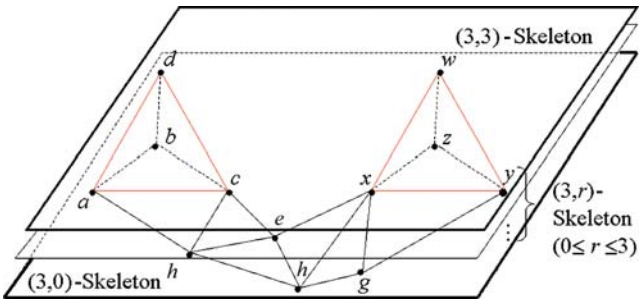


Fig. 3. This figure illustrates the layer structures of Example 1. The *top* layer is skeleton $(3, 3)$-Skeleton that has two distinct CONCEPTS Simplex(a, b, c, d) and Simplex(w, x, y, z). The *middle* layer $(3, 2)$-Skeleton has 8 CONCEPTS; it is not illustrate here. The layer $(3, 1)$-Skeleton is skipped. The *bottom* layer $(3, 0$-Skeleton) contains only one connected component; it is shown in the figure

closed 2*-simplex that consists of one* 2*-simplex and three 1-faces,* Simplex_2^1, Simplex_3^1 *and* Simplex_4^1 *(*0*-faces are ignored). These* r*-simplexes (*$0 \leq r \leq 2$*) represents frequent itemsets (term-associations) from* V*, where* $W=\{w_{A,B}, w_{C,A}, w_{B,C}, w_{A,B,C}\}$ *denote their corresponding supports. The lines connecting* Simplex_1 *and three vertices represent the incidences of* 2*-simplex and* 0*-simplex; the incidences with* 1*-simplexes are not shown to avoid overcrowding the figure.*

According to Example 1, it is obvious that simplexes within the higher level skeleton S_r^n is contained in the lower level skeleton S_k^n within the same n-complex, $r \geq k$. Figure 3 shows the hierarchy, each skeleton is represented as a layer. For the purpose of simplicity, we skip the middle layer, namely, S_r^n, $0 \leq r < 3$, are not shown.

By considering different skeletons, we can draw distinct layer of CONCEPTS:

1. In full complex $S = S_0^n$, this example only has one CONCEPT (one connected component).
2. In S_1^n, this complex still has only one CONCEPT.
3. In S_2^n, this complex has eight CONCEPTS.
4. in S_3^n, this complex has two CONCEPTS; they are two MAXIMAL PRIMITVE CONCEPTS.

For each choice, say S_2^n, we have, in this case, eight CONCEPTS to label the documents (or clustered the documents). A document is labeled $CONCEPT_k$, if the document has high TFITD values on the term-associations that defines $CONCEPT_k$. By consider different cases, we have layered clusters. In fact, we even could consider a very coarse clustering that is, we consider only the MAXIMAL PRIMITVE CONCEPTS; this is the case of S_3^n. For the purpose of illustrating the methodology, we have focused on this "oversimplified" one.

In general, the simplexes at the lower layers could have common faces between them. Therefore, to use all layers of CONCEPTS at the same time will produce vague discrimination as shown in Fig. 4, in which an overlapped CONCEPTS induced by (lower dimensional) common faces could exist. As seen in the skeleton S_1^3, the maximal connected components generated from simplex Simplex(a, b, c, d) and simplex Simplex(a, c, h) have a common face Simplex(a, c) that makes some documents not able to properly discriminated in accordance with the generated association rules from term a and term c, so are the other maximal connected components in the skeleton. Because of the intersection produced by such faces, a proper way is to ignore the lower the skeleton as much as application can tolerate.

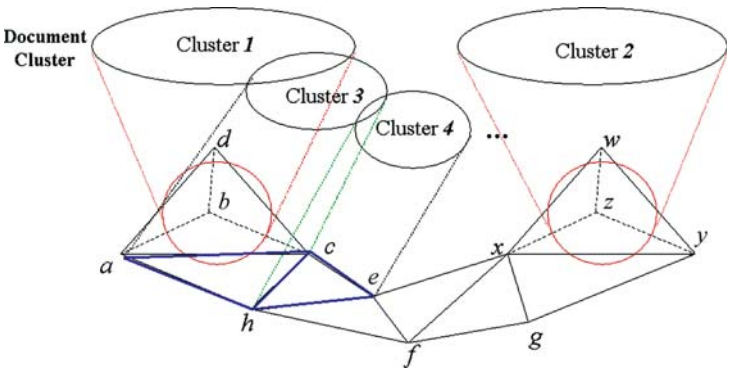


Fig. 4. Each cluster of documents is identified by a maximal connected component. Some clusters may overlap with other cluster because of the common face between them; this phenomenon is illustrated here. To handle such a situation properly, we need to ignore the lower dimensional simplexes. By so doing the overlapping will disappear (not shown)

4 Finding Connected Components

We can visualize that the latent semantic of a collection of documents is a space triangulated/partitioned/granulated by term-associations (simplexes). The space contains CONCEPTS, PRIMITIVE CONCEPTS. The algorithms for finding all CONCEPTS, i.e., maximal connected components will be introduced below.

4.1 Incidence Matrices

First, we need some geometric notations.

Definition 7. *Let $\mathcal{V}$ be the set of vertices of a simplicial complex i.e., 0-simplices, and let $\mathcal{E}$ be the set of all r-simplices, where $r \geq 0$. If* Simplex_A *is in $\mathcal{E}$, its support, denoted by $w_{(\text{Simplex}_\text{A})}$, is defined to be the* tfidf *of* Simplex_A.

The *incident matrix* and the *weighted incident matrix* of a complex can be defined as follows:

Definition 8. *The $n \times m$* incident matrix $A = (a_{ij})$ *associated to a complex is defined as*

$$a_{ij} = \begin{cases} 1 \text{ if } \text{Simplex}_\text{i} \text{ is a face of } \text{Simplex}_\text{j} \\ 0 \text{ otherwise.} \end{cases}$$

The corresponding weight incident matrix $A' = (a'_{ij})$ *is*

$$a'_{ij} = \begin{cases} w_{ij} \text{ if } \text{Simplex}_\text{i} \text{ is a face of } \text{Simplex}_\text{j} \\ 0 \quad \text{otherwise} \end{cases}$$

where the weight w_{ij} denotes the support *of a term-association.*

Example 3 *In Example 2, the 2-simplex is the set $\{t_A, t_B, t_C\}$, which is also the connected component that represents a CONCEPT. The incident matrix I and the weighted incident matrix I_W of the simplexes can be constructed. For clarity, we only illustrate the incidences between the key terms (0-simplexes) and term-associations (r-simplexes, r=1,2) as follows.*

$$I = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{pmatrix}.$$

$$I_W = \begin{pmatrix} w_{A,B,C} & 0 & w_{A,B} & w_{C,A} \\ w_{A,B,C} & w_{B,C} & w_{A,B} & 0 \\ w_{A,B,C} & w_{B,C} & 0 & w_{C,A} \end{pmatrix}.$$

Each row represents the incidence of a vertex with all r-simplexes. Each column corresponds to the incidence of a fixed simplex and all vertices.

4.2 Algorithm

As we already known, a r-simplex is a $(r+1)$-term-association (frequent $(r+1)$-itemset). Web pages can be clustered based on maximal simplexes of any dimension (CONCEPTS), i.e., associations. Note that Web pages clustered by CONCEPTS contains common lower dimensional faces (shorter associations, in particular 0-simplexes); this is consequence of a priori property. In this sense, the methodology provides a soft approach; we allow lower dimensional overlapped CONCEPTS exist within different clusters.

Since the intersection of connected components has lower dimensions. It is convenient for us to design an efficient algorithm for documents clustering in a skeleton by skeleton fashion. The algorithm for finding all maximal connected components based on a user query in a skeleton is listed as follows.

Require: $\mathcal{V} = \{t_1, t_2, \ldots, t_n\}$ be the vertex set of all reserved terms in a collection of documents.

Ensure: $\mathcal{H}$ is the hierarchy of connected components.

Let t_0 be the user query.
Let θ be a given minimal support.
Let $S_0 = \{t_0\}$ be the root of the hierarchy.
Let Support(S_0) be the *support* of associations of the terms in S_0.
$\mathcal{H} \Leftarrow S_0$
$i \Leftarrow 0$
while $S_i \neq \emptyset$ **do**
  while for all vertex $t_j \in V$ and $\notin S_i$ **do**
    Let $S_{(i+1)} \Leftarrow S_i \bigcup t_j$ if Support($S_{(i+1)}$) is bigger than θ.
    Add $S_{(i+1)}$ to be the child of S_i
  end while
  $i \Leftarrow (i+1)$
end while

Use our notation S_i is a skeleton of S_0^i. It is clear, one can get S_m^n for any n and m. A simplex will be constructed by including all those co-occurring terms whose support is bigger than or equal to a given minimal support θ. An external vertex will be added into a simplex if the produced support is no less than θ.

According to our algorithm, the simplex will be constructed through one term, that is, a user query. All the noun phrases in a Web pages returned from remote search engines will be selected for document clustering. Web pages can be decomposed into several categories based on the PRIMITIVE CONCEPTS. If a Web page contains a PRIMITIVE CONCEPT, it means that Web page highly equates to such concept, thereby, by the a priori property, all the sub-associations in the concept is also contained in this Web page. The Web page can be classified into the category identified with such a concept. A document often consists of more than one PRIMITIVE CONCEPTS context, in this case it can be classified into multicategories.

5 Experimental Results

LSS clustering organizes the returned Web pages hierarchically from a user query on-the-fly over two remote search engines (PubMed and GOOGLE). As we already known, GOOGLE provides a flat list of search result snippets and PubMed returns a summarized list of the abstracts of medical literatures associated with MeSH terms. In our experiments, all the PRIMITIVE CONCEPTs are bound a threshold, since the simplexes in the lower dimension is highly overlapped with several IDEA. In our system, the lower dimension the PRIMITIVE CONCEPTs are near to the root of hierarchy, and versus visa.

5.1 LSS System

The system is depicted in Fig. 5 that demonstrates the search results from PubMed for a search query "pain". A hierarchy of clusters is built on the return results. The abstract (unstructure) and MeSH terms (semi-structure) are in use to cluster them. The same term "pain" has been taken to retrieve information from GOOGLE. The returned result snippets are grouped into several clusters as shown in Fig. 6.

5.2 Results

The experimental evaluation of document clustering approaches usually measures their *effectiveness* rather than their *efficiency* [22], in the other word, the ability of an approach to make a *right* categorization. Entropy is involved

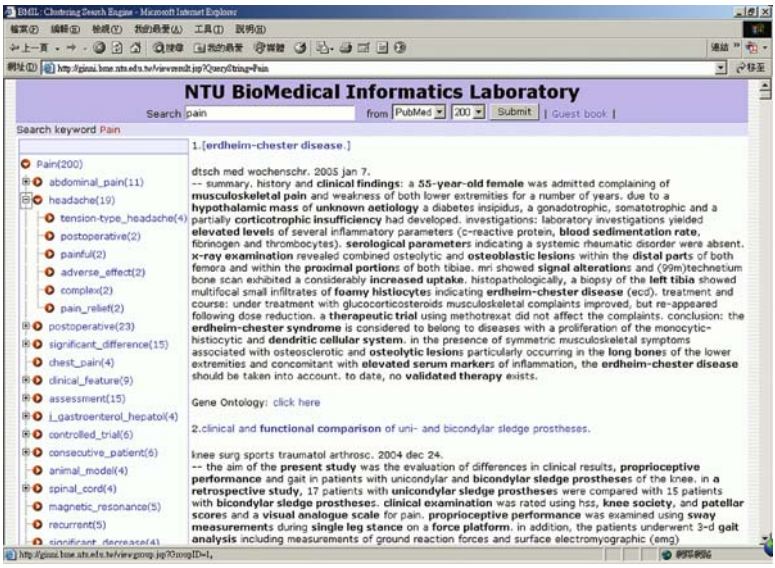


Fig. 5. LSS System oversearch engine PubMed

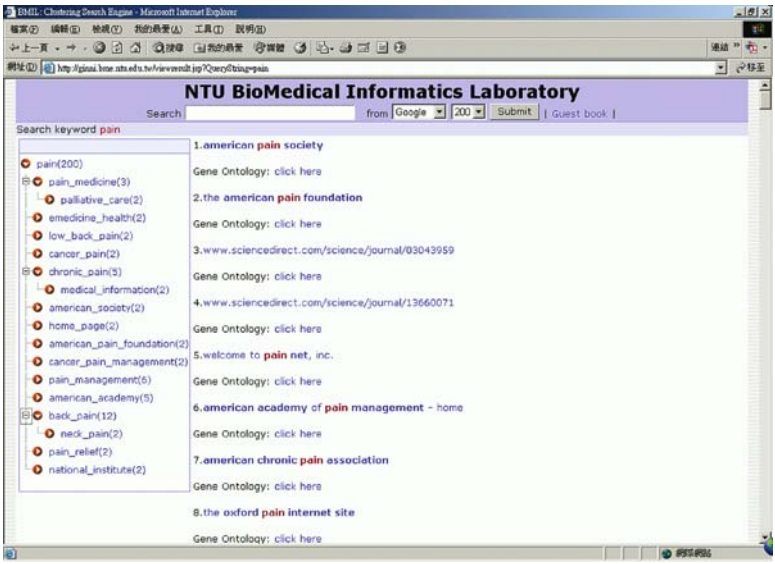


Fig. 6. LSS System oversearch engine Google

to evaluate the clustering performance [3] based on the human expert's decisions. More than one hundred queries related to medicine have been submitted from our system to clustering the returned results from PubMed and GOOGLE spectively. More than two hundred thousand Web pages or snippets have been returned. In general, the average entropy is around 0.14 ± 0.06 for PubMed and 0.27 ± 0.08 or so for GOOGLE. Because PubMed has defined meta-date for each medical literature by human experts. If without using these meta-data, the average entropy will become 0.21 ± 0.09. According to it, we can conclude courageously that the CONCEPTs organized by LSS can nearly make a precisely semantic concept clustering for Web pages.

6 Conclusion

Polysemy, *phrases* and *term dependency* are the limitations of search technology [12]. A single term is not able to identify a latent concept in a document, for instance, the term "Network" associated with the term "Computer", "Traffic", or "Neural" denotes different concepts. To discriminate term associations no doubt is concrete way to distinguish one category from the others. A group of solid term associations can clearly identify a concept. The term-associations (frequently co-occurring terms) of a given collection of Web pages, form a simplicial complex. The complex can be decomposed into connected components at various levels (in various level of skeletons). We believe each such a connected component properly identify a concept in a collection of Web pages.

We can effectively discover such a maximal simplexes and use them to cluster the collection of Web pages. Based on our web site and our experiments, we find that LSS is a very good way to organize the unstructured and semi-structure data into several semantic topics. It illustrates that geometric complexes are effective models for automatic Web pages clustering.

References

1. R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *Proceedings of the 1993 International Conference on Management of Data (SIGMOD 93)*, pages 207–216, May 1993
2. R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proceedings of the 20th VLDB Conference*, pages 487–499, 1994
3. D. Boley, M. Gini, R. Gross, E.-H. Han, K. Hastings, G. Karypis, V. Kumar, B. Mobasher, and J. Moore. Document categorization and query generation on the world wide web using webace. *Artificial Intelligence Review*, 13(5–6):365–391, 1999
4. S. Brin, R. Motwani, J. Ullman, and S. Tsur. Dynamic itemset counting and implication rules for market basket data. In *Proceedings of ACM SIGMOD International Conference on Management of Data*, pages 255–264, 1997
5. M. S. Chen, J. Han, and P. S. Yu. Data mining: An overview from a database perspective. *IEEE Transaction on Knowledge and Data Engineering*, 8(6):866–883, 1996
6. R. Feldman, Y. Aumann, A. Amir, W. Klósgen, and A. Zilberstien. Text mining at the term level. In *Proceedings of 3rd International Conference on Knowledge Discovery, KDD-97*, pages 167–172, Newport Beach, CA, 1998
7. R. Feldman, I. Dagan, and W. Klósgen. Efficient algorithms for mining and manipulating associations in texts. In *Cybernetics and Systems, The 13th European Meeting on Cybernetics and Research*, volume II, Vienna, Austria, April 1996
8. R. Feldman, M. Fresko, H. Hirsh, Y. Aumann, O. Liphstat, Y. Schler, and M. Rajman. Knowledge management: A text mining approach. In *Proceedings of 2nd International Conference on Practical Aspects of Knowledge Management*, pages 29–30, Basel, Switzerland, 1998
9. R. Feldman and H. Hirsh. Mining associations in text in the presence of background knowledge. In *Proceedings of 3rd International Conference on Knowledge Discovery*, pages 343–346, 1996
10. N. Fuhr and C. Buckley. A probabilistic learning approach for document indexing. *ACM Transactions on Information Systems*, 9(3):223–248, 1991
11. J. D. Holt and S. M. Chung. Efficient mining of association rules in text databases. In *Proceedings of CIKM*, Kansas City, MO, 1999
12. A. Joshi and Z. Jiang. Retriever: Improving web search engine results using clustering. In A. Gangopadhyay, editor, *Managing Business with Electronic Commerce: Issues and Trends*, chapter 4. World Scientific, New York, 2001
13. B. Lent, R. Agrawal, and R. Srikant. Discovering trends in text databases. In *Proceedings of 3rd International Conference on Knowledge Discovery, KDD-97*, pages 227–230, Newport Beach, CA, 1997

14. T. Y. Lin. Attribute (feature) completion – the theory of attributes from data mining prospect. In *Proceedings of 2002 IEEE International Conference on Data Mining (ICDM)*, pages 282–289, Maebashi, Japan, 2002
15. T. Y. Lin and I. J. Chiang. A simplicial complex, a hypergraph, structure in the latent semantic space of document clustering. *International Journal of Approximate Reasoning*, 40(1–2):55–80, 2005
16. M. E. Maron and J. K. Kuhns. On relevance, probabilistic indexing, and information retrieval. *Journal of ACM*, 7:216–244, 1960
17. A. Moffat and J. Zobel. Compression and fast indexing for multi-gigabit text databases. *Australian Computing Journal*, 26(1):19, 1994
18. J. S. Park, M. S. Chen, and P. S. Yu. Using a hash-based method with transaction trimming for mining association rules. *IEEE Transaction on Knowledge and Data Engineering*, 9(5):813–825, 1997
19. M. Rajman and R. Besanon. Text mining: Natural language techniques and text mining applications. In *Proceedings of 7th IFIP* 2.6 *Working Conference on Database Semantics (DS-7)*, Leysin, Switzerland, 1997
20. G. Salton and C. Buckley. Term weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523, 1960
21. G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, New York, 1983
22. F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, pages 1–47, 2002
23. E. Spanier. *Algebric Topology*. McGraw-Hill, New York, 1966

A Logical Framework for Template Creation and Information Extraction

David Corney[1], Emma Byrne[2], Bernard Buxton[1], and David Jones[1]

[1] Department of Computer Science, University College London, Gower Street, London WC1E 6BT, UK
D.Corney@ucl.ac.uk, B.Buxton@cs.ucl.ac.uk, D.Jones@cs.ucl.ac.uk

[2] School of Primary Care and Population Sciences, University College London, Highgate Hill, London N19 5LW, UK
emma.byrne@ucl.ac.uk

Summary. Information extraction is the process of automatically identifying facts of interest from pieces of text, and so transforming free text into a structured database. Past work has often been successful but ad hoc, and in this paper we propose a more formal basis from which to discuss information extraction. We introduce a framework which will allow researchers to compare their methods as well as their results, and will help to reveal new insights into information extraction and text mining practices.

One problem in many information extraction applications is the creation of templates, which are textual patterns used to identify information of interest. Our framework describes formally what a template is and covers other typical information extraction tasks. We show how common search algorithms can be used to create and optimise templates automatically, using sequences of overlapping templates, and we develop heuristics that make this search feasible. Finally we demonstrate a successful implementation of the framework and apply it to a typical biological information extraction task.

1 Introduction

Information extraction (IE) [7] has developed over recent decades with applications analysing text from news sources [8], financial sources [6], and biological research papers [1,5,12]. Competitions such as MUC and TREC have been promoted as using real text sources to highlight problems in the real world, and more recently TREC has included a genomics track [11], again highlighting biology and medicine as growing areas of IE research. It has long been recognised that there is a need to share resources between research groups in order to allow a fair comparison of their different systems and to motivate and direct further research. We strongly feel that there is also a need to provide a *theoretical* framework within which these information extraction

D. Corney et al.: *A Logical Framework for Template Creation and Information Extraction*, Studies in Computational Intelligence (SCI) **118**, 79–108 (2008)
www.springerlink.com

systems can be described, compared and developed, by identifying key issues explicitly. The framework we present here will allow researchers to compare their methods as well as their results, and also provides new methods for template creation. This will aid the identification of important issues within the field, allowing us to identify the questions to ask as well as to formulate some answers.

The terms information extraction and text mining are often used interchangeably. Some authors use the term text mining to suggest the detection of novelty, such as combining information from several sources to generate and test new hypotheses [25]. In contrast, IE extracts only that which is explicitly stated. This framework focuses on IE and template creation, but it also applies to text mining.

Information extraction is a large and diverse research area. One approach widely used is to develop modular systems such as GATE [9], so that each component can be optimised individually. One of the most challenging of these is the template component. A template is a textual pattern designed to identify "interesting" information to be extracted from documents, where "interesting" is relative to the user's intentions. An ideal template can be used to extract a large proportion of the interesting information available with only a little uninteresting information.

Different types of templates exist, but in general, they can be thought of as regular expressions over words and the features of those words. Standard regular expressions match sequences of characters, but IE templates can also match features of words. To guide our discussion, consider the sentence "The cat sat on the mat". Informally, one template that would match that sentence is "The ANIMAL VERB on the FLOOR_COVERING", where "ANIMAL" and "FLOOR_COVERING" are pre-defined semantic categories, and "VERB" is a part-of-speech label. A different template that would match the same sentence is "DETERMINER * * PREPOSITION DETERMINER *", where each "*" is a wildcard, matching any single word, and the other symbols are part-of-speech labels. Any sentence can be matched with a large number of templates, and many templates match a large number of sentences. This makes template creation a challenging problem.

Although it covers several key areas, this paper focuses on template creation. Currently, templates are typically designed by hand, which can be laborious and limits the rapid application of IE to new domains. There have been several attempts at automatic template creation [2,13,19], and there are likely to be more in the future. To the best of our knowledge, no such system has demonstrated widespread applicability, but tend to be successful only within narrow domains. Some systems are effective, but require extensive annotation of a training set [22], which is also laborious.

One way to view the automatic creation of useful templates is as a search problem of a kind familiar to the artificial intelligence community [24, chs. 3, 4]. To formulate it this way, we need to define the space of candidate solutions (i.e. templates); a means of evaluating and comparing these

candidate solutions; a means of generating new candidate solutions; and an algorithm for guiding the search (including starting and stopping). Any useful framework describing IE must provide a way to define and create templates, and our framework proposes using these AI search methods, an idea we expand in Sect. 6.2, where we "grow" useful templates from given seed phrases.

One alternative to using templates is *co-occurrence analysis* [16]. This identifies pieces of text (typically sentences, abstracts or entire documents) that mention two entities, and assumes that this implies that the two entities are in some way related. Within our framework, this can be seen as a special case of a template, albeit a very simple one, as we show in Sect. 2.3.

The framework itself is presented in Sects. 2–5, with the subsequent sections discussing various implementation issues.

Section 2 defines various concepts formally, moving from words and documents to templates and information extraction. Section 3 describes how templates can be ordered according to how specific or general they are, as a precursor to template creation and optimisation. Section 4 discusses how to modify a template to make it more general. Section 5 gives formal definitions of recall and precision within our framework and discusses how they might be estimated in practice. Section 6 discusses heuristic search algorithms and their implementation and includes a detailed example, before a concluding discussion.

A shorter form of this work is published in [4].

2 Basic Definitions

In this section, we define several terms culminating in a formal definition of information extraction templates.

Definition 1. *A* literal λ *is a word in the form of an ordered list of characters. We assume implicitly a fixed alphabet of characters.*

Examples: "cat", "jumped", "2,5-dihydroxybenzoic".

Definition 2. *A* document d *is a tuple (ordered list) of literals:* $d = <\lambda_1, \lambda_2, \ldots, \lambda_{|d|}>$.

Examples: $d_1 =$ <the, cat, sat, on, the, mat>, $d_2 =$ <a, mouse, ran, up, the, clock>.

Definition 3. *A* corpus D *is a set of documents:* $D = \{d_1, d_2, \ldots, d_{|D|}\}$.

Example: $D_1 = \{d_1, d_2\}$.

Definition 4. *A* lexicon Λ *is the set of all literals found in all documents in a corpus:* $\Lambda_D = \{\lambda | \lambda \in d \text{ and } d \in D\}$.

Example: $\Lambda_{D_1} = \{$the, cat, sat, on, mat, a, mouse, ran, up, clock$\}$.

Every word has a set of attributes, such as its part-of-speech or its membership of a semantic class, which we now discuss. Although particular attributes are not a formal part of the framework, they are used in various illustrative examples throughout this paper.

Words that share a common stem, or root, typically share a common meaning, such as the words "sit", "sitting" and "sits". It is therefore common practice in information retrieval to index words according to their stem to improve the performance [21]. Similarly in information extraction, it is often helpful to identify words that share a common stem. The most common approach is to remove suffixes to produce a single stem for each word [21], although in principle, each word could have multiple stems, such as if prefixes were removed independently of suffixes.

Words may also belong to pre-defined semantic categories, such as "business", "country" or "protein". One common way to define these semantic categories is by using gazetteers. A gazetteer is a named list of words and phrases that belong to the same category. Rather than simple lists, some ontologies are based on hierarchies or directed acyclic graphs, such as MeSH[1] and GO[2] respectively. In this framework, we are not concerned with the nature of such categories, but assume only that there exists some method for assigning such attributes to individual words.

The role of each word in a sentence is defined by its *part of speech*, or lexical category. Common examples are noun, verb and adjective, although these are often subdivided into more precise categories such as "singular common noun", "plural common noun", "past tense verb" and so on. The part of speech can usually only be ascertained for a word in a given context. For example, compare "He cut the bread" to "The cut was deep". In practice, an implementation may limit this to exactly one label per word, based on the context of that word. Following the Penn Treebank tags [17], in some examples we use the symbol "DT" to represent determiners such as "the", "a" and "this"; "VB" to represent verbs in their base form, such as "sit" and "walk"; "VBD" to represent past-tense verbs, such as "sat" and "walked"; "NN" to represent common singular nouns, such as "cat" and "shed" and so on.

We also introduce wildcards as an extension to the idea of word attributes. In regular expressions, a wildcard can "stand in" for a range of characters, and we use the same notion here to represent ranges of words. For example, we use the symbol "*" as the universal wildcard which can be replaced by any word in the lexicon. Then every word has the attribute "*". We also use the symbol "?" to represent any word *or no word at all*. We discuss these wildcards further in Sect. 4.3.

[1] MeSH, Medical Subject Headings, `http://www.nlm.nih.gov/mesh`
[2] Gene Ontology, `http://www.geneontology.org`

Other categories may be introduced to capture other attributes, such as orthography (e.g. upper case, lower case or mixed case), word length, language and so on. A parser could be used to label words as belonging to different types of phrases, such as verb phrases and noun phrases. We could also treat punctuation symbols as literals if required, or as a separate category. However, the categories described above are sufficient to allow us to develop and demonstrate the framework.

Definition 5. *A category κ is set of attributes of words of the same type. Common categories include "parts of speech" and "stems".*

For convenience, we will label certain categories in these and subsequent examples. This is not part of the framework but reflects categories likely to be used in a practical implementation. In particular, we use Λ to label the category "literals"; Π for "parts of speech"; Γ for "gazetteers"; Σ for "stems"; and Ω for "wildcards".

Example:
$\kappa_\Lambda = \{\text{the, cat, sat, on, mat, mouse, } \dots\}$
$\kappa_\Sigma = \{\text{the\_stem, cat\_stem, sit\_stem, on\_stem, mat\_stem, mouse\_stem, } \dots\}$
$\kappa_\Pi = \{\text{DT, NN, VBD, IN, } \dots\}$
$\kappa_\Gamma = \{\text{FELINE, RODENT, ANIMAL, FLOOR\_COVERING, } \dots\}$
$\kappa_\Omega = \{*, ?\}$

We use the suffix "_stem" in stem labels to avoid confusing them with the corresponding literal.

Definition 6. *Let K be a set of categories of attributes. Each element κ of K is a single category of word attributes.*

Example: $K_1 = \{\kappa_\Lambda, \kappa_\Sigma, \kappa_\Pi, \kappa_\Gamma, \kappa_\Omega\}$.

Definition 7. *A term t is a value that an attribute may take, i.e. an element of a category of word attributes.*

Examples: $t_1 = \text{cat}$, $t_2 = \text{NN}$, $t_3 = \text{FELINE}$, where $t_1 \in \kappa_\Lambda$, $t_2 \in \kappa_\Pi$, $t_3 \in \kappa_\Gamma$.

Definition 8. *We define a* template element *T to be a set of terms belonging to a single category. Let $T = \{t_1, t_2, \dots, t_n\}$, such that $t_i \in T$. Then $t_i \in \kappa \iff t_j \in \kappa$, $\forall t_j \in T$.*

Examples:
$T_1 = \{\text{NN, VBD}\}$
$T_2 = \{\text{FELINE, RODENT, FLOOR\_COVERING}\}$

The set {NN, FELINE } is not a template element because "NN" and "FELINE" belong to different categories, namely κ_Π and κ_Γ respectively.

The name "template element" refers to templates as defined in Definition 13 below.

Definition 9. *The* attributes *of a literal are the set of template elements defining the values of the literal in each category. We first define the set of attributes of a literal* λ *for a particular category* κ *as* $\alpha(\lambda, \kappa) = \{T | \forall t \in T,\ t \in \kappa$ *and* λ *has attribute* $t\}$*. The set of all attributes of a literal is the union of the attributes in each category:* $\alpha(\lambda) = \bigcup_{\kappa \in K} \alpha(\lambda, \kappa)$*. If a literal has no value for a particular category, then the category is omitted from the set* α*.*

When we say "λ has attribute t", we assume that this relationship is defined outside of the framework. For example, there maybe functions to assign a stem attribute to a word, or to assign a particular semantic category to any of a given list of words.

For convenience, we label the attributes using the category label as a subscript in these examples.

Examples:

$\alpha_{\Lambda}(\text{cat}) = \{\text{cat}\}$

$\alpha_{\Gamma}(\text{cat}) = \{\text{FELINE, ANIMAL}\}$

$\alpha_{\Pi}(\text{cat}) = \{\text{NN}\}$

$\alpha_{\Sigma}(\text{cat}) = \{\text{cat\_stem}\}$

$\alpha(\text{cat}) = \{\{\text{cat}\}, \{\text{NN}\}, \{\text{FELINE, ANIMAL}\}, \{\text{cat\_stem}\}\}$

In the case of α(the), the word "the" has a part-of-speech tag "DT" (determiner) and the obvious literal, but no gazetteer or stem entries. So $\alpha_{\Pi}(\text{the}) = \{\text{DT}\}$, and $\alpha_{\Gamma}(\text{the})$ is undefined, and so $\alpha(\text{the}) = \{\{\text{the}\}, \{\text{DT}\}\}$.

The set of literal attributes Λ has the special property that every word has exactly one literal. Other categories in K may contain terms such that one literal may correspond to one or more terms, or to no term at all. For example, one literal may belong to more than one gazetteer, while another literal may belong to none. Therefore for any λ, $|\alpha_{\Lambda}(\lambda)| = 1$. As a consequence, $|\alpha(\lambda)| \geq 1$.

2.1 Membership of Terms

We now define the concept of membership to refer to the set of literals that share a particular attribute.

Definition 10. *We define the members* μ *of a term* t *as being the set of all literals that share the attribute value defined by that term.* $\mu(t) = \{\lambda | t \in \alpha(\lambda)\}$*. Also, we define the members of a set of terms (such as a template element) as the union of the members of each term in the set:* $\mu(\{t_1, t_2, \ldots, t_n\}) = \bigcup_{i=1}^{n} \mu(t_i)$

Examples:

$\mu(\text{sit\_stem}) = \{\text{sit, sits, sat, sitting }\}$.

$\mu(\text{FELINE}) = \{\text{cat, lion, tiger, } \ldots\}$.

$\mu(\text{RODENT}) = \{\text{mouse, rat, hamster, } \ldots\}$.

$\mu(\{\text{FELINE}, \text{RODENT}\}) = \{\text{cat, lion, tiger, mouse, rat, hamster, } \ldots\}$.

$\mu(\alpha_{\Lambda}(\text{cat})) = \{\text{cat}\}$. I.e. the membership of the literal category of a literal is the set containing only the literal itself.

Definition 11. *We also define membership for a term limited to a particular document or set of documents:* $\mu(t,d) = \{\lambda|\lambda \in d, \lambda \in \mu(t)\}$*, and* $\mu(t,D) = \bigcup_{d\in D} \mu(t,d)$

Examples:
$\mu(\text{NN}, d_1) = \{\text{cat, mat}\}$.
$\mu(\text{ANIMAL}, D_1) = \{\text{cat, mouse}\}$.

2.2 Templates and Document Fragments

Definition 12. *We define a* fragment *of a document as being a tuple of successive literals taken from some document d. If*

$$d = <\lambda_1, \lambda_2, \ldots, \lambda_a, \lambda_{a+1}, \ldots, \lambda_{a+b-1}, \ldots, \lambda_{|d|}>,$$

then

$$f(d,a,b) = <\lambda_a, \lambda_{a+1}, \ldots, \lambda_{a+b-1}>.$$

I.e. the function $f(d,a,b)$ *returns a tuple of b words in order, from d, starting with the* a^{th} *word.* f_1 *is the first word of the fragment, i.e.* λ_a*,* f_2 *is the second word,* λ_{a+1}*, and so on. Note that* $|f| = b$*.*

Example: If $d_1 = <$the, cat, sat, on, the, mat$>$, then $f(d_1, 4, 3) = <$on, the, mat$>$.

Definition 13. *A* template τ *is a tuple of one or more template elements,* $<T_1, T_2, \ldots, T_n>$*, where* $T_1 = \{t_{1,1}, t_{1,2}, \ldots, \}, T_2 = \{t_{2,1}, t_{2,2}, \ldots\}$ *and so on.* $|\tau|$ *is the number of template elements in template* τ*, and is always greater than zero. Each template element* T_i *within a template consists of one or more terms of the same type.*

Example:
$\tau_1 = <\{\text{the}\}, \{\text{FELINE, RODENT}\}, \{\text{VB, VBN}\}>$.

Definition 14. *A template* matches *a fragment of a document d if each successive template element in the template contains a term whose membership includes each successive word in the fragment. Let* $f = <f_1, \ldots, f_n>$ *be a fragment. Given a template* $\tau = <T_1, T_2, \ldots, T_n>$*, we extend the membership function thus:*

$$\mu(\tau, d) = \{f | f \in d \text{ and } \forall i \in 1 \ldots |\tau|, f_i \in \mu(T_i)\}$$

For a corpus D, the template matches the union of the template membership for each document: $\mu(\tau, D) = \bigcup_{d\in D} \mu(\tau, d)$*.*

This function returns a set of fragments, each of which consists of a tuple of literals that matches each successive element of the template τ, and each of which is found in a document in the corpus. Matching terms in this way is

the core of information extraction. The words that are matched define the information that is to be extracted.
Example: Given a template τ_1 = <{DT}, {ANIMAL}, {VBD}> and a corpus D_1 (as defined after Definition 3), then $\mu(\tau_1, D_1)$ = {<the, cat, sat>, <a, mouse, ran>}.

2.3 Co-occurrence Analysis

Co-occurrence analysis assumes that two entities in the same piece of text are related, without attempting a more sophisticated linguistic analysis of the text. In our framework, this can be represented by a template (or set of templates) that defines the two entities with a series of wildcards between them.

For example, suppose we use co-occurrence analysis to discover every sentence in a corpus that mentions two entities, as matched by template elements T_i and T_j. Let us assume that all sentences to be considered are finite with a maximum length of Q words. Then we could define two template $\tau_1 = <T_1, \ldots, T_i, \ldots, T_j, \ldots, T_Q>$ and $\tau_2 = <T_1, \ldots, T_j, \ldots, T_i, \ldots, T_Q>$. Two templates are required if we wish to allow for sentences with the two entities in different orders. We replace every template element except for T_i and T_j with the wildcard element "?" so as to match any sentence containing words that match our terms. With three or more entities, larger sets of templates may be required.

3 Template Ordering

One motivation for creating this framework is to enable the use of common search algorithms for template creation. To do this effectively, we must define an *ordering* over the templates, which we can then use to develop practical search heuristics.

For any given document, each template matches a certain number of fragments. A template that matches every possible fragment is useless, as is one that matches no fragments at all. Somewhere between these two extremes of generic templates and specific templates, lie useful templates that match the interesting fragments only, so the aim of template creation is to find a suitable trade-off between the generic and the specific. We therefore suggest that a useful ordering is one based on the number of fragments that a template is likely to match. We can use such an order to search across a range of templates and explore the trade-off. For unseen text, it is impossible to predict the amount of information to be extracted in advance, so instead, we develop a heuristic ordering that approximates it.

In this section, we define possible orderings of terms and templates. In the next section, we define algorithms that use these orderings to modify

templates. In Sect. 6 we discuss some search algorithms that might be used to locate optimal templates.

Before we consider an ordering over templates, we consider ordering over the terms that make up a template. We want to define the relations $t_1 >_s t_2$ to mean that term t_1 is a more specific attribute than term t_2, and $t_1 \geq_s t_2$ to mean that t_1 is at least as specific as t_2. Similarly, $t_1 <_s t_2$ and $t_1 \leq_s t_2$ mean "less specific than" and "no more specific than" respectively. We now define these by introducing "superset ordering".

3.1 Superset Ordering of Terms and Template Elements

We start by defining a specificity ordering over terms such that each term matches every word that its antecedent matches, along with zero or more extra words. We call this superset ordering.

Definition 15. *Let $\geq_s$ be the ordering over superset specificity. Let t_1, t_2 be terms. If $\mu(t_1) \subseteq \mu(t_2)$ then $t_1 \geq_s t_2$, and we say that t_1 is at least as specific as t_2. If $\mu(t_1) \subset \mu(t_2)$ then $t_1 >_s t_2$, and we say that t_1 is more specific than t_2.*

Examples:

Let FELINE and ANIMAL be two terms (specifically, gazetteers), such that μ(FELINE) = {cat, lion, tiger, ...} and μ(ANIMAL) = {antelope, dog, cat, ..., lion, ..., tiger, ..., zebra}. Then μ(FELINE) $\subset$ μ(ANIMAL) and so FELINE $>_s$ ANIMAL.

$\mu(\alpha_\Lambda(\text{cat})) = \{\text{cat}\}$.

$\alpha_\Gamma(\text{cat}) = \{\text{FELINE, ANIMAL}\}$

$\mu(\{\text{FELINE, ANIMAL}\}) = \{\text{cat, lion, tiger, antelope, dog, ...}\}$

Therefore $\mu(\alpha_\Lambda(\text{cat})) \subset \mu(\alpha_\Gamma(\text{cat}))$.

Some specificity orderings are dependent on the categories of the two terms. For example, by definitions 9 and 10, it is clear that each literal is a member of all the terms that are attributes of the literal. Therefore $\mu(\lambda) \subseteq \mu(\alpha_\lambda) \forall \kappa \in K$ and therefore if $t_1 \in \kappa_\Lambda$, then $\forall t_2 \in K,\ t_1 \geq_s t_2$. In other words, terms that represent literals are at least as specific as any other terms. At the other extreme, the wildcards "*" and "?" match every word, so we can say that wildcard terms are no more specific than any other term. I.e. if $t_1 \in \kappa_\Omega$, then $\forall\, t_2 \in K,\ t_2 \geq_s t_1$.

Having defined an ordering over terms, we now consider sets of terms, and template elements in particular. Let T_1 and T_2 be two template elements. We say that T_1 is at least as specific as T_2 if and only if every literal matched by any term in T_1 is also matched by some term in T_2:

$$T_1 \geq_s T_2 \iff \forall \lambda \in \mu(T_1),\ \lambda \in \mu(T_2)$$

Similarly, $T_1 >_s T_2 \iff \forall \lambda \in \mu(T_1),\ \lambda \in \mu(T_2)$ and $\exists \lambda \in \mu(T_2)$ such that $\lambda \notin \mu(T_1)$. I.e. T_1 is more specific than T_2 if every literal matched by a

term in T_1 is also matched by a term in T_2, and at least one literal is matched by a term in T_2 and not by a term in T_1.

We can trivially extend these "more specific than" relations to define "less specific than" and "equally specific as" relations:

$$t_1 >_s t_2 \iff t_2 <_s t_1$$

$$t_1 \geq_s t_2 \iff t_2 \leq_s t_1$$

$$t_1 =_s t_2 \iff t_1 \geq_s t_2 \text{ and } t_1 \leq_s t_2$$

In this last case, $t_1 =_s t_2 \iff \mu(t_1) = \mu(t_2)$. Note that this is a narrower definition than requiring that $|\mu(t_1)| = |\mu(t_2)|$.

3.2 Ordering of Templates

Above, we have discussed ordering of terms and of template elements. Here we generalise this to discuss entire templates. We want to be able to compare two templates, τ_1 and τ_2, and say which is more specific, i.e. which one matches fewer fragments. If τ_1 and τ_2 are related through superset generalisation, then for a given set of documents D, this is testing whether $\mu(\tau_1, D) \supset \mu(\tau_2, D)$ and therefore whether $|\mu(\tau_1, D)| > |\mu(\tau_2, D)|$, or vice versa, or they are equal. E.g. $\tau_1 >_s \tau_2 \iff \mu(\tau_1, D) \subset \mu(\tau_2, D)$. This depends on the corpus D and is potentially slow to evaluate, especially if D contains a large number of documents. We would rather have an estimate of the relative specificity which is independent of D, which will be useful when developing search heuristics.

Suppose τ_1 and τ_2 are almost identical, and differ only in one template element:

$$\tau_1 = <T_1, T_2, \ldots, T_i, \ldots, T_n>$$

$$\tau_2 = <T_1, T_2, \ldots, T'_i, \ldots, T_n>$$

where $T_i \neq T'_i$. Then we can say that $\tau_1 >_s \tau_2 \iff T_i >_s T'_i$.

More generally, if τ_1 and τ_2 contain the same number of sets of terms, then

$$\tau_1 \geq_s \tau_2 \iff T_{1,i} \geq_s T_{2,i} \;\; i = 1 \ldots |\tau_1| \text{ , and}$$

$$\tau_1 >_s \tau_2 \iff T_{1,i} \geq_s T_{2,i} \;\; i = 1 \ldots |\tau_1| \text{ and } T_{1,j} >_s T_{2,j} \text{ for some } j.$$

In a slight extension to previous notation, we use $T_{n,i}$ to refer to the i^{th} element of template τ_n.

Also, if two templates are identical except that one is "missing" the first or last template element of the other, then the shorter of the two is less specific. I.e. if $\tau_1 = <T_1, T_2, \ldots, T_{n-1}, T_n>$, $\tau_2 = <T_1, T_2, \ldots, T_{n-1}>$ and $\tau_3 = <T_2, \ldots, T_{n-1}, T_n>$ then $\tau_1 \geq_s \tau 2$ and $\tau_1 \geq_s \tau 3$.

Although these relationships do not provide a complete ordering over all templates, they do allow us to compare similar templates, and this is sufficient to allow us to create and modify templates, and to develop useful search heuristics. We use this ordering to develop functions that create and modify templates in Sect. 4, and to develop methods to search efficiently for *good* templates in Sect. 6.

4 Template Generalisation

Whether created manually or automatically, templates are usually based on examples of "interesting" phrases. These phrases may be identified by hand (e.g. by a domain expert) or automatically (e.g. by information retrieval methods). These phrases are then used as "seeds" to help define more general templates. In this framework we will follow this "seed phrase" approach, and assume that we have some suitable phrases. We show how a wide range of templates can be created from each seed phrase. We first discuss how terms can be created and generalised, and then expand this to template creation and generalisation (Sect. 4.2).

4.1 Creating and Modifying Single Template Elements

We now bring together several concepts discussed above, and define functions that create and generalise template elements. This leads onto a discussion of creating and modifying entire templates.

Definition 16. *Given a literal, we want to create a new template element, which is simply a set containing the literal. We define the trivial function* initialise *for this purpose:* $\textsf{initialise}(\lambda) = \{\lambda\}$.

Have created a template element, we can then modify it. We now define a function that modifies any given template element to produce a new set of template elements that is at least as general as the element given. This is based on the notion of superset ordering (Sect. 3.1) in that the new template elements match a superset of the literals matched by the original template element. Furthermore, the new set of elements belongs to a specified category which is different from the category of the source element.

Definition 17. *We define a function to create a more general set of template elements from a given template element, such that all of the terms in each new template element are members of a specified category. Given a template element* $T = \{t_1, t_2, \ldots, t_{|T|}\}$ *of category* κ *and given a target category* $\kappa' \neq \kappa$*, we create a set of template elements* $\{T'_1, T'_2, \ldots, T'_n\}$*:*

$$\textsf{generalise}(T, \kappa') = \Bigg\{ T' | T' = \{t'_1, t'_2, \ldots, t'_m\},\ \textit{and}\ t'_1, t'_2, \ldots, t'_m \in \kappa',\ \textit{and}$$
$$\forall t'_i \in T',\ |\mu(t'_i) \cap \bigcup_{t \in T} \mu(t)| \geq 1\ ,\ \textit{and}$$
$$\bigcup_{t \in T} \mu(t) \subseteq \bigcup_{t' \in T'} \mu(t'),\ \textit{and there is no set}\ \{t'_p \ldots t'_q\}$$
$$\textit{such that}\ \{t'_p \ldots t'_q\} \subset T'\ \textit{and}\ \bigcup_{t \in T} \mu(t) \subseteq \bigcup_{j=p}^{q} \mu(t'_j) \Bigg\}.$$

I.e. For each template element produced by generalise(T, κ'), each term within that element belongs to category κ'; and each term within that element matches at least one literal matched by a term in T; and every literal matched by a term or terms in T is matched by at least one term in the template element; and that no subset of terms exists that meets these two requirements. Note that $\bigcup_{t \in T} \mu(t)$ is the set of all literals that are members of terms in the original template element T, and that $\bigcup_{t' \in T'} \mu(t')$ is the set of all literals that are members of terms in the new template elements T'. Each new template element has to be different from the original template element, as they belong to different categories. This ensures that the new element is *more* general than the original, and not just *as* general.

Example: Let category κ_Γ = {FELINE, CANINE, ANIMAL} contain three sets of literals:

$\mu(FELINE) = \{\text{cat}\}$,
$\mu(CANINE) = \{\text{dog}\}$, and
$\mu(ANIMAL) = \{\text{cat, dog, mouse, horse}\}$.

Let $T = \{t_1, \ldots, t_n\}$ such that $\bigcup_{i=1}^{m} \mu(t_i) = \{\text{cat, dog}\}$. Then $|\mu(FELINE) \cap \bigcup_{i=1}^{m} \mu(t_i)| = 1$, therefore the set $FELINE$ can be considered for inclusion in the sets in generalise(T, κ_Γ). $|\mu(CANINE) \cap \bigcup_{i=1}^{m} \mu(t_i)| = 1$, therefore the set $CANINE$ can be considered for inclusion in the sets in generalise(T, κ_Γ). $|\mu(ANIMAL) \cap \bigcup_{i=1}^{m} \mu(t_i)| = 2$, therefore the set $ANIMAL$ can be considered for inclusion in the sets in generalise(T, κ_Γ). The sets $\{FELINE\}$ and $\{CANINE\}$ are individually insufficient to represent all the literal members of generalise(T, κ_Γ) and so will not be in generalise(T, κ_Γ). The remaining candidate sets are

{{ANIMAL},
{FELINE, CANINE}, {FELINE, ANIMAL}, {CANINE, ANIMAL},
{FELINE, CANINE, ANIMAL}}.

Note that {ANIMAL} $\subset$ {FELINE, ANIMAL}, {ANIMAL} $\subset$ {CANINE, ANIMAL}, and {ANIMAL} $\subset$ {FELINE, CANINE, ANIMAL}. Likewise {FELINE, CANINE} $\subset$ {FELINE, CANINE, ANIMAL}. As a result, {FELINE, ANIMAL}, {CANINE, ANIMAL} and {FELINE, CANINE, ANIMAL} are excluded from generalise(T, κ_Γ), because there are subsets of these sets that meet the conditions of generalise. Therefore generalise(T, κ_Γ) = {{FELINE, CANINE}, {ANIMAL}}. This is a set of two template elements, the first consisting of two terms and the second consisting of one term.

Note that the cardinality of the sets returned by generalise(T, κ_Γ) is not necessarily an indication of their specificity. For example, the set {ANIMAL} matches more literals but represents a single semantic category, while the set {FELINE, CANINE} matches fewer literals but combines two semantic

categories. In an implementation we may decide to use the set {FELINE, CANINE} which is a more cumbersome, but more specific set of terms than {ANIMAL}, depending on our requirements. The decision made regarding which set to use in a template would depend on the application and on the details of the heuristic searches, as we discuss in Sect. 6.

Note also that generalise will return an empty set if no generalisation exists that matches all the required literals. Thus if the input set contains a literal that is not contained in any member of κ_x, then $\mathsf{generalise}(T, \kappa_x) = \emptyset$.

Example: $\mathsf{generalise}(\{\text{the}\}, \kappa_\Gamma) = \emptyset$, because the literal "the" is not in any gazetteer.

4.2 Creating and Modifying Entire Templates

In the previous section, we defined the creation and generalisation of template elements. Templates are ordered lists of template elements (Definition 13), and we now apply the above concepts to create and modify templates. Given a seed phrase, in the form of a tuple of literals (i.e. a fragment), we can easily define a very specialised template that matches only that fragment. We can then modify this to increase its generality.

Definition 18. *We extend the* initialise *function to create a specialised template from a fragment.*

$$\mathsf{initialise}(<\lambda_1, \lambda_2, \ldots, \lambda_n>) = <\{\lambda_1\}, \{\lambda_2\}, \ldots, \{\lambda_n\}>.$$

This can also be written as: $\mathsf{initialise}(f) = f$*, where* f *is a text fragment.*

We define a new function that generalises any given template to create a new set of templates by modifying a single element of the template using the element generalisation function defined in Sect. 4.1. One template will be created for each possible generalisation of the specified template element.

Definition 19. *Given the template,* $\tau = <T_1, T_2, \ldots, T_i, \ldots, T_n>$*. Then*

$$\mathsf{generalise}(\tau, \kappa, i) = \{\tau' | T'_i \in \mathsf{generalise}(T_i, \kappa) \text{ and } \tau' = <T_1, T_2, \ldots, T'_i, \ldots, T_n>\}$$

I.e. we replace the i^{th} template element with the result of its own generalisation.

Example: Let $\tau_1 =$ <the, cat, sat>. Then $\mathsf{generalise}(\tau_1, \kappa_\Gamma, 2) =$ {<the, FELINE, sat>, <the, ANIMAL, sat>}. In this case, $\mathsf{generalise}(\tau_1, \kappa_\Gamma, 2)$ returns two templates because the second literal "cat" belongs to two gazetteers. In contrast, $\mathsf{generalise}(\tau_1, \kappa_\Gamma, 1) = \emptyset$, because the first literal "the" does not belong to any gazetteer in the category κ_Γ.

4.3 Matching Fragments of Differing Lengths

So far, we have assumed that all templates are generated from a seed phrase by replacing literals in that phrase with other attributes. This restricts the templates created to be the same length as the seed phrase, and so will only match fragments of this fixed length. This is clearly undesirable, so we will now extend the framework to allow for templates that match fragments of various lengths, firstly to shorter fragments and then to longer ones.

Suppose we want to be able to define a template that can match a particular class of noun phrase, in the form of a determiner, zero or more words (e.g. adjectives), and an animal. To do this, we use the wildcard attribute "?" which matches any single word *or no word at all*. This can be used to define templates such as $\tau =$ <DT, ?, ?, ANIMAL>.

To use the "?" wildcard in our examples, we assume that "?" is in the wildcard category κ_Ω and use the existing $\mathsf{generalise}(T, \kappa_\Omega)$ function. So for example, we take a fragment and apply a series of generalise functions after initialisation like this:

$$\begin{aligned}
\tau_1 &= \text{<the, quick, brown, fox>} &&= \mathsf{initialise}(\text{the, quick, brown, fox}) \\
\tau_2 &= \text{<the, ?, brown, fox>} &&\in \mathsf{generalise}(\tau_1, \kappa_\Omega, 2) \\
\tau_3 &= \text{<the, ?, ?, fox>} &&\in \mathsf{generalise}(\tau_2, \kappa_\Omega, 3) \\
\tau_4 &= \text{<DT, ?, ?, fox>} &&\in \mathsf{generalise}(\tau_3, \kappa_\Pi, 1) \\
\tau_5 &= \text{<DT, ?, ?, ANIMAL>} &&\in \mathsf{generalise}(\tau_4, \kappa_\Gamma, 4)
\end{aligned}$$

Consider a document containing these fragments:

$$\begin{aligned}
f_1 &= \text{<the, cat>} \\
f_2 &= \text{<the, lazy, dog>} \\
f_3 &= \text{<the, quick, brown, fox>}
\end{aligned}$$

Of the above list of templates, τ_5 matches all three fragments, whereas $\tau' =$ <DT, *, *, ANIMAL> would not match f_1 or f_2.

Although this approach allows templates to match fragments of varying length as desired, it is restricted to matching fragments that are no longer than the original seed. The seed phrase must therefore be chosen with this in mind, or else modified before the search begins by inserting wildcards.

One solution is to extend the generalise function so that it inserts a wildcard into the template, making the template longer and allowing it to match phrases longer than the seed phrase. For example, we could define $\mathsf{generalise}(\tau, \kappa_{\Omega'}, i)$ as a function that creates new templates from template τ by inserting wildcards from $\kappa_{\Omega'}$ between template elements i and $i+1$. The category $\kappa_{\Omega'}$ contains the same wildcards as κ_Ω, with the distinction that they are inserted *after* existing template elements, rather than used to replace template elements. Introducing extra wildcards in this way complicates

the search algorithms substantially, so restrictions would have to be introduced, either limiting how many wildcards could be added to a template, or limiting the maximum length of a template. These heuristics could be chosen for a particular implementation after experimentation.

Note that there can be no gain from inserting "?" wildcards before the first non-wildcard template element, or after the last non-wildcard element, as the resulting template would match exactly the same fragments with or without these optional "external" wildcards.

5 Measuring Template Quality

In this section, we define recall, precision and related terms formally within our framework. We then discuss how these can be estimated in practice. These measures are needed to guide the automatic search for templates discussed in the next section.

5.1 Defining Recall and Precision

We have already defined a function $\mu(\tau, D)$ that retrieves the set of document fragments that are matched by template τ from document set D (Definition 14).

Definition 20. *We define $I(D)$ as the set of interesting, relevant fragments contained in corpus D.*

The ideal template would match these, and only these, fragments. This set is generally not known[3], but allows us to define concepts such as "true positive" and the precision rate. Figure 1 shows the relationships diagrammatically. We use the notation $\wp_f(D)$ to show the set of all fragments in D, independent of any template. This is the set of all tuples of all lengths obtainable as fragments using $f(d, a, b)$ (Definition 12).

We now define a series of sets with respect to a template τ and a corpus D.

Definition 21. *True-positives* $TP(\tau, D) = \mu(\tau, D) \cap I(D)$

Definition 22. *False-positives* $FP(\tau, D) = \mu(\tau, D) \setminus I(D)$

Definition 23. *False-negatives* $FN(\tau, D) = I(D) \setminus \mu(\tau, D)$

Definition 24. *True-negatives* $TN(\tau, D) = D \setminus \{I(D) \cup \mu(\tau, D)\}$

Definition 25. *Recall* $r(\tau, D) = \frac{|\mu(\tau,D) \cap I(D)|}{|I(D)|} = \frac{|TP|}{|I(D)|}$

[3]This set is task-dependent and user-dependent, as different people will pick different fragments as being interesting, even if attempting the same task [3].

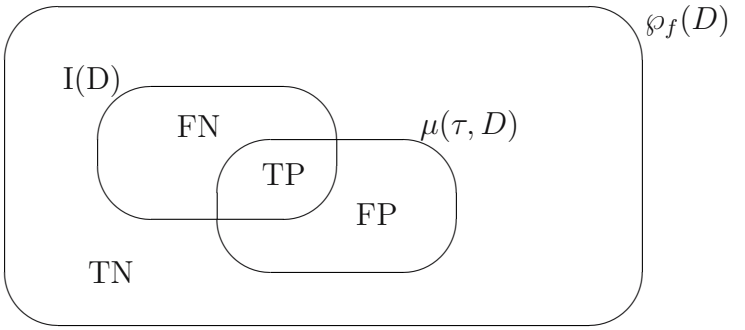


Fig. 1. Venn diagram showing the relationship between: the set of all possible fragments $\wp_f(D)$; the results of applying template τ to it – match(τ, D); the ideal set of results $I(D)$; and the false negative (FN), true positive (TP), false positive (FP) and true negative (TN) regions

Definition 26. *Precision* $p(\tau, D) = \frac{|\mu(\tau,D) \cap I(D)|}{|\mu(\tau,D)|} = \frac{|TP|}{|\mu(\tau,D)|}$

We can conceive of a "perfect" template, τ^*, which matches all the positive fragments, and nothing else. I.e. $\mu(\tau^*, D) = I(D)$. This has a recall and a precision of one. Although such a perfect template may not be known in general, we can use these definitions of true positive and false positive counts to guide a search for templates.

An ideal template would have $|TP(\tau, D)| = |I(D)|$ and $|FP(\tau, D)| = 0$. We therefore wish to maximise $|TP(\tau, D)|$ while minimising $|FP(\tau, D)|$, but there is typically a trade off between the two. This is an example of multi-objective optimisation. If we knew the relative value of true positives and the cost of false positives, then we could combine these into a single objective function, such as maximising $|TP(\tau, D)| - k \cdot |FP(\tau, D)|$. In practice, such a weighting is not usually available, but a number of evolutionary approaches have been successfully applied to similar problems [10], as we discuss further in Sect. 7.

6 Searching for Good Templates

In 6.2 we will discuss the development of search algorithms, but first we need a practical way to estimate recall and precision.

6.1 Estimating Recall and Precision

As noted earlier, we do not know which fragments are interesting *a priori*, and so the above definitions cannot be used directly in calculating the recall or precision of a template. Instead, we need something that we can measure

in practice, and which should approximate the "ideal" values above. We now consider several options.

Suppose that we had a set of documents such that every fragment was labelled as either "interesting" or "not interesting". Then we could use standard supervised learning algorithms to construct useful templates and directly measure the number of true positives, false positives and so on, to find an optimal template. This could then be used to find further information in the same field. However, while a small number of such labelled corpora do exist (e.g. [14]), they are for a few very precisely defined application areas, and so of little general use, as they cannot be used to aid IE in other application areas. Annotating documents in this way is very time consuming for a domain expert, and one aim of information extraction is to reduce the time and effort required to find relevant information.

Suppose instead that we have one set of documents where for each sentence, the probability that it contains a relevant piece of information is above some threshold, and a second set of documents, where the probability is below a threshold. We could then treat this as a classification problem, albeit with noisy labels on the data. However, such a set of irrelevant documents is hard to define, and furthermore, even interesting documents are likely to contain irrelevant facts such as background information, although this approach has been used successfully [23].

Suppose that instead of having irrelevant (negative) documents, we have a set of "neutral" documents, each of which may or may not contain relevant information. I.e. we have no prior knowledge about relevant information in neutral documents. We can then compare the proportion of information retrieved from neutral and from positive documents to evaluate a template. We assume that a "good" template will retrieve more information from positive documents than from neutral documents, even if we don't know in advance which pieces of information are useful, or how much useful information exists in any particular document.

Let D be a corpus containing $|D|$ documents. We define the set of positive documents as D^+ and neutral documents as D^N, such that $D = D^+ \cup D^N$ and $D^+ \cap D^N = \emptyset$. A "positive" document is one that the user believes is likely to contain information of interest. A "neutral" document is one where the user has no reason to believe that the document does or does not contain information of interest.

We now use these two sets of documents to define estimates of the numbers of true-positive fragments and false-positive fragments matched by a template τ.

Definition 27. *We define an estimated true-positive set* $\widehat{TP}(\tau, D) = \mu(\tau, D^+)$*, for a template* τ *and a set of positive documents,* $D^+ \subseteq D$*.*

Definition 28. *We define an estimated false-positive set* $\widehat{FP}(\tau, D) = \mu(\tau, D^N)$*, for a template* τ *and a set of neutral documents,* $D^N \subseteq D$*.*

These are very crude estimates, as they assume that every fragment matched by τ in D^+ contains information of interest, and that every fragment matched by τ in D^N contains *no* information of interest. Thus they cannot be used to estimate the precision or recall scores[4], but they are sufficient to guide the search of useful templates. In fact, as a successful search progresses and the quality of the template improves, then these estimates will become more accurate, although they are unreliable, especially at the start of the search process.

Let us consider some other properties of templates generated using superset generalisation. Suppose we have two templates, τ_1 and τ_2, and that $\tau_1 >_s \tau_2$. Then by definition, $|\mu(\tau_1, D)| < |\mu(\tau_2, D)|$. If τ_2 is a generalisation of τ_1 derived using superset generalisation, then $\mu(\tau_1, D) \subset \mu(\tau_2, D)$. This relative specificity relation holds for any set of documents, so if one template matches fewer fragments than another in one corpus, then it will in any other corpus as well. Thus given two corpora D^a and D^b:

$$|\mu(\tau_1, D^a)| < |\mu(\tau_2, D^a)| \iff |\mu(\tau_1, D^b)| < |\mu(\tau_2, D^b)|.$$

This property will be useful in developing heuristic search methods, because it allows us to rationally prune search graphs, as we discuss in Sect. 6.2.

We defined terms such as "true positive" and "false positive" above. Now we can say that if template τ_1 is at least as specific as template τ_2, then the number of true positives returned by τ_1 is no more than the number returned by τ_2, and equivalently for other scores. I.e. if $\tau_1 \geq_s \tau_2$ then:

$$\begin{aligned}
|\mathrm{TP}(\tau_1, D)| &\leq |\mathrm{TP}(\tau_2, D)|. \\
|\mathrm{FP}(\tau_1, D)| &\leq |\mathrm{FP}(\tau_2, D)|. \\
|\mathrm{TN}(\tau_1, D)| &\geq |\mathrm{TN}(\tau_2, D)|. \\
|\mathrm{FN}(\tau_1, D)| &\geq |\mathrm{FN}(\tau_2, D)|.
\end{aligned}$$

The equivalent inequalities also hold for the estimates defined above:

$$\begin{aligned}
|\widehat{\mathrm{TP}}(\tau_1, D)| &\leq |\widehat{\mathrm{TP}}(\tau_2, D)|. \\
|\widehat{\mathrm{FP}}(\tau_1, D)| &\leq |\widehat{\mathrm{FP}}(\tau_2, D)|.
\end{aligned}$$

As these relationships hold for any set of documents D, we can predict some properties of templates without fully evaluating them. We can use these properties to guide heuristic searches.

If our assumptions here are correct, then the probability of finding an interesting fragment is higher in positive documents than in neutral documents. We can write this assumption as $p\left(f \in I(D) | f \in \mu(\tau, D^+)\right) > p\left(f \in I(D) | f \in \mu(\tau, D^N)\right)$.

[4]Substitution into definitions 25 and 26 gives recall $\equiv$ precision $\equiv 1$ for every template, which is clearly optimistic.

When comparing two templates, we can say that one is certainly better than the other if either: a) it matches more true positive fragments and no more false positives; or b) it matches fewer false positive fragments and no fewer true positives. Unfortunately, we cannot be certain of whether it is better when c) it matches more true positives *and* more false positives. We need some way of exploring this trade-off, an issue we return to later.

We can now generate a series of templates of varying generality and compare them with each other in order to guide our search for useful templates. We discuss this further in the following section.

6.2 Search Algorithms

As stated in the introduction, heuristic searching requires: definitions of candidate solutions; a means to generate and evaluate candidate solutions; and a suitable search algorithm. We have now defined the candidate solutions (i.e. templates, Definition 13) and means to generate (Sect. 4) and evaluate them (using estimates of true positive and false positive scores given in Sect. 5, Definitions 27–28). We now turn to the search algorithms themselves. First, we define an exhaustive search over all possible templates, given a particular seed phrase. We will then show that in general, this is computationally infeasible, owing to the combinatorial growth of the search space with respect to the length of the template. We then introduce heuristics to make the search feasible while still (we hope) producing good templates, and define and discuss a simple best-first search algorithm. We also discuss possible merits of evolutionary algorithms.

In all cases, we assume that we are given a seed fragment f. The root node of the search corresponds to a template consisting of a tuple of template elements, each containing a single literal: $\tau_{root} = \mathsf{initialise}(f)$ (Definition 18). From this, we can modify each element in the template by a single application of the $\mathsf{generalise}(\tau, \kappa, x)$ function (Definition 19). We can estimate the number of true positives and false positives for each of these new templates, and then decide which template to explore and modify next. The exact number of templates produced at each stage depends on the words themselves, because each $\mathsf{generalise}(\tau, \kappa, x)$ function will return 0, 1 or more templates (see Definition 17 and the accompanying discussion). We must also decide when to terminate the search, as we do not know *a priori* the quality of the best possible template, as we are assuming that we do not have a fully labelled corpus.

A simple exhaustive search method would be to start with a literal template created from the seed phrase using the initialise function. For each element in this template, we then apply the generalisation function, using each category in turn, so that each application generates a new template. We need some fixed order over the categories, but the actual order is not critical here[5].

[5] One example would be to use the order: literals, stems, gazetteers, parts-of-speech and wildcards. This order reflects the typical size of the membership functions.

For simplicity, let us assume that we have an implementation with five categories, where every word has one literal, one stem, one gazetteer, one part of speech, and one wildcard. We can then list all possible templates derived from a two-term phrase, starting with two sets of literals, and ending with two sets of wildcards:

$$
\begin{array}{l}
<\lambda,\lambda><\sigma,\lambda><\gamma,\lambda><\pi,\lambda><\omega,\lambda> \\
<\lambda,\sigma><\sigma,\sigma><\gamma,\sigma><\pi,\sigma><\omega,\sigma> \\
<\lambda,\gamma><\sigma,\gamma><\gamma,\gamma><\pi,\gamma><\omega,\gamma> \\
<\lambda,\pi><\sigma,\pi><\gamma,\pi><\pi,\pi><\omega,\pi> \\
<\lambda,\omega><\sigma,\omega><\gamma,\omega><\pi,\omega><\omega,\omega>
\end{array}
$$

If every literal has exactly $|\alpha|$ attributes, then a seed phrase of $|f|$ literals has $|\alpha|^{|f|}$ possible templates. Thus a 20-word seed phrase consisting of literals having 5 attributes will be matched by $5^{20} \approx 10^{14}$ templates. In practice, some words will have fewer attributes (e.g. words that don't appear in any gazetteers), some will have more, and all may have more than one wildcard. We therefore need to introduce heuristics to make the search feasible, unless the seed fragment is very short.

6.3 Feeding Knowledge Forward

After estimating the numbers of true positives and false positives for any template, we can place a lower bound on those values for all templates that are derived using the generalise function. This is because we know that the derived templates will match a superset of the fragments matched by the ancestor template (Sect. 6.1).

For example, suppose we have a template τ_1 and we evaluate this and find it matches 20 positive fragments and 50 neutral fragments, i.e. $|\mu(\tau_1, D^+)| = 20$ and $|\mu(\tau_1, D^N)| = 50$. If we then modify τ_1 using superset generalisation to create τ_2, then we know that $\mu(\tau_1, D) \subseteq \mu(\tau_2, D)$ for any corpus D. We therefore know that τ_2 matches *at least* 20 positive fragments and *at least* 50 neutral fragments, from D^+ and D^N respectively. Furthermore, any other templates derived using superset generalisation from τ_1 or from τ_2 will also match at least those numbers.

Therefore, as we carry out a search, we can calculate lower bounds on the estimates of true positives and false positives for a wide range of templates without the computational expense of evaluating each template. Instead, we can just choose the best template from the range available, if we use a best-first search. By "best" here, we might mean the template with the (estimated) most true positives, which will tend to produce templates with a high recall, though possibly with a low precision. If instead we choose the template that matches the least false-positive fragments then we will tend to produce templates with

E.g. each stem has only a few members; gazetteers often contain hundreds or thousands; parts-of-speech such as verb or noun contain millions of members and so on.

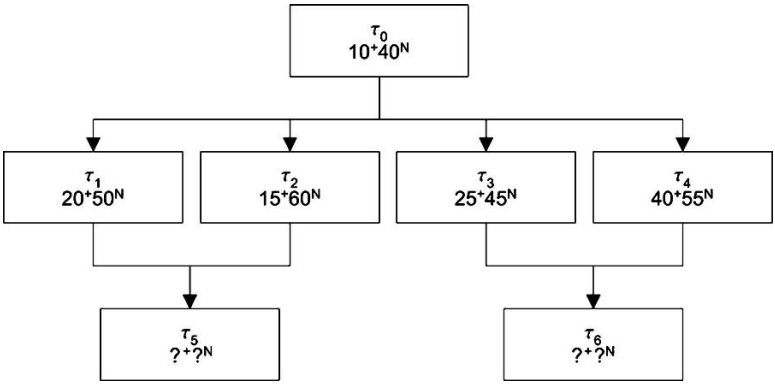


Fig. 2. Part of a search tree. Each node is a template with the number of true positives (x^+) and false positives (y^N) shown, with a "?" for unknown values. Each child node can be created by superset generalisation from any of its parents

a high precision, though possibly with a low recall. Which of these options is more appropriate depends on the task at hand.

Every template has at least one parent (except for the single root template), because they are created using superset generalisation; but most templates can be created in more than one way, from several parent templates. Therefore, many templates will have more than one parent. The whole space may be considered as a lattice. Consider the partial search graph shown in Fig. 2. Here, templates τ_0 – τ_4 have been evaluated, and the number of true positives and false positives are shown for each. For example, τ_1 matches 20 fragments from D^+ and 50 from D^N, and is therefore assumed to match 20 true positives and 50 false positives. At this stage of the search, the decision to be made is which node to evaluate next: τ_5 or τ_6? We can feed forward the facts that the two parent templates of τ_5, τ_1 and τ_2 have 20 and 15 true positives respectively, and that therefore τ_5 must have at least 20 true positives. Similarly, it must have at least 60 false positives. On the other hand, τ_6 must have at least 40 true positives, and at least 55 false positives. We would therefore decide to evaluate τ_6 in preference to τ_5 at this point, because it has a higher lower-bound on the number of true positives, and a lower lower-bound on the number of false positives.

6.4 A Best-First Algorithm

We now present a formal definition of a best-first algorithm suitable for identifying good templates, followed by an example. We start by defining two sets of templates. Set O ("open") contains templates that have been created (via initialise or generalise), but not yet evaluated. Set C ("closed") contains templates that have been evaluated, i.e. had values of TP and FP calculated. Note that $O \cap C \equiv \emptyset$.

1. **begin**
2. $C \leftarrow \emptyset$, $O \leftarrow \emptyset$
3. initialise$(f) = \tau_{root} \rightarrow O$
4. **while not finished do**
   a) find estimated best template τ in O
   b) evaluate τ
   c) delete τ from O
   d) add τ to C
   e) expand τ by adding to O all templates that can be created by superset generalisation of τ
   f) update lower bounds on TP and FP for all templates in O
5. return best template from C.
6. **end**

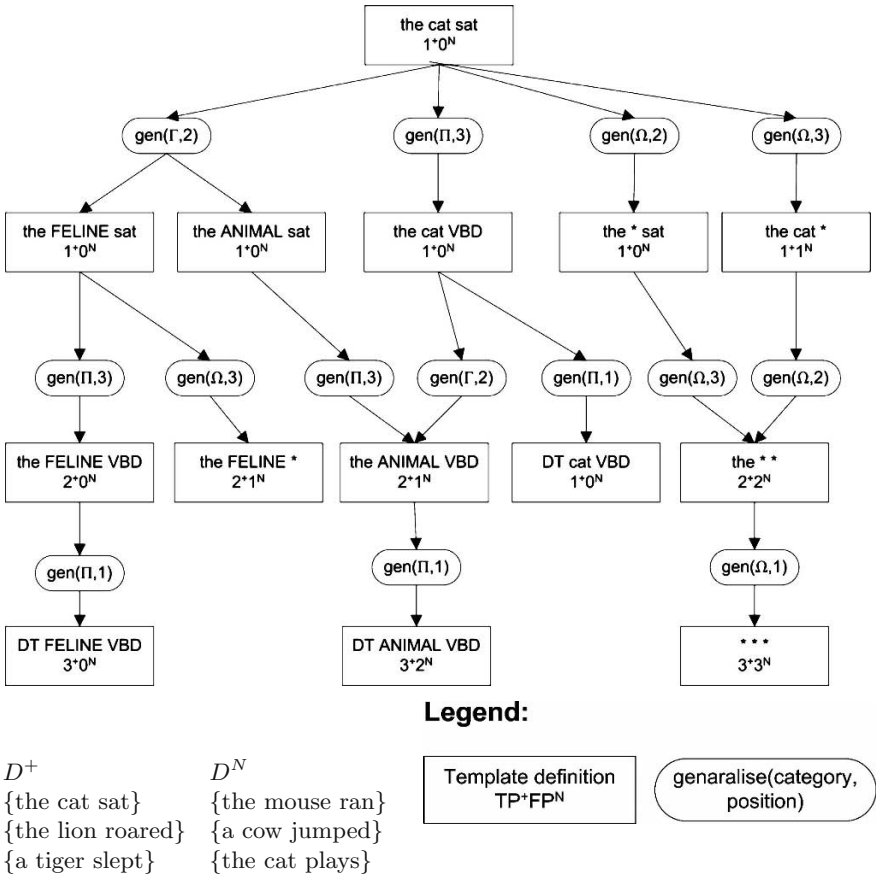
Fig. 3. A best-first algorithm. C is the "closed" set of evaluated templates and O is the "open" set of unevaluated templates. See Sect. 6.4 for further details

Figure 3 gives the algorithm. After initialisation, we take the best template that has not yet been evaluated, and evaluate it, i.e. calculate its true positive and false positive scores. We then generalise it in every way possible to create child templates, and update the lower bounds on the true and false positive scores for these children. Because there are several ways that each template could be created, some children will have more than one parent. In these cases, the lower bounds are the *maximum* of the lower bounds of all the parents.

By "best template" (Fig. 3, Steps 4a and 5), we mean select the template with the highest lower-bound on the number of true positives, as inherited from each template's ancestors. If more than one template has the same maximum true positive lower-bound, then we choose between them by selecting the template with the smallest lower-bound on false positives. If this still selects more than one template, we can either pick one randomly; use them all successively; or use all the selected templates together in the subsequent steps (i.e. evaluate and generalise more than one template in one pass through the main loop).

In Step 4f we could choose to either update the bounds of just the descendents of τ in O, or else update the bounds of the descendents of every template in C. The latter would be more computationally expensive, but would lead to better estimates of the lower bounds. For each new template created, it would require a search through C to find all the other possible ancestors, besides τ, in order to calculate the new lower bounds.

Finally, we terminate the search when some pre-specified criterion is satisfied. Possible criteria include terminating: when O is empty (in which case every possible template has been evaluated); or when a limit on the number of evaluations is reached (e.g. stop after 1,000 templates have been evaluated); or when a certain number of true positives (or false positives) are matched by the current best template. It would be quite possible for the user to stop the search and consider the current best template, before re-starting the search if necessary.



D^+	D^N
{the cat sat}	{the mouse ran}
{the lion roared}	{a cow jumped}
{a tiger slept}	{the cat plays}

Fig. 4. Part of a search graph for a three-term template. Each rectangle shows a template, starting with three sets of literals at the top initialised from the fragment "the cat sat". Various generalisation functions are then applied to it. For example, $\mathsf{gen}(\Gamma, 2)$ means apply the $\mathsf{generalise}(\tau, \kappa_\Gamma, 2)$ function to the upper template, τ, to produce the lower template(s). The graph includes part-of-speech labels "VBD" meaning past-tense verb and "DT" meaning determiner. The numbers in each box below the template represent the estimated numbers of true-positive and false-positive matches for each template, with respect to the positive and neutral document sets D^+ and D^N shown. Note that not every node or edge is shown

As a more concrete example, Fig. 4 shows part of a search graph containing various templates created from the seed fragment "the cat sat", and evaluated with respect to the two small corpora shown.

Consider the right-hand portion of the graph. Near the top are two templates: <the, $*$, sat> and <the, cat, $*$>, both created using the $\mathsf{generalise}(\tau, \kappa_\Omega, x)$ function. The first matches one true positive and no false positives (shown as 1^+0^N in the figure). The second matches one true

positive and one false positive. By the definition of superset generalisation, we know that every template derived from this second template will have at least one true positive and one false positive. So if, at one stage of a search, we only want to consider templates with no false positives, then we need not consider *any* descendent of this template. The two descendents shown (<the, $*, *$> and <$*, *, *$>) need not be evaluated explicitly therefore; they can be annotated as having at least one false positive, although in the figure, the fully evaluated scores are shown.

Now consider the third row of templates in Fig. 4, i.e. those created after two generalisation functions. From the left of the graph, three of these have two true positives (<the, FELINE, VBD>, <the, FELINE, $*$> and <the, ANIMAL, VBD>) and one has only one true positive, so we focus the search on the first three. These have zero, one and one false positives respectively, making the first one look most promising: <the, FELINE, VBD>. This is the best unevaluated template so far. Several further generalisations are possible from this template, including applying generalise$(\tau, \kappa_\Omega, 3)$ to form <the, FELINE, $*$>. But this has already been evaluated, so need not be considered again. Applying generalise$(\tau, \kappa_\Pi, 1)$ forms <DT, FELINE, VBD> which has three true positives, and still no false positives. Given the two very small document sets, this is an optimal template, so we stop the search here. In a more realistic application, the search would continue until a stopping criterion was reached, but would not find any superior template.

One modification to the algorithm would require more memory but should lead to a faster convergence to a (possibly) superior solution. This is to expand the unevaluated template set O by repeated applications of the generalise function until it contains every template that could possibly be derived from the seed phrase, or as many as is practically possible. We would then proceed with a best-first search, but with the advantage that after each evaluation, a large number of unevaluated templates will have the lower bounds of their true- and false-positive estimates updated, because their ancestry would be explicitly represented on the search graph. This should improve the selection of the best template at each stage.

The algorithm above does not prune any part of the search graph, but merely tends to search promising areas first. In practice, this is likely to lead to very large memory requirements, which can be avoided through pruning. Pruning search *graphs* is a lot more difficult than pruning *trees* because each node can have multiple parents, and so after a node is pruned, it may reappear as part of a different path. Although algorithms such as A* are inappropriate here,[6] recent variations may provide useful pruning methods, such as SMA* [24, p. 104] and Sweep A* [27].

[6] We have no notion of a path cost, and are only interested in the final solution rather than the path to it.

6.5 Sample Results

The algorithm outlined in Fig. 3 has been implemented as an extension to the BioRAT tool [5]. As a proof of principle of the methods described here, we used the implementation to derive a template which could be used to identify protein–protein interactions from biological literature.

We start with the seed phrase "Rad53p protein binds to Dbf4p", where "Rad53p" and "Dbf4p" are proteins, and "binds to" suggests a direct physical interaction. We create a positive corpus by extracting 500 abstracts that are listed in the Database of Interacting Proteins (DIP [26]). We assume that most of these abstracts mention at least one protein–protein interaction, although the information will be expressed in many different ways. We also create a neutral corpus by selecting the first 500 abstracts dated September 2005 retrieved from PubMed. This is essentially a random set of biomedical texts. It may contain some references to particular protein–protein interactions, but we assume that most of these abstracts will not.

To select the "best template" during the search (Fig. 3, Steps 4a and 5), we define a quality score thus:

score = (number of positive matches) – $w\times$ (number of neutral matches)

The weight w is used to control the trade-off between true- and false-positives and hence balance the search for high recall and high precision templates. (As explained in Sect. 6.1, we are assuming that each match in the neutral corpus is a false-positive result.) We start with a small value of $w = 0.5$ and slowly increase this as the search progresses. This encourages early exploration of the search space while later penalising false positive matches heavily. We select the unevaluated template with the highest score at each step. In cases where two or more templates have the same score, candidate templates are selected if they contain more gazetteer elements and fewer literal elements than the others. This deliberately introduces a slight bias favouring templates that contain references to gazetteers, as these represent useful domain knowledge. Table 1 summarises the search as it progresses for the first 50 iterations.

On looking at Table 1, we see that the template evaluated on the 30th iteration matches 286 fragments from the positive corpus, and four from the neutral corpus. The template pattern is: [Γ: protein] [Ω: ?????] [Γ: binding] [Ω: ???] [Γ: protein, sp]. This matches a protein followed by between 0 and 5 words, followed by a protein-binding term, followed by between 0 and 3 other words, followed by another protein (of sub-type "sp"; this refers to terms in a gazetteer derived from the SwissProt database). The five sentences listed below were among those found in the positive corpus. The *italicised* portions show the fragments matched by the template; the rest is given for context only:

- *Protein kinase C delta associates with and phosphorylates Stat3* in an interleukin-6-dependent manner.
- Furthermore, *Stat3 was phosphorylated by PKC delta* in vivo on Ser-727...

Table 1. Each row shows the template evaluated at one iteration, starting with the literal template derived from the given seed phrase at iteration zero.

Iteration	Measured matches		Inherited matches		Template pattern
	$\|TP\|$	$\|FP\|$	$\|TP\|$	$\|FP\|$	
0	1	0	–	–	[Λ: Rad53p] [Λ: protein] [Λ: binds] [Λ: to] [Λ: Dbf4p]
10	2	0	1	0	[Γ: protein, sp] [Λ: protein] [Γ: binding] [Ω: *] [Γ: protein, sp]
20	86	1	25	1	[Γ: protein] [Ω: ?] [Γ: binding] [Ω: *] [Γ: protein, sp]
30	286	4	247	4	[Γ: protein] [Ω: ?????] [Γ: binding] [Ω: ???] [Γ: protein, sp]
40	1,202	34	1,202	34	[Ω: ??] [Ω: ?????] [Γ: binding] [Ω: ?????] [Γ: protein]
50	2,621	151	2,621	151	[Ω: ???] [Ω: ?????] [Γ: binding] [Ω: ?????] [Ω: *]

The number of matches in the positive (TP) and neutral (FP) corpora are shown for the current template ("measured matches") and for its parent ("inherited matches"). Each template element is shown delimited by square brackets in the form [CATEGORY: value], with categories literal (Λ), gazetteer (Γ) and wildcard (Ω). A wildcard element with n query symbols matches between 0 and n words. The sequence of templates generated tends to generalise from literals through gazetteers to wildcards

- We report here that *Grb2 also interacts with tyrosine-phosphorylated IRS-1* in response to gastrin.
- ... RII alpha fused to *endonexin II formed dimers but did not bind MAP2.*
- *A protein interaction map* for cell polarity development.

While the first three results correctly show protein–protein interactions, the last two highlight imperfections in this particular template. The phrase "did not bind" indicates that no interaction was observed, but the template still treats this as a positive statement. In the last example shown, the word "map" is not a protein, although the same word is used to refer to a protein in other contexts and so it is in the protein gazetteer.

Table 1 also shows that the template evaluated in the 50th iteration matches 2,621 fragments in the positive corpus, and 151 in the neutral, with the template pattern: [Ω: ???] [Ω: ?????] [Γ: binding] [Ω: ?????] [Ω: *]

This might be seen as an example of over-generalisation, in that the template consists almost entirely of wildcards. However, the quality of any template depends on the requirements of the user, and for some tasks, even this very general template may still be useful. Note that the total runtime on a stand-alone PC was 20 min for these 50 iterations.

7 Discussion and Extensions

We now briefly discuss a few of the possible extensions to the framework and its implementation.

We could introduce other wildcards, such as a wildcard which matches an entire phrase, which could itself be defined as a series of terms, much like a template. This would allow optional subclauses, such as subordinate clauses, to be matched. Let $?^{\tau_1}$ designate an optional wildcard that matches a sequence of literals defined by template τ_1 or matches nothing at all. Then if $\tau_1 =$ <{which}, {*}, {*}>, the template $\tau_2 =$ <{DT}, {ANIMAL}, $\{?^{\tau_1}\}$,{sat}> would match the fragment <the, cat, which, was, black, sat> as well as <the, cat, sat>.

An additional approach to finding good templates is to repeatedly *merge* useful templates to produce more general templates [18]. Our framework could easily be extended to allow this, by ensuring that the product of merging two templates matches every fragment that either template matches. This could be achieved by considering each pair of template elements in turn, and either performing a simple set union if they belong to the same category, or else generalising them both to the same category before such a union. In either case, the new template would match the union of the true-positives matched by the two parents, and the union of the false-positives, allowing the lower-bounds on each to be calculated.

So far, we have considered template that exist in isolation, whereas in practical systems, it is more common to apply a set of templates together. Our framework can be extended to include this by using a sequential covering algorithm. Suppose we have a template τ that matches some true positives and some false positives. We could reduce the number of false positives by creating a second template τ' that is optimised to match just the false positive fragments matched by τ. This could be achieved by defining two new versions of D^+ and D^N based on the fragments matched by τ, and using these to guide the search for τ'. We could then apply τ and τ' together, predicting interesting fragments as $\mu(\tau, D) \setminus \mu(\tau', D)$ (i.e. fragments matched by τ but not by τ'.). In many practical applications, more than one template will be applied to a set of documents, each designed to match a different piece of information, or a different way of expressing that information.

We have assumed that we do not have a set of annotated examples, i.e. fragments known in advance to be positive or negative. Creating and annotating large sets of examples is extremely time consuming for a user, although giving a yes/no response to automatic annotations is simpler [20]. One enhancement to our system therefore would be to start with the estimates of true positive and false positive as outlined above, and search for a good template, and then use this template to annotate a number of fragments and to present these to the user. The user then marks each fragment as interesting or not interesting, and this could then be used to improve the quality of

the function used to estimate the numbers of true and false positives. This improved function could be used to guide a new template search.

The best-first search described above may miss out on good templates because of its greedy decision making. This would be true even if the estimates of the numbers of true and false positives were perfect, owing to the structure of the graph: we can't guarantee that the best parents will have the best children. One likely improvement therefore would be a population based search, such as a simple beam search or an evolutionary algorithm. Evolutionary algorithms have been successfully used to solve a wide range of multi-objective optimisation problems [10], including problems where evaluations must be limited due to time or financial constraints [15]. Extracting information from a large corpus takes considerable computing effort, so this is an aspect worth considering. Multi-objective evolutionary algorithms can efficiently generate a range of Pareto-optimal solutions, and so explore the trade-off between the different objectives. In our case, this means that for each number of true positives, we find the template with the fewest false positives, and for each number of false positives, we find the template with the most true positives. This produces a range of solutions from which the user can then select whichever template or templates are most suitable for their particular problem. This is more flexible than the simple weighting suggested in Sect. 6.5.

Finally, rather than starting with a seed fragment and a template consisting solely of literals, we could start the search using a hand-written template. This would *not* have to be optimised in advance, and in some cases, would be easy to create. The search could then start from a point chosen to be useful and optimised further through similar search processes to those outlined above.

8 Conclusion

We have presented a formal framework to describe information extraction, focusing on the definition of the template patterns used to convert free text into a structured database. The framework has allowed us to explicitly identify some of the fundamental issues underlying information extraction and to formulate possible solutions. We have shown that the framework allows computationally feasible heuristic search methods to be developed for automatic template creation. We have shown that a practical implementation of this framework is feasible and allows automatic template creation. We also hope that the framework will allow other researchers to gain further insights into the theory and practice of information extraction and text mining.

Acknowledgements

This work is partly funded by the BBSRC grant BB/C507253/1, "Biological Information Extraction for Genome and Superfamily Annotation."

References

1. Blaschke, C., Valencia, A.: The Frame-Based Module of the SUISEKI Information Extraction System, *IEEE Intelligent Systems*, **17**(2), March 2002, 14–20
2. Collier, R.: *Automatic Template Creation for Information Extraction*, Ph.D. Thesis, Department of Computer Science, University of Sheffield, 1998
3. Colosimo, M. E., Morgan, A. A., Yeh, A. S., Colombe, J. B., Hirschman, L.: Data Preparation and Interannotator Agreement, *BMC Bioinformatics*, **6**(Suppl 1), 2005
4. Corney, D., Byrne, E., Buxton, B., Jones, D.: A Logical Framework for Template Creation and Information Extraction, *Foundations of Semantic Oriented Data and Web Mining workshop, part of ICDM2005 (the Fifth IEEE International Conference on Data Mining)*, 2005
5. Corney, D. P. A., Buxton, B. F., Langdon, W. B., Jones, D. T.: BioRAT: Extracting Biological Information from Full-Length Papers, *Bioinformatics*, **20**(17), 2004, 3206–3213
6. Costantino, M.: *Financial Information Extraction Using Pre-Defined and User-Definable Templates in the LOLITA System*, Ph.D. Thesis, University of Durham, Department of Computer Science, 1997
7. Cowie, J., Lehnert, W.: Information Extraction, *Communications of the ACM*, **39**(1), 1996, 80–91
8. Cowie, J., Wilks, Y.: Information Extraction, in: *Handbook of Natural Language Processing* (Dale R., Moisl H., Somers H., Eds.), Marcel Dekker, New York, 2000
9. Cunningham, H.: GATE, a General Architecture for Text Engineering, *Computers and the Humanities*, **36**(2), May 2002, 223–254
10. Fonseca, C. M., Fleming, P. J.: An Overview of Evolutionary Algorithms in Multiobjective Optimization, *Evolutionary Computation*, **3**(1), 1995, 1–16
11. Hersh, W., Bhuptiraju, R., Ross, L., Cohen, A., Kraemer, D.: TREC 2004 Genomics Track Overview, *The Thirteenth Text Retrieval Conference (TREC 2004) NIST Special Publication SP 500–261*, 2004
12. Hirschman, L., Yeh, A., Blaschke, C., Valencia, A.: Overview of BioCreAtIvE: Critical Assessment of Information Extraction for Biology, *BMC Bioinformatics*, **6**(Suppl 1), 2005
13. Huang, M., Zhu, X., Hao, Y., Payan, D. G., Qu, K., Li, M.: Discovering Patterns to Extract Protein–Protein Interactions from Full Texts, *Bioinformatics*, **20**(18), 2005, 3604–3612
14. Kim, J., Ohta, T., Tateisi, Y., Tsujii, J.: GENIA Corpus – Semantically Annotated Corpus for Bio-Textmining, *Bioinformatics*, **19**(Suppl 1), 2003, 180–182
15. Knowles, J., Hughes, E. J.: Multiobjective Optimization on a Budget of 250 Evaluations, *Evolutionary Multi-Criterion Optimization (EMO 2005), LNCS 3410*, Springer, Berlin Heidelberg New York, 2005, 176–190
16. Koike, A., Niwa, Y., Takagi, T.: Automatic Extraction of Gene/Protein Biological Functions from Biomedical Text, *Bioinformatics*, **21**(7), April 2005, 1227–1236
17. Marcus, M., Santorini, B., Marcinkiewicz, M. A.: Building a Large Annotated Corpus in English: the Penn Treebank, *Computational Linguistics*, **19**, 1993, 313–330
18. Nobata, C., Sekine, S.: Towards Automatic Acquisition of Patterns for Information Extraction, *International Conference of Computer Processing of Oriental Languages*, 1999

19. Pierce, D., Cardie, C.: Limitations of Co-Training for Natural Language Learning from Large Datasets, *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics Research, 2001
20. Pierce, D., Cardie, C.: User-Oriented Machine Learning Strategies for Information Extraction: Putting the Human Back in the Loop, *Working Notes of the IJCAI-2001 Workshop on Adaptive Text Extraction and Mining*, 2001, 80–81
21. Porter, M. F.: An Algorithm for Suffix Stripping, *Program*, **14**(3), 1980, 130–137
22. Riloff, E.: Automatically Constructing a Dictionary for Information Extraction Tasks, *National Conference on Artificial Intelligence*, 1993, 811–816
23. Riloff, E.: Automatically Generating Extraction Patterns from Untagged Text, *Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI-96)*, 1996, 1044–1049
24. Russell, S., Norvig, P.: *Artificial Intelligence: A Modern Approach*, 2 edition, Prentice-Hall, Englewood Cliffs, NJ, 2003
25. Sehgal, A.: *Text Mining: The Search for Novelty in Text*, Ph.D Comprehensive Examination Report, Department of Computer Science, The University of Iowa, April 2004
26. Xenarios, I., Salwinski, L., Duan, X., Higney, P., Kim, S., Eisenberg, D.: DIP: The Database of Interacting Proteins. A Research Tool for Studying Cellular Networks of Protein Interactions, *Nucleic Acids Research*, **30**(1), January 2002, 303–305
27. Zhou, R., Hansen, E.: Sweep A: Space-Efficient Heuristic Search in Partially-ordered Graphs, *Fifteenth IEEE International Conference on Tools with Artificial Intelligence*, Sacremento, CA, November 2003

A Bipolar Interpretation of Fuzzy Decision Trees*

Tuan-Fang Fan[1], Churn-Jung Liau[2], and Duen-Ren Liu[1]

[1] Institute of Information Management, National Chiao-Tung University, Hsinchu 300, Taiwan
tffan.iim92g@nctu.edu.tw, dliu@iim.nctu.edu.tw
[2] Institute of Information Science, Academia Sinica, Taipei 115, Taiwan
liaucj@iis.sinica.edu.tw

Summary. Decision tree construction is a popular approach in data mining and machine learning, and some variants of decision tree algorithms have been proposed to deal with different types of data. In this paper, we present a bipolar interpretation of fuzzy decision trees. With the interpretation, various types of decision trees can be represented in a unified form. The edges of a fuzzy decision tree are labeled by fuzzy decision logic formulas and the nodes are split according to the satisfaction of these formulas in the data records. We present a construction algorithm for general fuzzy decision trees and show its application to different types of training data.

1 Introduction

Decision trees play an important role in machine learning and data mining [11, 13–15], and classifiers based on decision trees work well for precise data. However, imprecision, uncertainty, and incompleteness prevail in real-world data. To deal with different kinds of uncertainty, a variety of modifications and generalizations of decision trees, such as the fuzzy decision tree [6] and the multi-valued decision tree [2], have been proposed.

As these variants of decision trees were proposed independently, there has not been a comparative study using a common framework. In this paper, we try to address this situation by proposing a bipolar interpretation of fuzzy decision trees. The interpretation makes it possible to represent various types of decision trees in a uniform framework. Our framework can deal with very general forms of fuzzy data and fuzzy rules. When specialized for specific

*Preliminary results of this paper were presented at the IEEE 2003 ICDM workshop on Foundations and New Directions in Data Mining and appeared in the informal proceedings of the workshop with limited distribution.

T.-F. Fan et al.: *A Bipolar Interpretation of Fuzzy Decision Trees*, Studies in Computational Intelligence (SCI) **118**, 109–123 (2008)
www.springerlink.com

instances, it provides interesting alternatives to the variety of decision trees proposed in the literature.

The rest of this paper is organized as follows. In Sects. 2 and 3, we present the data format and rule form of the classification problem respectively. In particular, we propose fuzzy data table for data representation, and use fuzzy decision logic as the rule representation language. In Sect. 4, we introduce a uniform framework, called *general fuzzy decision trees*. The edges of a general fuzzy decision tree are labeled by fuzzy decision logic formulas and the nodes are split according to the satisfaction of these formulas in the data records (or objects). We also present a construction algorithm for general fuzzy decision trees. In Sect. 5, we show the application of our framework to different types of training data by instantiating it to some specific cases. In particular, the bipolar interpretation of general fuzzy decision trees results in ordinary fuzzy decision trees [6] and multi-valued decision trees [2]. Finally, in Sect. 6, we briefly conclude this paper and indicate some further research directions.

2 Data Representation

A data table is normally used as means of storing data. A formal definition of a data table is given in [12].

Definition 1. *A data table*[1] *is a pair* $S = (U, A)$ *such that*

- $U = \{x_1, x_2, \cdots, x_n\}$ *is a nonempty finite set, called the universe*
- $A = \{f_1, f_2, \cdots, f_m\}$ *is a nonempty finite set of primitive attributes*
- *For* $1 \leq i \leq m$, $f_i : U \to V_i$ *is a total function, where* V_i *is the set of values for* f_i*, called the domain of values of* f_i*.*

To distinguish data tables from fuzzy data tables, we call them precise data tables. Hereafter, when we mention a data table $S = (U, A)$, we assume that the cardinalities of U and A are respectively n and m, f_i denotes the ith attribute in A, and V_i is its domain of values. Each element in U represents a data record. Since each data record describes the attributes of an object, we identify a data record with the object described by the data record. Thus, the elements of U are also called objects. In the following presentation, we treat the terms "data records" and "objects" interchangeably.

In a precise data table, it is assumed that $f_i(x)$ is exactly known for each object x and attribute f_i. However, in some practical situations, we have only incomplete information about $f_i(x)$ for some f_i and x. To accommodate such situations, incomplete information systems have been proposed [8–10, 16, 17]. Furthermore, many practical data mining problems need to deal with multi-valued data [2].

[1] Also called knowledge representation system, information system, or attribute-value system.

To represent incomplete information or multi-valued data, we can use *fuzzy data tables* (FDT). An FDT is a pair $S = (U, A)$, where U is a finite set of objects, $A = \{f_i : U \to \tilde{\mathcal{P}}(V_i) \mid 1 \leq i \leq m\}$, and $\tilde{\mathcal{P}}(V_i)$ denotes the class of fuzzy sets of domain V_i.

3 Rule Representation

In [12], decision logic (DL) is proposed as a means to represent knowledge discovered from data tables. This logic is called decision logic because it is particularly useful for a *decision table,* which is a data table $S = (U, A)$, where A can be partitioned into two sets, C (condition attributes) and D (decision attributes). Through data analysis, decision rules relating condition and decision attributes can be derived from the table. A rule is then represented as an implication between two formulas of DL.

Since DL can represent knowledge discovered from precise data tables, we generalize it to fuzzy decision logic (FDL) for rule representation in FDT. The basic alphabet of FDL consists of a finite set of attribute symbols $\mathcal{A} = \{a_1, a_2, \ldots, a_m\}$ and, for $1 \leq i \leq m$, a finite set of linguistic terms $\mathcal{L}_i$. The atomic formula of an FDL is a descriptor (a_i, l_i), where $a_i \in \mathcal{A}$ and $l_i \in \mathcal{L}_i$. The set of well-formed formulas (wff) of FDL is the smallest set containing the atomic formulas and closed under the Boolean connectives $\neg, \wedge$, and $\vee$. If φ and ψ are wffs of FDL, then $\varphi \longrightarrow \psi$ is a rule in FDL, where φ is the antecedent of the rule and ψ is the consequent.

Each element in the universe of an FDT corresponds to an object and an atomic formula (i.e., an attribute-value pair) describes the value of an individual attribute of an object. Thus, atomic formulas (and wffs) can be verified or falsified in each object. This gives rise to a satisfaction relation between the universe and the set of wffs.

Many natural language terms are highly context-dependent. For example, the word "tall" in "a tall basketball player" has a quite different meaning than it has in "a tall child". To model context-dependency, we associate a context with each FDL. The context determines the domain of values of each attribute and assigns an appropriate meaning to each linguistic term. Formally, given an FDT (U, A), a context associated with an FDL is a function, ct, that maps each linguistic term $l_i \in \mathcal{L}_i$ to $ct(l_i) \in \tilde{\mathcal{P}}(V_i)$ for $1 \leq i \leq m$, where V_i is the domain of values of attribute f_i. We assume each FDT has a fixed context.

Each linguistic term is interpreted as a fuzzy subset of attribute values, so an object may satisfy an atomic formula in FDL to some degree. Thus, the satisfaction between data records and wffs is a quantitative relation.

The semantics of FDL depend on how the fuzzy sets in the FDT and FDL contexts are interpreted. A fuzzy set can be interpreted disjunctively or conjunctively, and the difference between disjunctive and conjunctive interpretations corresponds to the bipolar representation of possibilistic logic [1].

3.1 Disjunctive Interpretation

In disjunctive interpretation, a fuzzy set is considered as constraints imposed by a linguistic term over the domain. Let V be the domain of possible values and X be a fuzzy set in $\tilde{\mathcal{P}}(V)$; then, the membership degree of an element, v, in X stipulates the possibility that the actual value is v. The disjunctive interpretation of fuzzy sets is appropriate for incomplete information systems. When we do not know the exact value of an attribute, we can encode the incomplete information by a fuzzy set. This is the interpretation adopted in the ordinary possibility theory by Zadeh [19].

Given a domain V, a *possibility distribution* on V is a function $\pi : V \to [0, 1]$. A possibility distribution π is called normalized if $\sup_{v \in V} \pi(v) = 1$. Two measures on V can be derived from π. They are called the possibility and the necessity measures and are denoted by Π and N respectively. Formally, Π and $N : 2^V \to [0, 1]$ are defined as

$$\Pi(A) = \sup_{v \in A} \pi(v),$$
$$N(A) = 1 - \Pi(\overline{A}),$$

where $\overline{A}$ is the complement of A with respect to V.

These two measures correspond to our uncertainty about the crisp event A when a piece of vague information π is available. They can be extended to measure the uncertainty of fuzzy events [4]. The extended measures, still denoted by Π and N, are defined as Π and $N : \tilde{\mathcal{P}}(V) \to [0, 1]$,

$$\Pi(X) = \sup_{v \in V} \mu_X(v) \otimes \pi(v),$$
$$N(X) = \inf_{v \in V} \pi(v) \to_{\otimes} \mu_X(v),$$

where μ_X is the membership function of a fuzzy event X, $\otimes : [0, 1] \times [0, 1] \to [0, 1]$ is a t-norm,[2] and $\to_{\otimes}$: $[0, 1] \times [0, 1] \to [0, 1]$ is the residuated implication function for $\otimes$ defined as $a \to_{\otimes} b = \sup\{x \mid x \otimes a \leq b\}$.

If an FDT $S = (U, A)$ represents an incomplete information system, for each $x \in U$ and $f_i \in A$, $f_i(x)$ is considered as a possibility distribution over V_i. Thus, its membership function is equivalent to a possibility distribution $\pi_{i,x}$ over the domain V_i. The wffs of FDL can then be evaluated in each data record of U according to the following evaluation function, E:

1. $E(x, (a_i, l_i)) = N_{i,x}(ct(l_i))$, where $N_{i,x}$ is the necessity measure corresponding to $\pi_{i,x}$
2. $E(x, \neg\varphi) = 1 - E(x, \varphi)$

[2] A binary operation $\otimes$ is a t-norm iff it is associative, commutative, and increasing in both places, and $1 \otimes a = a$ and $0 \otimes a = 0$ for all $a \in [0, 1]$.

3. $E(x, \varphi \wedge \psi) = E(x, \varphi) \otimes E(x, \psi)$
4. $E(x, \varphi \vee \psi) = E(x, \varphi) \oplus E(x, \psi)$

where $\oplus$ is a t-conorm defined by $a \oplus b = 1 - (1 - a) \otimes (1 - b)$.

In the disjunctive interpretation, $f_i(x)$ indicates the incomplete information about the value of the attribute f_i of the object x. To measure the extent to which x satisfies the atomic formula (a_i, l_i), we have to evaluate the truth degree of the statement "for every possible value v in the domain of attribute f_i, if x has the value v, then v satisfies the linguistic label l_i". In fuzzy logic, the truth degree of such a formula is defined by

$$\inf_{v \in V_i} \mu_{f_i(x)}(v) \rightarrow_{\otimes} \mu_{ct(l_i)}(v) = \inf_{v \in V_i} \pi_{i,x}(v) \rightarrow_{\otimes} \mu_{ct(l_i)}(v),$$

which is equal to $N_{i,x}(ct(l_i))$. In other words, $E(x, (a_i, l_i))$ is the necessity degree of the fuzzy event $ct(l_i)$ in accordance with the incomplete information $f_i(x)$. This explains the intuition behind the definition of the evaluation function E for the atomic formulas in the disjunctive case. The definition of the evaluation function for other wffs follows the standard approach in fuzzy logic.

3.2 Conjunctive Interpretation

In conjunctive interpretation, a fuzzy set represents positive knowledge in the domain, which is appropriate for the representation of multi-valued data. For example, in an FDT, the attribute "programming skills" may have a fuzzy set value $(\mathrm{C++}:1,\ \mathrm{Java}:0.8,\ \mathrm{Pascal}:0.6)$. By using conjunctive interpretation, the object with this attribute value has the programming ability of all three languages. If the fuzzy set were interpreted disjunctively, the object would only have the programming skills of one of these languages.

Mathematically, we can consider each linguistic label l_i as a possibility distribution over domain V_i. Thus, the membership function of $ct(l_i)$ is equivalent to a possibility distribution π_{l_i} and its corresponding necessity measure is denoted by N_{l_i}.

If an FDT $S = (U, A)$ represents multi-valued data, then the atomic wffs of FDL are evaluated in each $x \in U$ by

$$E(x, (a_i, l_i)) = N_{l_i}(f_i(x)),$$

and the evaluation function can be extended to any wffs in the same manner as in disjunctive interpretation.

In conjunctive interpretation, the fuzzy set $f_i(x)$ indicates all properties that x has with respect to its attribute f_i. Therefore, x satisfies the atomic formula (a_i, l_i) if it possesses all properties stipulated by the linguistic label l_i. In fuzzy logic, to evaluate if x possesses all properties stipulated by the linguistic label l_i, we have to evaluate the truth degree of the statement "for

every possible value v in the domain of attribute f_i, if v is stipulated by l_i, then v belongs to $f_i(x)$". The truth degree of such a formula is defined by

$$\inf_{v \in V_i} \mu_{ct(l_i)}(v) \rightarrow_{\otimes} \mu_{f_i(x)}(v),$$

which is equal to $N_{l_i}(f_i(x))$. This explains the intuition behind the definition of the evaluation function, E, for the atomic formulas in the conjunctive case.

4 General Fuzzy Decision Trees

In this paper, we assume that training data is stored in an FDT $S = (U, A)$, where A is partitioned into a set of condition attributes $A' = \{f_1, \ldots, f_{m-1}\}$ and the decision attribute f_m. In the terminology of the classification problem, any fuzzy subset of V_m is called a *class label*.

4.1 The Construction Phase

The construction algorithm for general fuzzy decision trees is presented in Fig. 1. It follows the standard framework of classical decision trees [13].

Below, we describe in detail the following three aspects of the algorithm:

1. How to label each node s with a fuzzy subset of V_m in the last step.
2. What is the stopping condition?
3. How to choose the best attribute in A_s for next split each time.

```
/*Initialization/

Let r denote the root of tree T
U_r ← U              /*The fuzzy subset of objects at the root is U/
A_r ← A'             /*The set of un-split attributes at the root is A'/

/*Main loop/

for each unlabeled leaf node s of T
    if the stopping condition is not satisfied by s and A_s ≠ ∅        /*See Sect. 4.1/
    then choose the best attribute f_i in A_s for the next split.      /*See Sect. 4.1/
        for each linguistic label l ∈ L_i
            Add a child node, s', to s and let the edge between s and s' be labeled (a_i, l)
            A_s' = A_s − {f_i}      /*Delete f_i from the set of un-split attributes/
            μ_{U_s'}(x) = μ_{U_s}(x) ⊗ E(x, (a_i, l))      /*Update the fuzzy subset of objects left at node s'/
        endfor
    endthen
    else label s with a fuzzy subset of V_m          /*See Sect. 4.1/
endfor
```

Fig. 1. The construction algorithm

Labeling the Nodes

The first step is to assign a class label to a leaf node of the tree. The class label of a node corresponds to a value of the decision attribute in the FDT, so it should be a fuzzy subset of V_m. The membership function of the class label for a node s, denoted by $\mu_s : V_m \to [0, 1]$, is determined by f_m and U_s, where U_s denotes the fuzzy subset of objects left at s.

There are several approaches for computing μ_s. First, by the qualitative approach:

$$\mu_s(v) = \min_{x \in U}(\mu_{U_s}(x) \to_{\otimes} \mu_{f_m(x)}(v)). \tag{1}$$

The value $\mu_s(v)$ indicates the degree of appropriateness of v to serve as a decision label for all objects left at s. Intuitively, a value v is appropriate for such a purpose if every object, x, left at s has the value v in its decision attribute value $f_m(x)$. This intuition is translated into (1) in fuzzy logic sense.

Second, an alternative approach is to use average supports:

$$\mu_s(v) = \sum_{x \in U} \frac{\mu_{U_s}(x)}{SC} \cdot \mu_{f_m(x)}(v), \tag{2}$$

where

$$SC = \sum_{x \in U} \mu_{U_s}(x)$$

is the sigma count [7] of objects appearing in s. In this equation, $\mu_{f_m(x)}(v)$ is considered as the support that x should be classified to the decision class v. Thus, (2) calculates the weighted sum of the supports that objects left at s should be classified to class v, where the weight of an object is determined by its proportion in the fuzzy subset U_s.

Third, another approach is to use normalized supports:

$$\mu_s(v) = \frac{\sum_{x \in U}(\mu_{U_s}(x) \cdot \mu_{f_m(x)}(v))}{M} \tag{3}$$

where

$$M = \max_{v \in V_m} \left(\sum_{x \in U} (\mu_{U_s}(x) \cdot \mu_{f_m(x)}(v)) \right)$$

is the maximum support among the values of V_m. The calculation of (3) is similar to that of (2). However, unlike in (2), the weights of the objects are not normalized in advance. Thus, the weight of each object is simply its degree of membership in the set U_s. To keep the final value $\mu_s(v)$ within [0,1], we normalize it by the maximal degree of support for all possible decision classes.

Finally, if a precise class label is required, then the standard center-of-gravity approach can be applied to defuzzify μ_s into a precise value in V_m.

The Stopping Condition

Regarding the stopping condition and the choice of the best attribute for the split, we must measure the diversity of decision values for objects appearing in a node. This can be achieved through global or local approaches. The global approach depends on how similar the decision value of an object is to the class label of the node. The class label of the node can be seen as the average of decision values for objects appearing in that node. Therefore, we must measure the similarity between two fuzzy subsets of V_m. There are many proposals for measuring the similarity of two fuzzy sets, so we do not specify what the similarity function is. We simply assume that sim is a similarity function mapping two membership functions, μ_1 and μ_2, into a number $sim(\mu_1, \mu_2) \in [0, 1]$. Several proposals of similarity functions are reviewed in the Appendix. Let $x, y \in U$ be objects and s be a node in the tree T. We also write $sim(x, s)$ for $sim(\mu_{f_m(x)}, \mu_s)$ and $sim(x, y)$ for $sim(\mu_{f_m(x)}, \mu_{f_m(y)})$.

The diversity of decision values for objects appearing in a node, s, can be measured by aggregating $sim(x, s)$ for all x left at s. The aggregated result is called the *global degree of concentration,* and we denote the global degree of concentration of a node s by gdc_s. There are at least two ways to define gdc_s. First, by qualitative means:

$$gdc_s = \min_{x \in U}(\mu_{U_s}(x) \rightarrow_{\otimes} sim(x, s)), \tag{4}$$

and, second, by quantitative means,

$$gdc_s = \sum_{x \in U} \frac{\mu_{U_s}(x)}{SC} \cdot sim(x, s), \tag{5}$$

where SC is the sigma count of U_s as defined above. A smaller gdc_s value indicates a more diverse decision value of objects appearing in s. The qualitative gdc_s measures the degree of truth of the statement "the decision class of every object left at s is similar to the label of s" in a fuzzy logic sense. The quantitative gdc_s measures the average similarity of the decision class of every object left at s to the label of s. Note that to calculate the global degree of concentration, we have to assign labels not only to leaf nodes, but also to internal nodes. The label assignment procedure for internal nodes is exactly the same as that introduced in Sect. 4.1.

As suggested in [2], we can also measure the diversity of decision values of objects appearing in a node, s, by using the (average) mutual similarity between the decision values. This is called the *local degree of concentration*, denoted by ldc_s, and can also be defined in two ways, i.e., by qualitative means:

$$ldc_s = \min_{1 \le i < j \le n}(\mu_{U_s}(x_i) \otimes \mu_{U_s}(x_j) \rightarrow_{\otimes} sim(x_i, x_j)); \tag{6}$$

or by quantitative means:

$$ldc_s = \sum_{1 \le i < j \le n} \frac{\mu_{U_s}(x_i) \cdot \mu_{U_s}(x_i) \cdot sim(x_i, x_j)}{\sum_{1 \le i < j \le n} \mu_{U_s}(x_i) \cdot \mu_{U_s}(x_i)} \tag{7}$$

recalling that $U = \{x_1, \cdots, x_n\}$. The qualitative ldc_s measures the degree of truth of the statement "decision classes of all objects left at s are pairwisely similar" in a fuzzy logic sense, whereas the quantitative ldc_s measures the average similarity of the decision values. Unlike in the global case, we do not have to assign decision labels to the internal nodes of the decision tree. However, in the local case, the similarity of decision classes of each pair of objects has to be calculated.

If we let dc_s denote either gdc_s or ldc_s, then the stopping condition is satisfied by a node s if $dc_s \ge \theta$ for some preset threshold value θ. The choice of an appropriate definition of dc_s depends on the problem context and the complexity consideration.

Choice of the Best Attribute

To choose the best attribute for split each time, we simply split a node s with each attribute in A_s to find which attribute results in the maximum average concentration degree. Let f_i be an attribute in A_s and for each $l \in \mathcal{L}_i$, s^l be the child node of s corresponding to (a_i, l), provided that s is split with attribute f_i. We denote the sigma count of U_{s^l} as SC_{s^l} for any $l \in \mathcal{L}_i$. The resultant average degree of concentration after splitting s with attribute f_i is

$$\overline{dc}_{i,s} = \sum_{l \in \mathcal{L}_i} \frac{SC_{s^l}}{\sum_{l \in \mathcal{L}_i} SC_{s^l}} \cdot dc_{s^l}. \tag{8}$$

Therefore, the attribute chosen for next split should be the attribute $f_i \in A_s$ that maximizes $\overline{dc}_{i,s}$, i.e.,

$$\arg \max_{f_i \in A_s} \overline{dc}_{i,s}.$$

4.2 The Decision Phase

Once a general fuzzy decision tree has been constructed, we can use it to classify new data. Let x be an object with attributes $f_1, \ldots, f_{m-1}$ such that $f_i(x) \in \tilde{\mathcal{P}}(V_i)$ for $1 \le i \le m-1$. Since labels on the edges of the decision tree are atomic formulas based on attributes $f_1, \ldots, f_{m-1}$, the evaluation function E can be applied to the object x and the atomic formulas for assignment of a decision value $f_m(x)$ to x.

We associate an FDL wff, φ_s, with each leaf node s of the decision tree. The wff φ_s is the conjunction of all atomic formulas appearing on the edges

of the path from the root to s. The decision value $f_m(x)$ is then defined by the membership function

$$\mu_{f_m(x)} = \bigoplus_{s \in LN} E(x, \varphi_s) \otimes \mu_s, \tag{9}$$

where LN is the set of all leaf nodes, and the t-norm and t-conorm operations are extended to membership functions pointwisely.

The decision tree can be used as a set of decision rules. Each leaf node s is roughly equivalent to a decision rule whose antecedent is φ_s and consequent is μ_s. Strictly speaking, μ_s is a membership function, instead of a wff of FDL, so it is not really a decision rule. However, we can consider μ_s as the membership function of a fuzzy subset corresponding to a linguistic term in domain V_m. Therefore, the definition of $f_m(x)$ results from the approximate reasoning scheme in fuzzy logic [19]. This is also the approach to decision inference in fuzzy decision trees in [6]. However, if a precise decision is required, then the standard center-of-gravity approach can be employed to defuzzify $f_m(x)$ into a precise value in V_m.

5 Variants of Decision Trees

General fuzzy decision trees provide a common framework for expressing different types od decision trees. An instance of general fuzzy decision trees is characterized by the following parameters:

1. data format in the FDT,
2. rule form in the FDL,
3. bipolar interpretation of data and wffs (disjunctive or conjunctive),
4. assignment of class labels to decision tree nodes,
5. computation of degrees of concentration.

In this section, we consider instances related to the classical decision tree [13], fuzzy decision tree [6], and multi-valued decision tree [2].

5.1 Classical Decision Trees Revisited

One classical instance of general fuzzy decision trees is characterized by the following parameters:

1. data format in the FDT: $f_i(x)$ is a singleton subset of V_i for all $f_i \in A$ and $x \in U$.
2. rule form: in the FDL, we restrict $\mathcal{L}_i = V_i$ for all $f_i \in A$.
3. interpretation of wffs: with the restrictions on FDT and FDL, conjunctive and disjunctive interpretations collapse. We choose the t-norm $\otimes = \min$, so $\otimes, \oplus$, and $\rightarrow_\otimes$ correspond respectively to the classical Boolean operations $\wedge, \vee$, and $\rightarrow$. Therefore, $E(x, \varphi) \in \{0, 1\}$ holds for each $x \in U$ and

wff φ. In particular, $E(x,(a_i,v))=1$ iff $f_i(x)=v$. Consequently, U_s is a crisp subset of U for each node s of the decision tree.

4. assignment of class labels to decision tree nodes: we use the average support. Let $|U_s|=n_s$ and $V_m=\{v_1,\cdots,v_k\}$ and assume that the number of objects in U_s with decision value v_i is n_i. Then, according to (2), the class label of s is a fuzzy subset of V_m with the membership function

$$\mu_s(v_i)=\frac{n_i}{n_s}\triangleq p_i,\ \ 1\le i\le k.$$

   Note that $\sum_{i=1}^{k}p_i=1$ holds in this case.
5. computation of degrees of concentration: we use the *sim* defined in (10) and compute the *gdc* according to (5), then

$$gdc_s=\sum_{i=1}^{k}p_i\cdot\frac{p_i}{1+\sum_{j\neq i}p_j}=\sum_{i=1}^{k}\frac{p_i^2}{2-p_i}.$$

This kind of classical decision tree uses different stopping conditions and selection criteria than those based on information gains derived from entropy [13] or the Gini index [11].

5.2 Multi-Valued Decision Trees

An instance of multi-valued decision trees is characterized as follows:

1. data format in the FDT: $f_i(x)$ is a crisp non-empty subset of V_i for all $f_i\in A$ and $x\in U$.
2. rule form in the FDL: we restrict $\mathcal{L}_i=V_i$ for all $f_i\in A$.
3. interpretation of wffs: we use conjunctive interpretation, and still choose the t-norm $\otimes=\min$. Again, $E(x,\varphi)\in\{0,1\}$ holds for each $x\in U$ and wff φ. In particular, $E(x,(a_i,v))=1$ iff $v\in f_i(x)$. Therefore, U_s is also a crisp subset of U for each node s of the decision tree.
4. assignment of class labels to decision tree nodes: we use average support. Let $|U_s|=n_s$ and $V_m=\{v_1,\cdots,v_k\}$ and assume that the number of objects in U_s whose decision values contain v_i is n_i. Then, according to (2), the class label of s is a fuzzy subset of V_m with the membership function

$$\mu_s(v_i)=\frac{n_i}{n_s}\triangleq p_i,\ \ 1\le i\le k.$$

   Note that $\sum_{i=1}^{k}p_i=1$ no longer holds in this case.
5. computation of degrees of concentration: we use the *sim* defined in (10) and compute the *ldc* according to (7). Then ldc_s is equal to the *set-similarity* function defined in [2].

This kind of multi-valued decision tree is very similar to that in [2]. There are, however, two subtle differences. One is the assignment of class labels. In

our approach, we assign a fuzzy subset of V_m as the class label of a node, whereas in [2] this subset is further defuzzified into a crisp subset of V_m. The other difference is the stopping condition. In our approach, the stopping condition is based on ldc_s, whereas in [2] a criterion based purely on μ_s is given. In [2], with a user-specified parameter σ, the set V_m is partitioned into $large_s$ and $small_s$ in a node s, where $large_s = \{v \in V_m \mid \mu_s(v) \geq \sigma\}$ and $small_s = V_m - large_s$. A node s is called clear if $\min\{\mu_s(v) \mid v \in large_s\} - \max\{\mu_s(v) \mid v \in small_s\} > \delta$, where δ is again a user-specified parameter. Roughly speaking, if a node s is clear, then no further expansion is needed and the class label assigned to the node is $large_s$.

5.3 Fuzzy Decision Trees

For an instance of fuzzy decision trees, we use the following parameter setting:

1. data format in the FDT: $f_i(x)$ is a singleton subset of V_i for all $f_i \in A$ and $x \in U$.
2. rule form in the FDL: we only restrict that $\mathcal{L}_i$ is finite for all $f_i \in A$.
3. interpretation of wffs: we use disjunctive interpretation, and still choose the t-norm $\otimes = \min$. However, $E(x, (a_i, l)) = \mu_{ct(l)}(f_i(x))$ is now a real number in $[0, 1]$. Therefore, U_s is a fuzzy subset of U for each node s of the decision tree.
4. assignment of class labels to decision tree nodes: we use average support, and still assume $V_m = \{v_1, \ldots, v_k\}$. Let SC denote the sigma count of U_s and r_i denote $\sum_{x:f_m(x)=v_i} \mu_{U_s}(x)$ for $1 \leq i \leq k$. Then, the class label of s is a fuzzy subset of V_m with the membership function

$$\mu_s(v_i) = \frac{r_i}{SC} \triangleq p_i, \quad 1 \leq i \leq k.$$

   Note that $\sum_{i=1}^{k} p_i = 1$ holds in this case.
5. computation of degrees of concentration: we use the sim defined in (10) and compute the gdc according to (5). Then, analogous to the case of classical decision trees,

$$gdc_s = \sum_{i=1}^{k} \frac{p_i^2}{2 - p_i}.$$

6 Conclusion

Decision tree approach is important since decision trees provide solutions to classification problems and extract rules effectively. To deal with different kinds of data, decision trees have been generalized along different directions in the past. In this paper, we propose a quite general framework for fuzzy decision trees. Some particular instances of this framework prove to be interesting

alternatives to previous proposals. Detailed comparison of our approach with these related works is ongoing. Implementation and experimental testing of our approach will be covered by future research.

A Definitions of Similarity Functions

Here, we review some basic definitions of the similarities between fuzzy sets over finite domains [3]. Given two membership functions, μ_1 and μ_2, for fuzzy subsets F_1 and F_2, respectively, of a finite domain V, $sim(\mu_1, \mu_2)$ can be defined in several ways.

One way is based on the cardinality of the intersection and union of two sets. This is a generalization of the Jaccard index (also called the coefficient of similarity or index of commonality) between crisp sets [5]. For the crisp case, the Jaccard index between two crisp sets F and G is defined as $\frac{|F \cap G|}{|F \cup G|}$, where $|\cdot|$ denotes the cardinality of a set. Analogously, for fuzzy sets,

$$sim(\mu_1, \mu_2) = \frac{\sum_{v \in V} \mu_1(v) \otimes \mu_2(v)}{\sum_{v \in V} \mu_1(v) \oplus \mu_2(v)} = \frac{|F_1 \cap F_2|}{|F_1 \cup F_2|}, \tag{10}$$

where $|\cdot|$ denotes the sigma count of a fuzzy set.

Another definition of similarity based on cardinality is called the simple matching coefficient [18], defined as

$$\frac{|F_1 \cap F_2| + |\overline{F_1} \cap \overline{F_2}|}{|V|}, \tag{11}$$

where $\overline{F_i}$ is the complement set of F_i in V, for $i = 1, 2$.

Yet another definition measures the degree of mutual inclusion of two fuzzy sets.

$$sim(\mu_1, \mu_2) = \frac{\sum_{v \in V} \mu_1(v) \leftrightarrow_{\otimes} \mu_2(v)}{|V|}, \tag{12}$$

where $a \leftrightarrow_{\otimes} b = (a \rightarrow_{\otimes} b) \otimes (b \rightarrow_{\otimes} a)$ is the equivalence function with respect to a t-norm $\otimes$.

As $|V| = k$, μ_1 and μ_2 can be seen as two k-dimensional vectors, then the similarity can be measured with the cosine of the angle between two vectors:

$$sim(\mu_1, \mu_2) = \frac{\mu_1 \circ \mu_2}{\|\mu_1\| \cdot \|\mu_2\|}, \tag{13}$$

where $\mu_1 \circ \mu_2$ is the inner product of μ_1 and μ_2, and $\|\mu_i\|$ is the length of μ_i, for $i = 1, 2$.

Other approaches based on the Hamming distance, Euclidean distance, or more general Minkowski distance metrics are also possible. The Minkowski distance of the pth order between μ_1 and μ_2 is defined as

$$dis_p(\mu_1, \mu_2) = \left(\sum_{v \in V} (|\mu_1(v) - \mu_2(v)|)^p \right)^{\frac{1}{p}}.$$

The similarity based on this distance metric is then defined as

$$sim(\mu_1, \mu_2) = 1 - \frac{dis_p(\mu_1, \mu_2)}{|V|^{\frac{1}{p}}}. \tag{14}$$

References

1. S. Benferhat, D. Dubois, S. Kaci, and H. Prade. Bipolar possibilistic representations. In *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence*, pages 45–52, Morgan Kaufmann, San Francisco, CA, 2002.
2. Y.L. Chen, C.L. Hsu, and S.C. Chou. Constructing a multi-valued and multi-labeled decision tree. *Expert Systems with Applications*, 25:199–209, 2003.
3. V.V. Croos and T.A. Sudkamp. *Similarity and Compatibility in Fuzzy Set Theory.* Physica-Verlag, Wurzburg (Wien), 2002.
4. D. Dubois and H. Prade. An introduction to possibilistic and fuzzy logics. In P. Smets, A. Mamdani, D. Dubois, and H. Prade, editors, *Non-Standard Logics for Automated Reasoning*, pages 253–286. Academic Press, 1988.
5. P. Jaccard. Nouvelles recherches sur la distribution florale. *Bulletin de la Societe de Vaud des Sciences Naturelles*, 44:223, 1908.
6. C.Z. Janikow. Fuzzy decision trees: Issues and methods. *IEEE Transcations on Systems, Man, and Cybernetics*, 28(1):1–14, 1998.
7. A. Kandel. *Fuzzy Mathematical Techniques with Applications.* Addison-Wesley, 1986.
8. M. Kryszkiewicz. Properties of incomplete information systems in the framework of rough sets. In L. Polkowski and A. Skowron, editors, *Rough Sets in Knowledge Discovery*, pages 422–450. Physica-Verlag, 1998.
9. M. Kryszkiewicz and H. Rybiński. Reducing information systems with uncertain attributes. In Z. W. Raś and M. Michalewicz, editors, *Proceedings of the 9th ISMIS*, LNAI 1079, pages 285–294. Springer, Berlin Heidelberg New York, 1996.
10. M. Kryszkiewicz and H. Rybiński. Reducing information systems with uncertain real value attributes. In *Proceedings of the 6th International Conference on Information Processing and Management of Uncertainty in Knowledge-based Systems*, pages 1165–1169, 1996.
11. M. Mehta, R. Agrawal, and J. Rissanen. Sliq: A fast scalable classifier for data mining. In *Proceedings of the 5th International Conference on Extending Database Technology*, pages 18–32, Avigon, France, 1996.
12. Z. Pawlak. *Rough Sets–Theoretical Aspects of Reasoning about Data.* Kluwer, 1991.
13. J.R. Quinlan. Induction on decision trees. *Machine Learning*, 1:81–106, 1986.

14. J.R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Francisco, CA, 1993.
15. R. Rastogi and K. Shim. Public: A decision tree classifier that integrates building and pruning. *Data Mining and Knowledge Discovery*, 1(4):315–344, 2000.
16. R. Słowiński and J. Stefanowski. Rough-set reasoning about uncertain data. Technical Report ICS-26/94, Warsaw University of Technology, Warsaw, Poland, 1994.
17. R. Słowiński and J. Stefanowski. Rough-set reasoning about uncertain data. *Fundamenta Informaticae*, 27(2–3):229–243, 1996.
18. R.R. Sokal and P.H. Sneath. *Principles of Numerical Taxonomy*. Freeman, San Francisco, CA, 1963.
19. L.A. Zadeh. Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems*, 1(1):3–28, 1978.

A Probability Theory Perspective on the Zadeh Fuzzy System

Qing Shi Gao[1], Xiao Yu Gao[1], and Lei Xu[2]

[1] University of Science and Technology, Beijing, 100083
[2] Chinese University of Hong Kong, Shatin, N.T., Hong Kong

Summary. Probability theory (P-theory) and Zadeh fuzzy system (Z-system) have both been used as foundations for mining structures or relations among data. The goal of this paper is to study the links and differences between the two systems. We start by considering the Z^--system, derived by discarding the complement set definition in the Z-system. A theorem is proved to state that P-theory and the Z^--system perform equivalently when one set is a subset of the other (i.e., $A \subseteq B$ or $B \subseteq A$) either in a sense of almost surely or in a sense of the Zadeh fuzzy set. Furthermore by jointly considering the B-system that modifies the Z-system with the "MIN–MAX" operations replaced by so-called *Bold* operations, another theorem is proved to state that the Z^--system and the B-system attempt to approximate the P-theory in two opposite ways, with success in some cases and failure in others. The failures come from either or both of an incapability of capturing additive structures and an inadequate handling of the dependence relation across two sets. Finally, we examine the controversial definition of a complement set in the Z-system and clarify that it arises from confusion about the "complement" concept and the "conjugate" concept. The confusion and thus the controversy can be avoided by honoring the additive principle, as in the B-system, or by renaming the complement set as the conjugate set.

1 Introduction

Probability theory (P-theory) and the Zadeh fuzzy system (Z-system) are two most widely explored tools for implementing computer reasoning in various fields, and each has got a large population of supporters. In recent years, efforts have also been made in the data mining field on adopting each of two as a foundation for mining dependence or relations among data.

The key feature of the Z-system is that it is easy to understand and simple to implement; thus, it attracted a lot of practitioners, especially in some control system applications in 1970s and 1980s. Even so, its roughness in nature has generated a great deal of controversy in the literatures [3, 8].

In a classic sense, a probability is interpreted as the frequency or chance that an event will occurs. This is often used in the fuzzy literature as an

Q.S. Gao et al.: *A Probability Theory Perspective on the Zadeh Fuzzy System*, Studies in Computational Intelligence (SCI) **118**, 125–137 (2008)
www.springerlink.com

evidence to argue how the Z-system is capable to handle a reasoning task that is very different from tasks by Probability theory (P-theory) [2]. However, this classic interpretation is not intrinsic to the P-theory. Starting with Bayes more than 200 years ago, non-frequency interpretation has also been often adopted in the literature of so-called Bayesian formalism. For further details, readers are referred to a well written overview in 1988 Pearl's famous book [7]. Some preliminary investigations were also been made in [9]. In the past two decades, studies of P-theory have increased dramatically in the AI field under the name of Bayesian networks, Belief networks, or graphical probabilistic model [6, 7], resulting in effective algorithms and a large number of practical successes.

There has been a long lasting argument between those "pro-probabilistic" people, who simply refuse to accept the Z-system with certain type of ignorance, and those "pro-fuzzy" peoples, who worship the Z-system with another type of ignorance. This paper aims at studying their links and differences. In Sect. 2, we introduce certain fundamentals of set theory, the Z-system and P-theory, through which we show that probability is not only a model of uncertainty, as argued in the fuzzy literature, but also applicable to partial membership to handle the same tasks handled by the Z-system. In Sect. 3, we present a detailed analysis and two theorems to elucidate where the Z-system works and fails. In Sect. 4, we examine the controversy over a definition of the complement set in the Z-system. Finally, some concluding remarks are given in Sect. 5.

2 Set Theory, Fuzzy Set System, and Probability Theory

2.1 Set Theory

A set $A \subseteq U$ describes a collection of elements $a, b, c, \cdots$ in a universe U, ranging from one extreme case $A = \emptyset$, where A is empty, to the other extreme where $A = U$. Given two sets $A \subseteq U$ and $B \subseteq U$, there are three types of mutual relation:

(a) One is a subset of the other, i.e.,

$$\begin{aligned} &\textit{either } A \subseteq B, \textit{ i.e., we have } x \in B \textit{ if } x \in A, \\ &\textit{or } B \subseteq A, \textit{ i.e., we have } x \in A \textit{ if } x \in B. \end{aligned} \tag{1}$$

(b) A and B are disjoint, i.e., they do not share a common element.
(c) A and B share some common elements, but neither $A \subseteq B$ nor $B \subseteq A$.

The following two basis operations are introduced to describe the above scenarios and the corresponding new concepts:

$$A \cap B = \{x : \ x \in B \textit{ and } x \in A\}, \quad A \cup B = \{x : \ x \in B \textit{ or } x \in A\}. \tag{2}$$

The intersect $A \cap B$ describes the common part of A and B, while the union $A \cup B$ describes the total coverage of A and B. For case (b), we have $A \cap B = \emptyset$, while for case (c) we have $A \cap B \neq \emptyset$, where neither $A \subseteq B$ nor $B \subseteq A$.

Based on the concepts of $\emptyset, U, \cup, \cap$ jointly, for any set A we can uniquely find a set X that satisfies

$$(a)\ A \cup X = U, \quad (b)\ A \cap X = \emptyset. \tag{3}$$

Such a unique X is called the complement set of A, denoted by $\neg A$. We can write it equivalently as follows

$$\neg A = \{x \mid every\ x\ in\ U\ but\ not\ in\ A\}. \tag{4}$$

Alternatively, a set A can also be represented by the following characteristic function :

$$\mu_A(a) = \begin{cases} 1, & a \in A, \\ 0, & otherwise, \end{cases} \tag{5}$$

which is a binary valued function that characterizes the set A.

It further follows that the subset $A \subseteq B$ means "$\mu_A(a) = 0 = \mu_B(a)$ for every a outside B, $\mu_A(a) < \mu_B(a)$ for every a in B but not in A, and $\mu_A(a) = 1 = \mu_B(a)$ for every a in both A and B". In a compact representation, the subset concept by (1) can be expressed as follows

$$either\ \mu_A(a) \leq \mu_B(a), \forall a \in U,\ or\ \mu_B(a) \leq \mu_A(a), \forall a \in U. \tag{6}$$

Also, the operations $A \cap B$ and $A \cup B$ in (2) can be expressed as

$$\begin{aligned} &(a)\ \mu_{A \cup B}(a) = \max\{\mu_A(a), \mu_B(a)\}, \\ &(b)\ \mu_{A \cap B}(a) = \min\{\mu_A(a), \mu_B(a)\},\ respectively \end{aligned} \tag{7}$$

Moreover, the complement set by (3) can be expressed by $\mu_{\neg A}(a)$, which is the unique solution of

$$(a)\ 1 = \max\{\mu_A(a), \mu_{\neg A}(a)\}, \quad (b)\ 0 = \min\{\mu_A(a), \mu_{\neg A}(a)\}. \tag{8}$$

In term of (5), it can also be explicitly expressed as

$$\mu_{\neg A}(a) = 1 - \mu_A(a). \tag{9}$$

2.2 Fuzzy Set System

The Zadeh Fuzzy set system (Z-system) is a straightforward extension of the above set theory with (7) and (9) being extended from the binary valued membership by (5) to the following real value membership:

$$0 \leq \mu_A(a) \leq 1. \tag{10}$$

That is, a belongs to A as a partial membership or so-called fuzzy membership. Under this extension, the subset concept of (6) and many other natures owned under the binary valued membership of (5) remain true. The "MIN–MAX" operations are simple in implementation and easy to understand, which makes the Z-system adopted widely in past decades, especially in control systems [8].

However, use of the Z-system is very controversial due to many reported failures and certain conflicts in the system. One example is the rejection of the excluded middle law $A \cup \neg A$. That is, we get $\mu_{\neg A}(a) = 0.5$ from (9) when $\mu_A(a) = 0.5$. It follows from (9) that $\mu_{A\cup\neg A}(a) = 0.5 = \mu_{A\cap\neg A}(a)$, which is difficult to be accepted in a traditional logic sense. For example, it has been shown that requiring $\mu_{A\cup\neg A}(a) = 1$ and certain known logical equivalences will cause the system consisting of (7), (9) and (10) collapse into a binary valued logic [3, 8].

Some "pro-fuzzy" people do not see these conflicts as problems, arguing that they are inevitable and understandable due to missing information. Some even insist that this is the main attraction of the Z system, while others attempt to resolve the controversy by modifying (7)as follows [1, 5]:

$$\begin{aligned} &(a)\ \mu_{A\cup B}(a) = \min\{1, \mu_A(a) + \mu_B(a)\}, \\ &(b)\ \mu_{A\cap B}(a) = \max\{0, \mu_A(a) + \mu_B(a) - 1\}. \end{aligned} \tag{11}$$

These are called *bold* operations (or the B-system). It is easy to see that $\mu_{A\cup\neg A}(a) = 1$ and $\mu_{A\cap\neg A}(a) = 0$. That is, the above-mentioned conflicts are resolved.

2.3 Probability Theory

In the Bayesian formalism, Probability theory (P-theory) consists of the following three basic axioms (see (2.1)–(2.3) in [7]):

$$\begin{aligned} &\text{(a) } 0 \le P(A) \le 1, \\ &\text{(b) } P(T) = 1, \\ &\text{(c) } P(A \vee B) = P(A) + P(B), \text{ if } A, B \text{ are mutually exclusive,} \end{aligned} \tag{12}$$

where A, B denote propositions and T denotes the true proposition.

Set theory and the Z-system are both expressed in terms of the relation between an element a and a set A. To facilitate a comparison, whether a proposition A is true is written as whether $a \in A$ is true, and the true proposition T is written as $a \in U$. Thus, it follows from (12a) that $P(a \in A) = P_A(a)$ with

$$0 \le P_A(a) \le 1 \tag{13}$$

in a same format as (10). In terms of classic probability theory, we can interpret $P_A(a)$ as the probability that a belongs to a set A, that is, a probabilistic interpretation of the membership in set theory. For the subset by (1), we can also have

$$either\ P_A(a) \le P_B(a), \forall a \in U,\ or\ P_A(a) \le P_B(a), \forall a \in U. \tag{14}$$

Limited to the above probabilistic interpretation of $P_A(a)$, it is argued in the fuzzy literature that probability is a model of uncertainty, whereas fuzzy set is for partial membership. In other words, $\mu_A(a)$ in (10) describes a partial membership that a belongs to A, whereas $P_A(a)$ describes an actual or believed degree of uncertainty on the fact "a is in A". However, this is only one type of interpretations, which is intrinsic to neither the algebraic system by (5), (7) and (9) or the algebraic system by (12).

Forgetting additional interpretations, (10) and (13) have no difference on describing the relation between a and A. The difference between the P-theory and the Z-system lays in the differences between the operations by (7) and (9) and the operation by (12b) and (12c). We can also explain (13) from the perspective that A is regarded as a "soft" or "fuzzy" set with $P_A(a)$ as a membership function. Also, (14) can be explained from the same perspective as (6). In other words, $P_A(a)$ can be interpreted from a perspective that can handle a partial truth or a multi-valued truth in the same format as (10).

Compared to the MIN–MAX operations in (8), the operations by $P(A \vee B) = P(a \in A \vee b \in B)$ and $P(A \wedge B) = P(a \in A \wedge b \in B)$ are not only more general in their coverage but also more accurate in describing the underlying relations.

First, it follows from $P(A \vee B) = P(a \in A \vee b \in B) = P(a \in A \vee a \in B)P(a = b)$ that we reduce our coverage to consider special cases with $A \subseteq U$ and $B \subseteq U$ as well as $a = b$. We have $P(a \in A \vee a \in B) = P(a \in A \cup B) = P_{A\cup B}(a)$ and $P(a \in A \wedge a \in B) = P(a \in A \cap B) = P_{A\cap B}(a)$, from which we get the counterpart of (7) as follows:

$$\begin{aligned} &(a)\ P_{A\cup B}(a) = P_A(a) + P_B(a) - P_{A\cap B}(a), \\ &(b)\ P_{A\cap B}(a) = P_A(a)P_{B|A}(a)\ or\ P_{A\cap B}(a) = P_B(a)P_{A|B}(a), \end{aligned} \tag{15}$$

which leads to (12c) when $P_{A\cap B}(a) = 0$. On the other hand, (15) can also be obtained from (12a) and (12c).

Second, we compare (15) with (7) and find out where the P-theory differs from the Z-system. For a union connection, the composition by (15) is made additively with the effect from the corresponding intersect connection in consideration, while the composition by (7) is made non-additively without considering the intersect connection. To get the intersect connection, (7) simply takes the shared portion of the membership, while (15) computes a Bayesian product in help of the following concept that does not exist in the Z-system:

$$P_{B|A}(a) = P_{A\cap B}(a)/P_A(a),\ P_{A|B}(a) = P_{A\cap B}(a)/P_B(a). \tag{16}$$

In terms of classic probability theory, (16) represents the conditional probability, e.g., $P_{A|B}(a)$ for the probability that $a \in A$ under the condition $x \in B$. However, $P_{A|B}(a)$ can also be interpreted as the ratio of the partial truth

where $a \in A \cap B$ over the partial truth where a more relaxed proposition $a \in A$. Alternatively, it can also be explained as the believed truth on $a \in A \cap B$ under the assumption that $a \in A$ is completely true.

Furthermore, it follows from (12b) that $P(a \in U) = P_U(a) = 1$. Similar to (8), the extension of the complement set by (3) can be defined by the unique solution of

$$(a)\ 1 = P_{A \cup \neg A}(a) = P_A(a) + P_{\neg A}(a) - P_{A \cap \neg A}(a),\ (b)\ 0 = P_{A \cap \neg A}(a),$$

which leads to the following definition for the complement set $\neg A$:

$$P_{\neg A}(a) = 1 - P_A(a),\ P_{A \cup \neg A}(a) = 0. \tag{17}$$

3 Cases Where the Z-System Works or Fails

3.1 Cases Where a Reduced Z-System Works Well

We consider a reduced Z-system (denoted by Z^--system) by ignoring its definition of the complement set.

Theorem 1. *When either $P_{B|A}(a) = 1$ or $P_{A|B}(a) = 1$, we have*

$$\begin{array}{l}(a)\ P_{A \cup B}(a) = \max\{P_A(a), P_B(a)\}, \\ (b)\ P_{A \cap B}(a) = \min\{P_A(a), P_B(a)\}.\end{array} \tag{18}$$

Proof. When $P_{B|A}(a) = 1$, it follows from (15b) that we have $P_{A \cap B}(a) = P_A(a) \leq P_B(a)$ or equivalently $P_{A \cap B}(a) = \min\{P_A(a), P_B(a)\}$. It further follows from (15a) that $P_{A \cup B}(a) = P_B(a) + P_A(a) - P_A(a) = P_B(a) = \max\{P_A(a), P_B(a)\}$. When $P_{A|B}(a) = 1$, we can also prove similarly. □

Note that (18) is same as (7). That is, when we have $A \subseteq B$ almost surely (i.e., $1 = P_{B|A}(a) = P(a \in B | a \in A) = P(A \subseteq B)$), or when we have $B \subseteq A$ almost surely, reasoning by the Z^--system in these cases is equivalent to that by the P-theory, if the complement operator in (9) is not involved. The following lemma states that this equivalence is also true when $A \subseteq B$ or $B \subseteq A$ even in a sense of the Zadeh fuzzy set (i.e., in terms of (6) or (14)).

Lemma 1. *(1) $P_{B|A}(a) = 1$ if $A \subseteq B$*
(2) $P_{B|A}(a) = 1$, $P_{A \cap B}(a) = P_A(a)$, and $P_A(a) \leq P_B(a), \forall a \in U$ for are equivalent

Proof. (1) It is straightforward.
(2) It follows from (16) that $P_{A \cap B}(a) = P_A(a)$ from $P_{B|A}(a) = 1$. Moreover, it follows from $P_{A \cap B}(a) = P_A(a)$ and (15a) that $P_{A \cup B}(a) = P_B(a)$. Thus $P_A(a) = P_{A \cap B}(a) \leq P_{A \cup B}(a) = P_B(a), \forall a \in U$. Furthermore, it follows from (15)(b) and $P_{B|A}(a) \leq 1$ that $P_{A \cap B}(a) \leq P_A(a)$,

$\forall a \in U$, and that $P_{B\cap A}(a) \leq P_B(a), \forall a \in U$. Thus, $P_{A\cap B}(a) = min\{P_A(a), P_B(a)\}$. It further follows from (16) that $P_{B|A}(a) = P_{A\cap B}(a)/P_A(a) = min\{P_A(a), P_B(a)\}/P_A$. Since $P_A(a) \leq P_B(a)$, we have $min\{P_A(a), P_B(a)\} = P_A(a)$ and $P_{B|A}(a) = 1$. □

It is straightforward that the above Lemma still holds by exchanging A and B.

The Z^--system may still work approximately even when the condition of Theorem 1 does not hold exactly. This analysis explains some successful applications of the Z-system, e.g., those in control systems.

3.2 Examples: Z-System and B-System Fail to Work Well

In addition to the above cases, the "MIN–MAX" operations may not yield reasonable results. Let us to observe the following examples.

Example 1. Consider a set for youth B and a set for juvenile A. It is partially true to regard a person $a = 17$-year-old who is 17 years old as a youth (e.g., a degree of 0.7), but it is also partially true to regard this person as a juvenile (e.g., a degree 0.4). On the other hand, based on common sense, we can be 100% sure that this person belongs to the union set C of youths and juveniles. However, according to the Z^--system by (7), we have $\mu_B(17) = 0.7$, $\mu_A(17) = 0.4$, and $\mu_C(17) = \max\{0.7, 0.4\} = 0.7$. In a contrast, according to the P-theory, by (15) we have $P_B(17) = 0.7$, $P_A(17) = 0.4$, and $P_C(17) = 0.7 + 0.4 - P_{A\cap B}(17) \leq 1$, which becomes 1 if we have additional information that $P_{A\cap B}(17) = 0.1$. In other words, the P-theory can give a result that is consistent with our common sense, but the Z^--system does not.

For the B-system given by (11), we also get $\mu_C(17) = \min\{0.7+0.4, 1\} = 1$ correctly. However, for the cases where $A \subseteq B$ or $B \subseteq A$, the B-system can not give a reasonable result, while the Z^--system works well, as illustrated by the following example:

Example 2. Again, based on common sense, saying B=$\{a\ person\ a = 30 - years - old\ is\ a\ youth\}$ is only partially true with a degree of 0.2 and saying C=$\{\ a\ person\ who\ is\ either\ a\ youth\ or\ a\ juvenile\}$ is also partially true with a degree of 0.2. Obviously, $C \vee B = C$ still represents a 0.2 degree of truth. Based on the Z^--system, we can correctly calculate $\mu_{C\vee B}(30) = \max\{0.2, 0.2\} = 0.2$, $\mu_{C\wedge B}(30) = \min\{0.2, 0.2\} = 0.2$. According to the P-theory, by (15) we can also correctly calculate $P_{C\vee B}(30) = 0.2+0.2-0.2 = 0.2$ and $P_{C\wedge B}(30) = 0.2$. However, according to the B-system, we get $\mu_{C\vee B}(30) = \min\{0.2 + 0.2, 1\} = 0.4$ and $\mu_{C\wedge B}(30) = \max\{0.2 + 0.2 - 1, 0\} = 0$, both of which are incorrect.

There are also cases where both the B-system by (11) and the Z^--system by (7) fail to produce a reasonable result. The following is an example.

Example 3. Consider a set B for persons of medium height. Based on common sense, it is partially true that a 1.7 m person is of medium height with a degree of 0.7. Also, consider a union set E of tall and medium height people, and a union set F of medium height and short people. We can believe that a 1.7 m person belongs to E and F with an increased degree of 0.8 and 0.8, respectively. We observe that $E \cap F = B$. According to the P-theory, by (15) we correctly get $P_{E\cap F}(1.7\,\mathrm{m}) = P_B(1.7\mathrm{m}) = 0.7$. However, the Z^--system incorrectly gives $\mu_{E\cap F}(1.7\mathrm{m}) = \min\{0.8, 0.8\} = 0.8$, while the B-system incorrectly gives $\mu_{E\cap F}(1.7\mathrm{m}) = \max\{0.8 + 0.8 - 1, 0\} = 0.6$.

3.3 A Theorem: Where the Z-System and the B-System Work or Fail

Theorem 2. *For $A \subseteq U$ and $B \subseteq U$ with $P_{A\cap B}(a) \neq 0$, $P_{A\cup B}(a) \neq 1$, $P_{B|A}(a) \neq 1$ and $P_{A|B}(a) \neq 1$, we have*

$$\begin{aligned} &(i)\ \min\{P_A(a) + P_B(a), 1\} > P_{A\cup B}(a) > \max\{P_A(a), P_B(a)\},\\ &(ii)\ \max\{P_A(a) + P_B(a) - 1, 0\} < P_{A\cap B}(a) < \min\{P_A(a), P_B(a)\}. \end{aligned} \tag{19}$$

Proof. (i) With $P_{A\cap B}(a) \neq 0$ or equivalently $P_{A\cap B}(a) > 0$, it follows from (15)(a) that $P_A(a) + P_B(a) > P_{A\cup B}(a)$. Together with $P_{A\cup B}(a) \neq 1$ or $P_{A\cup B}(a) < 1$, we get $\min\{P_A(a) + P_B(a), 1\} > P_{A\cup B}(a)$. Moreover, it follows from (15)(a) and (15)(b) that $P_{A\cup B}(a) = P_B(a) + P_A(a) - P_A(a)P_{B|A}(a) = P_B(a) + P_A(a)[1 - P_{B|A}(a)] > P_B(a)$, when $P_{B|A}(a) \neq 1$. Similarly we also get $P_{A\cup B}(a) > P_A(a)$. That is, $P_{A\cup B}(a) > \max\{P_A(a), P_B(a)\}$.

(ii) It follows from $P_{A\cup B}(a) < 1$ and (15)(a) that $P_{A\cup B}(a) = P_A(a) + P_B(a) - P_{A\cap B}(a) < 1$ or $P_{A\cap B}(a) > P_A(a) + P_B(a) - 1$, that is, $\max\{P_A(a) + P_B(a) - 1, 0\} < P_{A\cap B}(a)$. Moreover, it follows directly from (15)(b) that $P_{A\cap B}(a) < P_A(a)$ when $P_{B|A}(a) < 1$. Similarly we also get $P_{A\cap B}(a) < P_B(a)$. That is, $P_{A\cap B}(a) < \min\{P_A(a), P_B(a)\}$. □

The condition $P_{A\cap B}(a) \neq 0$ or $P_{A\cap B}(a) > 0$ means that an overlap between A and B is measurable, while the condition $P_{A\cup B}(a) \neq 1$ or $P_{A\cup B}(a) < 1$ means that the difference between U and $A \cup B$ is measurable. In the special case where $P_{A\cap B}(a) = 0$ and $P_{A\cup B}(a) = 1$, the B-system is equivalent to the P-theory. This equivalence happens for the $\cup$ operation only if $P_{A\cup B}(a) = 1$, and for the $\cap$ operation only if $P_{A\cap B}(a) = 0$.

Moreover, when either $P_{B|A}(a) = 1$ or $P_{A|B}(a) = 1$, the B-system is equivalent to the P-theory, as stated in Theorem 1.

To gain further insight, we rewrite (19) as

$$\begin{aligned} &\mathbf{B}_{A\cup B}(a) > P_{A\cup B}(a) > Z_{A\cup B}(a) \geq Z_{A\cap B}(a) > P_{A\cap B}(a) > \mathbf{B}_{A\cap B}(a), \\ &\mathbf{B}_{A\cup B}(a) = \min\{P_A(a) + P_B(a), 1\},\ Z_{A\cup B}(a) = \max\{P_A(a), P_B(a)\}, \\ &Z_{A\cap B}(a) = \min\{P_A(a), P_B(a)\},\ \mathbf{B}_{A\cap B}(a) = \max\{P_A(a) + P_B(a) - 1, 0\}. \end{aligned} \tag{20}$$

$P_{A\cup B}(a) > Z_{A\cup B}(a)$ means that the confidence in a disjunctive proposition is discounted by the Z^--system, as in *Example 1*; while $Z_{A\cap B}(a) > P_{A\cap B}(a)$ means that the confidence in a conjunctive proposition is exaggerated by the Z^--system, as in *Example 3*. The difference $e_{dis} = P_{A\cup B}(a) - Z_{A\cup B}(a)$ and the difference $e_{con} = Z_{A\cap B}(a) - P_{A\cap B}(a)$ vary from case to case. When e_{con} and e_{dis} are below a given threshold, the approximation is acceptable, but if they are above a threshold, the approximation becomes unacceptable. This explains why the Z-system works in some cases and fails in others.

Interestingly, the nature of approximation with the B-system complements approximation with the Z^--system. The confidence in a disjunctive proposition is exaggerated because $e_{dis} = P_{A\cup B}(a) - \mathbf{B}_{A\cup B}(a) < 0$, as in *Example 2*; while the confidence in a conjunctive proposition is discounted because $e_{con} = P_{A\cap B}(a) - \mathbf{B}_{A\cap B}(a) > 0$, as in *Examples 2 and 3*. Again, e_{dis} and e_{con} vary from case to case.

Examining (20) further, we observe that the B-system and the Z^--system respectively compute $\mathbf{B}_{A\cup B}(a)$, $\mathbf{B}_{A\cap B}(a)$ and $Z_{A\cup B}(a)$, $Z_{A\cap B}$ from $P_A(a)$ and $P_B(a)$ only. In a contrast, it follows from (15) that the P-theory computes $P_{A\cup B}(a)$ and $P_{A\cap B}(a)$ not only from $P_A(a), P_B(a)$, but also from $P_{B|A}(a), P_{A|B}(a)$. This explains the superiority of the P-theory. In fact, how to handle $P_{B|A}(a), P_{A|B}(a)$ is the core part of making the P-theory based reasoning. For further details, readers are referred to [7], especially Chap 2.

Of course, handling $P_{B|A}(a), P_{A|B}(a)$ incurs higher computation costs and more information. The advantage of the easy implementation of the Z^--system comes from ignoring $P_{B|A}(a), P_{A|B}(a)$, but with a deterioration on performance. It should be noted that many effective implementing algorithms have been developed for the P-theory based reasoning in the past two decades [6, 7]. Thus, there is no need to use the Z^--system with its poor performance, if we have no difficulty using the P-theory.

4 Avoiding the Controversial Definition of the Complement Set

4.1 Complement Concept and Complement Set

For an entity E comprised of two parts, A and B, part A is commonly referred as the complement of part B. In other words, the "complement" concept is associated with two points. One is its way of composition $\bigoplus$, and the other is that two parts via $\bigoplus$ becomes one entity, i.e., $E = A \bigoplus B$. More precisely, A is said to be the complement of B with respect to E under $\bigoplus$.

The "complement" concept has been widely adopted in mathematical systems. A simple example is that 6 is a complement number of 4 with respect to 10 under the addition $+$. Generally, we say a number $0 \le n \le b$ is a complement number of $0 \le m \le b$ with respect to b under the addition $+$ if there is a unique number n satisfying $n + m = b$. For simplicity, one usually says

that 6 is a complement number of 4, without explicitly mentioning 10 and +. Such simplification, however, may cause some confusion about the "complement" concept. Choosing (9) as the complement set of A in the Z-system is an example of such confusion.

For two sets, A and B, we have $\cup$ as $\bigoplus$ and U as E; that is, $U = A \cup B$. However, we are still unable to say that A is the complement set of B with respect to U under $\cup$, because there are many A that satisfy $U = A \cup B$ for a given B. To make a unique A, we need to add $A \cap B = \emptyset$. This leads to the complement set definition given by (3). For a characteristic function by (5), the complement set definition is given by (8). It says that $\mu_{\neg A}(a)$ is the complement of $\mu_A(a)$ with respect to the constant function $\mu_U(a) = 1$ under the operation in (7), subject to $0 = \min\{\mu_A(a), \mu_{\neg A}(a)\}$ for its uniqueness. For the binary valued function by (5), the complement $\mu_{\neg A}(a)$ by (8) is equivalent to the direct expression given by (9). Therefore, in such a case, either (8) or (9) can be used as the definition of the complement $\mu_{\neg A}(a)$.

For a real valued case by (10), the complement $\mu_{\neg A}(a)$ by (8) is no longer equivalent to that given by (9). In this case, the correct way is to choose (8) to define the complement $\mu_{\neg A}(a)$. However, except for those $\mu_A(a)$ in the degenerated case by (5), there is no solution for $\mu_{\neg A}(a)$ by (8) for a real-valued $\mu_A(a)$ by (10), i.e., the complement set does not exist.

4.2 Avoiding a Controversial Definition in Zadeh's Complement Set

Unfortunately, Zadeh mistakenly selected (9) as the definition of the complement $\mu_{\neg A}(a)$ in the Z-system, though the $\mu_{\neg A}(a)$ given by (9) is the complement of $\mu_A(a)$ with respect to $\mu_T(a) = 1$ under +. In fact, the operation + has been excluded from (7).

More precisely, $1 - \mu_A(a)$ can be regarded as the "mirror" or "conjugate" function with respect to $\mu_T(a) = 1$. To distinguish it from $\mu_{\neg A}(a)$, we denote this conjugate function as

$$\mu_{\Xi A}(a) = 1 - \mu_A(a). \tag{21}$$

The confusion of this concept with the commonly adopted complement concept discussed in Sect. 4.1 has caused bad consequences at least in two aspects.

First, it produces unnecessary mistakes in applications of the Z-system. Though it maybe clear to certain senior fuzzy researchers that the "complement" concept is different from the commonly adopted complement concept, it may be not clear to many new comers or those who simply apply the Z-system for practical uses. In their applications, a classical logical reasoning problem is extended into a fuzzy logic problem via simply turning a binary valued characteristic function into a fuzzy membership. This type of practice may cause mistakes and lead to unsuccessful applications, unless the original logical problem does not involve logical negation either directly or indirectly

(e.g., via the logical implication $A \rightarrow B$). This type of failures can be avoided by renaming the Zadeh's "complement" definition. Also, it is a well adopted convention in the scientific community that one should not use duplicately a name or terminology, that has already been well adopted, on a different concept for unnecessary confusions.

Second, it contradicts classical logic as well as our common sense. If we follow our common sense to define the truth $\mu_T(a)$ by 1 or even a constant c, it follows the MAX operation by (7)(a) that $c = \mu_T(a) = \mu_{A\cup\neg A}(a) = \max\{\mu_A(a), 1 - \mu_A(a)\}$, which is only possible when $\mu_A(a)$ takes a binary value of 0 or 1. Thus, the Z-system collapses back to classical logic. To be consistent, we can only let $\mu_T(a) = \max\{\mu_A(a), 1 - \mu_A(a)\}$, which is actually a variable that varies between 0.5 and 1 as a varies. Similarly, it follows from (7)(b) that $\mu_F(a) = \mu_{A\cap\neg A}(a) = \min\{\mu_A(a), 1-\mu_A(a)\}$, which forces us either revert to classical logic or to accept a false membership function $\mu_F(a) = \min\{\mu_A(a), 1 - \mu_A(a)\}$ that varies between 0 and 0.5 as a varies. In other words, a truth in the Z-system is no different to any fuzzy membership function $0.5 \leq \mu_A(a) \leq 1$, while a false in the Z-system is no different to any fuzzy membership function $0 \leq \mu_A(a) \leq 0.5$.

The above confusion and conflicts incur a long-running debate in the literature [3, 8]. Facing the challenges raised by critics, various kinds of reactions have arisen from the "pro-fuzzy" community. There are some "pro-fuzzy" people who worship the Z-system and simply ignore any controversy or overreact to challenges, which will not be further discussed here. There are also some researchers who attempt to resolve the controversy [1, 5]. To avoid the awkwardness, one remedy without explicitly discarding away (9) is the bold operations in (11) that modifies (7) such that $1 - \mu_A(a)$ becomes equivalent to the complement set in a similar way to (8). As a result, $\mu_{A\cup\neg A}(a) = 1$ and $\mu_{A\cap\neg A}(a) = 0$ still hold, and the above mentioned confusion and conflicts have been removed.

Moreover, there are also "pro-fuzzy" researchers who choose to defense the Z-system. One key argument is that the above so-called conflict represents naturally a feature for handling partial membership or a multi-valued truth due to missing information. For example, one argument maybe illustrated via a scenario that an agent knows exactly "John is 55 years old", and is asked the question "Is John old?". Considering partial membership, the agent's answer may be "Well, John is a little old, but not quite so". That is, it is not necessary to assign 1 to $old(55) \vee \neg old(55)$. It appears naturally from (7) that $\mu_{old\cup\neg old}(55) = \max\{\mu_{old}(55), 1 - \mu_{old}(55)\} < 1$ has no problem. However, there is also a hidden confusion. We fell acceptable that the value of $old(55) \vee \neg old(55)$ is smaller than 1 does not mean that we can accept $\mu_{old\cup\neg old}(55) = \max\{\mu_{old}(55), 1 - \mu_{old}(55)\} < 1$. For $0 < \mu_{old}(a) < 1$, the value of $old(55) \vee \neg old(55)$ is not equal to $\mu_{old\cup\neg old}(55) = 1$ with $old \cup \neg old = U$. That is, there is a confusion between the membership $\mu_{p\vee q}(a)$ and the value of $p(a) \vee q(a)$. The latter already involves a predicate logic.

The definition $\mu_{p\vee q}(a) = \max\{\mu_p(a), \mu_q(a)\}$ can not automatically lead to that the value of $p(a) \vee q(a)$ is given by $\max\{\mu_p(a), \mu_q(a)\}$.

Also, missing information can not be a real excuse too. It would be one if the conflicts and confusion could not be avoided by any other operation based on the same information. Actually, the bold operations in (11) refute this argument. The success of (11) on resolving the above discussed conflicts and confusion actually comes from honoring the additive principle behind both the set theory and the P-theory. Still, one may further argue that (11) brings other problems, e.g., making the idempotent law (i.e. $A \cup A = A$ and $A \cap A = A$) invalid. Actually, the reason behind this problem is just what has been discussed at the end of Sect. 3.3, and can be solved if the dependence type of $P_{B|A}(a), P_{A|B}(a)$ in (16) are also considered [4]. Further ahead along this line, we are finally lead to the P-theory.

If one wants all the axioms of Boolean algebra to remain hold, the P-theory is a best choice for the development of models of uncertainty or partial truth. If one does not want to use the P-theory due to higher computation costs and requiring extra information for handling $P_{B|A}(a), P_{A|B}(a)$, one has to abandon some axioms of Boolean algebra. Of course, depending on applications, one may choose to abandon the idempotent law or the exclude-middle law or other. However, one can not introduce a conceptual confusion by duplicately using a terminology with a well known meaning to name a new and different concept. If one abandons the complement definition $1 - \mu_A(a)$ or just simply renaming $1 - \mu_A(a)$ by (21), everything is fine with the min–max operations, which may be worth some further study, especially on the joint role of $\mu_A(a)$ and $\mu_{\Xi A}(a)$ in the Z-system.

5 Conclusions

We have elucidated the links and differences between probability theory and Zadeh's fuzzy system. When one set is included in the other (i.e., $A \subseteq B$ or $B \subseteq A$ either in a almost surely sense or in a Zadeh fuzzy set sense), we have proved that the P-theory and the Z^--system perform equivalently in computer reasoning that does not involve complement operation. Moreover, we have proved that both the Z^--system and the B-system attempt to approximate the P-theory. In some cases, these approximations are acceptable, and they have the advantage of easy implementation. In other cases, however we get bad approximations that are incapable of producing reasonable results. The failures arise from either its incapability of capturing additive structures or inadequate handling of the dependence relation across two sets, or both. It seems that many efforts are needed to investigate some error bounds of approximation and to control the bounds. Furthermore, we show that the controversy about the definition of the complement set arises from confusion over the "complement" concept.

References

1. Dubois, D. and Prade, H. (1980) *Fuzzy Sets and Systems: Theory and Applications*, Academic, New York
2. Dubois, D. and Prade, H. (1994) Partial Truth is not Uncertainty, *IEEE Expert*, pp 15–19
3. Elkan, C. (1994) The Paradoxical Success of Fuzzy Logic, *IEEE Expert*, pp 3–8. Also obtained a honorable mention in a best-written paper competition on *Proceedings of AAAI'93*, July 1993, pp 698–703
4. Gao, Q.S. (2005) The errors and shortcomings of Zadeh's fuzzy set theory and its overcoming – C-fuzzy set theory, *Journal of Dalian University of Technology*, 45(5)
5. Giles, R. (1976) Lukasiewicz logic and fuzzy theory, *International Journal of Man–Machine Studies*, 8, pp 313–327
6. Jensen, F.V. (1996) *An Introduction to Bayesian Networks*, University of College London Press, London, UK
7. Pearl, J. (1988) *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufman, San Fransisco, CA
8. Special Issue, (1994) Responses on the Paradoxical Success of Fuzzy Logic, *IEEE Expert*, pp 9–46
9. Xu, L. and Yan, P.F. (1985) Some Investigations on Subjective Probability Distribution, *Proceedings of 5th National Conference on Pattern Recognition and Machine Intelligence*, May 27–30, 1986. Xian, China

Three Approaches to Missing Attribute Values: A Rough Set Perspective

Jerzy W. Grzymala-Busse

Department of Electrical Engineering and Computer Science, University of Kansas, Lawrence, KS 66045, USA
and
Institute of Computer Science, Polish Academy of Sciences, 01-237 Warsaw, Poland
Jerzy@ku.edu
http://lightning.eecs.ku.edu/index.html

Summary. A new approach to missing attribute values, based on the idea of an attribute-concept value, is studied in the paper. This approach, together with two other approaches to missing attribute values, based on "do not care" conditions and lost values are discussed using rough set methodology, including attribute-value pair blocks, characteristic sets, and characteristic relations. Characteristic sets are generalization of elementary sets while characteristic relations are generalization of the indiscernibility relation. Additionally, three definitions of lower and upper approximations are discussed and used for induction of certain and possible rules.

1 Introduction

In this chapter data sets are presented in the form of decision tables, where columns are labeled by variables and rows by case (or example) names. Variables are categorized into independent variables, also called attributes, and dependent variables, also called decisions. Usually decision tables have only one decision. The set of all cases that correspond to the same decision value is called a concept (or a class).

In most papers on rough set theory it is assumed that values, for all variables and all cases, are specified. For such tables the indiscernibility relation, one of the most fundamental ideas of rough set theory, describes cases that can be distinguished from each other.

However, in many real-life applications, data sets have missing attribute values, or, in different words, the corresponding decision tables are incompletely specified. For simplicity, incompletely specified decision tables will be called incomplete decision tables.

In data mining two main strategies are used to deal with missing attribute values. The former strategy is based on conversion of incomplete data sets (i.e., data sets with missing attribute values) into complete data sets and then

J.W. Grzymala-Busse: *Three Approaches to Missing Attribute Values: A Rough Set Perspective*, Studies in Computational Intelligence (SCI) **118**, 139–152 (2008)
www.springerlink.com

acquiring knowledge, e.g., by rule induction or tree generation from complete data sets. In this strategy conversion of incomplete data sets to complete data sets is a preprocessing to the main process of data mining. In the later strategy, knowledge is acquired from incomplete data sets taking into account that some attribute values are missing. The original data sets are not converted into complete data sets.

Typical examples of the former strategy include [4, 11]:

- Replacing missing attribute values by the most common (most frequent) value of the attribute.
- Replacing missing attribute values restricted to the concept. For each concept missing attribute values are replaced by the most common attribute value restricted to that concept.
- For numerical attributes, missing attribute value may be replaced by the attribute average value.
- For numerical attributes, missing attribute value may be replaced by the attribute average value restricted to the concept.
- Assigning all possible values of the attribute. A case with a missing attribute value is replaced by a set of new cases, in which the missing attribute value is replaced by all possible values of the attribute.
- Assigning all possible values of the attribute restricted to the concept.
- Ignoring cases with missing attribute values. An original data set, with missing attribute values, is replaced by a new data set with deleted cases containing missing attribute values.
- Considering missing attribute values as special values.

The later strategy is exemplified by the C4.5 approach to missing attribute values [21] or by a modified LEM2 algorithm [9,13]. In both algorithms original data sets with missing attribute values are not preprocessed, i.e., data sets are not preliminarily converted into complete data sets.

Note that from the view point of rough set theory, in the former strategy the conventional indiscernibility relation may be applied to describe the process of data mining since, after preprocessing, the data set is complete (has no missing attribute values). Furthermore, lower and upper approximations, other basic ideas of rough set theory, are also conventional.

In this chapter we will concentrate on the later strategy used for rule induction, i.e., we will assume that the rule sets are induced from the original data sets, with missing attribute values, not preprocessed as in the former strategy.

We will assume that there are three reasons for decision tables to be incomplete. The first reason is that an attribute value, for a specific case, is lost. For example, originally the attribute value was known, however, due to a variety of reasons, currently the value is not available. Maybe it was recorded but later it was erased. The second possibility is that an attribute value was not relevant – the case was decided to be a member of some concept, i.e., was classified, or diagnosed, in spite of the fact that some attribute values were not

known. For example, it was feasible to diagnose a patient in spite of the fact that some test results were not taken (here attributes correspond to tests, so attribute values are test results). Since such missing attribute values do not matter for the final outcome, we will call them "do not care" conditions. The third possibility is a partial "do not care" condition: we assume that the missing attribute value belongs to the set of typical attribute values for all cases from the same concept. Such a missing attribute value will be called an attribute-concept value. Calling it concept "do not care" condition would be perhaps better, but this name is too long.

The main objective of this chapter is to study incomplete decision tables, assuming that in the same decision table some attribute values may be lost, some may be "do not care" conditions, and some may be attribute-concept values. Decision tables with lost values and "do not care" conditions were studied in [7–9, 12].

For such incomplete decision tables there are three special cases: in the first case all missing attribute values are lost, in the second case all missing attribute values are "do not care" conditions, and in the third case all missing attribute vales are attribute-concept values. Incomplete decision tables in which all attribute values are lost, from the viewpoint of rough set theory, were studied for the first time in [13], where two algorithms for rule induction, modified to handle lost attribute values, were presented. This approach was studied later in [23–25], where the indiscernibility relation was generalized to describe such incomplete decision tables.

On the other hand, incomplete decision tables in which all missing attribute values are "do not care" conditions, again from the view point of rough set theory, were studied for the first time in [4], where a method for rule induction was introduced in which each missing attribute value was replaced by all values from the domain of the attribute. Originally such values were replaced by all values from the entire domain of the attribute, later by attribute values restricted to the same concept to which a case with a missing attribute value belongs. Such incomplete decision tables, with all missing attribute values being "do not care conditions", were extensively studied in [14, 15], including extending the idea of the indiscernibility relation to describe such incomplete decision tables.

Rough set methodology for incomplete decision tables with missing attribute values of the type attribute-concept values is presented in this chapter for the first time, though it was briefly mentioned in [9].

In general, incomplete decision tables are described by characteristic relations, in a similar way as complete decision tables are described by indiscernibility relations [7].

For complete decision tables, once the indiscernibility relation is fixed and the concept (a set of cases) is given, the lower and upper approximations are unique.

For incomplete decision tables, for a given characteristic relation and the concept, there are three different possible ways to define lower and upper

approximations, called singleton, subset, and concept approximations [7]. The singleton lower and upper approximations were studied in [14,15,23–25]. Similar ideas were studied in [2, 22, 26–28]. In this chapter we further discuss applications to data mining of all three kinds of approximations: singleton, subset and concept. As it was observed in [7], singleton lower and upper approximations are not applicable in data mining.

The next topic of this chapter is demonstrating how certain and possible rules may be computed from incomplete decision tables. An extension of the well-known LEM2 (Learning from Examples Module, version 2) rule induction algorithm [1, 5], called MLEM2, was introduced in [6]. LEM2 is a component of the LERS (Learning from Examples based on Rough Sets) data mining system. Originally, MLEM2 induced certain rules from incomplete decision tables with numerical attributes and with missing attribute values interpreted as lost. Using the idea of lower and upper approximations for incomplete decision tables, MLEM2 was further extended to induce both certain and possible rules from a decision table with some numerical attributes and with some attribute values being lost and some attribute values being "do not care" conditions.

A preliminary version of this chapter was presented at the Workshop on Foundation of Data Mining, associated with the Fourth IEEE International Conference on Data Mining, Brighton, UK, November 1–4, 2004 [10].

2 Complete Data: Elementary Sets and Indiscernibility Relation

An example of a decision table, taken from [9], is presented in Table 1.

Rows of the decision table represent *cases*, while columns are labeled by *variables*. The set of all cases will be denoted by U. In Table 1, $U = \{1, 2, \ldots, 8\}$.

Table 1. A complete decision table

	Attributes			Decision
Case	Temperature	Headache	Nausea	Flu
1	High	Yes	No	Yes
2	Very high	Yes	Yes	Yes
3	High	No	No	No
4	High	Yes	Yes	Yes
5	High	Yes	Yes	No
6	Normal	Yes	No	No
7	Normal	No	Yes	No
8	Normal	Yes	No	Yes

Independent variables are called *attributes* and a dependent variable is called a *decision* and is denoted by d. The set of all attributes will be denoted by A. In Table 1, $A = \{Temperature, Headache, Nausea\}$. Any decision table defines a function ρ that maps the direct product of U and A into the set of all values. For example, in Table 1, $\rho(1, Temperature) = high$. Function ρ describing Table 1 is completely specified (total). A decision table with completely specified function ρ will be called *completely specified*, or, for the sake of simplicity, *complete*.

Rough set theory [19,20] is based on the idea of an indiscernibility relation, defined for complete decision tables. Let B be a nonempty subset of the set A of all attributes. The indiscernibility relation IND(B) is a relation on U defined for $x, y \in U$ as follows

$$(x, y) \in IND(B) \text{ if and only if } \rho(x, a) = \rho(y, a) \text{ for all } a \in B.$$

The indiscernibility relation IND(B) is an equivalence relation. Equivalence classes of IND(B) are called *elementary sets* of B and are denoted by $[x]_B$. For example, for Table 1, elementary sets of IND(A) are {1}, {2}, {3}, {4, 5}, {6, 8}, {7}. The indiscernibility relation IND(B) may be computed using the idea of blocks of attribute-value pairs. Let a be an attribute, i.e., $a \in A$ and let v be a value of a for some case. For complete decision tables if $t = (a, v)$ is an attribute-value pair then a block of t, denoted $[t]$, is a set of all cases from U that for attribute a have value v. For Table 1,

[(Temperature, high)] = {1, 3, 4, 5},
[(Temperature, very_high)] = {2},
[(Temperature, normal)] = {6, 7, 8},
[(Headache, yes)] = {1, 2, 4, 5, 6, 8},
[(Headache, no)] = {3, 7},
[(Nausea, no)] = {1, 3, 6, 8},
[(Nausea, yes)] = {2, 4, 5, 7}.

The indiscernibility relation IND(B) is known when known are all elementary sets of IND(B). Such elementary sets of B are intersections of the corresponding attribute-value pair blocks, i.e., for any case $x \in U$,

$$[x]_B = \cap\{[(a, \rho(a, v))] \mid a \in B\}$$

We will illustrate the idea how to compute elementary sets of B for Table 1 and $B = A$.

$[1]_A = [(Temperature, high)] \cap [(Headache, yes)] \cap [(Nausea, no)] = \{1\}$,
$[2]_A = [(Temperature, very\_high)] \cap [(Headache, yes)] \cap [(Nausea, yes)] = \{2\}$,
$[3]_A = [(Temperature, high)] \cap [(Headache, no)] \cap [(Nausea, no)] = \{3\}$,
$[4]_A = [5]_A = [(Temperature, high)] \cap [(Headache, yes)] \cap [(Nausea, yes)] = \{4, 5\}$,

$[6]_A = [8]_A = [(Temperature, normal)] \cap [(Headache, yes)] \cap [(Nausea, no] = \{6, 8\}$,

$[7]_A = [(Temperature, normal)] \cap [(Headache, no] \cap [(Nausea, yes)] = \{7\}$.

3 Incomplete Data: Characteristic Sets and Characteristic Relations

For data sets with missing attribute values, the corresponding function ρ is incompletely specified (partial). A decision table with incompletely specified function? will be called *incompletely specified*, or *incomplete*.

In the sequel we will assume that all decision values are specified, i.e., they are not missing. Also, we will assume that all missing attribute values are denoted by "?", by "*" or by "–", lost values will be denoted by "?", "do not care" conditions will be denoted by "*", and attribute-concept values by "–". Additionally, we will assume that for each case at least one attribute value is specified.

Incomplete decision tables are described by characteristic relations instead of indiscernibility relations. Also, elementary sets are replaced by characteristic sets. The characteristic set was called a (binary) neighborhood in [16–18]. An example of an incomplete table is presented in Table 2.

For incomplete decision tables the definition of a block of an attribute-value pair must be modified.

- If an attribute a there exists a case x such that $\rho(x, a) = ?$, i.e., the corresponding value is lost, then the case x should not be included in any block $[(a, v)]$ for all values v of attribute a.
- If for an attribute a there exists a case x such that the corresponding value is a "do not care" condition, i.e., $\rho(x, a) = *$, then the corresponding case x should be included in blocks $[(a, v)]$ for all specified values v of attribute a.

Table 2. An incomplete decision table

	Attributes			Decision
Case	Temperature	Headache	Nausea	Flu
1	High	–	No	Yes
2	Very high	Yes	Yes	Yes
3	?	No	No	No
4	High	Yes	Yes	Yes
5	High	?	Yes	No
6	Normal	Yes	No	No
7	Normal	No	Yes	No
8	–	Yes	*	Yes

- If for an attribute a there exists a case x such that the corresponding value is a attribute-concept value, i.e., $\rho(x, a) = -$, then the corresponding case x should be included in blocks $[(a, v)]$ for all specified values v of attribute a that are members of the set $V(x, a)$, where

$$V(x, a) = \{\rho(y, a) \mid \rho(y, a) \textit{ is specified}, y \in U, \rho(y, d) = \rho(x, d)\},$$

and d is the decision.

These modifications of the definition of the block of attribute-value pair are consistent with the interpretation of missing attribute values, lost, "do not care" conditions, and attribute-concept values. Also, note that the attribute-concept value is the most universal, since if $V(x, a) = \emptyset$, the definition of the attribute-concept value is reduced to the lost value, and if $V(x, a)$ is the set of all values of an attribute a, the attribute-concept value becomes a "do not care" condition.

For Table 2, for case 1, $\rho(1, Headache) = -$, and V(1, Headache) = {yes}, so we add the case 1 to [(Headache, yes)]. For case 3, $\rho(3, Temperature) = ?$, hence case 3 is not included in either of the following sets: [(Temperature, high)], [(Temperature, very_high)], and [(Temperature, normal)]. Similarly, $\rho(5, Headache) = ?$, so the case 5 is not included in [(Headache, yes)] and [(Headache, no)]. Also, $\rho(8, Temperature) = -$, and $V(8, Temperature) = \{high, very\_high\}$, so the case 8 is a member of both [(Temperature, high)] and [(Temperature, very_high)]. Finally, $\rho(8, Nausea) = *$, so the case 8 is included in both [(Nausea, no)] and [(Nausea, yes)]. Thus,

[(Temperature, high)] = {1, 4, 5, 8},
[(Temperature, very_high)] = {2, 8},
[(Temperature, normal)] = {6, 7},
[(Headache, yes)] = {1, 2, 4, 6, 8},
[(Headache, no)] = {3, 7},
[(Nausea, no)] = {1, 3, 6, 8},
[(Nausea, yes)] = {2, 4, 5, 7, 8}.

For a case $x \in U$, the *characteristic set* $K_B(x)$ is defined as the intersection of the sets $K(x, a)$, for all $a \in B$.

If $\rho(x, a)$ is specified, then $K(x, a)$ is the block $[(a, \rho(x, a)]$ of attribute a and its value $\rho(x, a)$. If $\rho(x, a) = *$ or $\rho(x, a) = ?$ then the set $K(x, a) = U$. If $\rho(x, a) = -$ and $V(x, a)$ is nonempty, then the corresponding set $K(x, a)$ is equal to the union of all blocks of attribute-value pairs (a, v), where $v \in V(x, a)$. If $V(x, a)$ is empty, then $K(x, a) = \{x\}$.

The way of computing characteristic sets needs a comment. For both "do not care" conditions and lost values the corresponding set $K(x, a)$ is equal to U because the corresponding attribute a does not restrict the set $K_B(x)$: if $\rho(x, a) = *$, the value of the attribute a is irrelevant; if $\rho(x, a) = ?$, only existing values need to be checked. However, the case when $\rho(x, a) = -$ is different, since the attribute a restricts the set $K_B(x)$. Furthermore, the description of

$K_B(x)$ should be consistent with other (but similar) possible approaches to missing attribute values, e.g., an approach in which each missing attribute value is replaced by the most common attribute value restricted to a concept. Here the set $V(x, a)$ contains a single element and the characteristic relation is an equivalence relation. Our definition is consistent with this special case in the sense that if we compute a characteristic relation for such a decision table using our definition or if we compute the indiscernibility relation as for complete decision tables using definitions from Sect. 2, the result will be the same. For Table 2 and $B = A$,

$K_A(1) = \{1, 4, 5, 8\} \cap \{1, 2, 4, 6, 8\} \cap \{1, 3, 6, 8\} = \{1, 8\}$,
$K_A(2) = \{2, 8\} \cap \{1, 2, 4, 6, 8\} \cap \{2, 4, 5, 7, 8\} = \{2, 8\}$,
$K_A(3) = U \cap \{3, 7\} \cap \{1, 3, 6, 8\} = \{3\}$,
$K_A(4) = \{1, 4, 5, 8\} \cap \{1, 2, 4, 6, 8\} \cap \{2, 4, 5, 7, 8\} = \{4, 8\}$,
$K_A(5) = \{1, 4, 5, 8\} \cap U \cap \{2, 4, 5, 7, 8\} = \{4, 5, 8\}$,
$K_A(6) = \{6, 7\} \cap \{1, 2, 4, 6, 8\} \cap \{1, 3, 6, 8\} = \{6\}$,
$K_A(7) = \{6, 7\} \cap \{3, 7\} \cap \{2, 4, 5, 7, 8\} = \{7\}$, and
$K_A(8) = (\{1, 4, 5, 8\} \cup \{2, 8\}) \cap \{1, 2, 4, 6, 8\} \cap U = \{1, 2, 4, 8\}$.

The characteristic set $K_B(x)$ may be interpreted as the smallest set of cases that are indistinguishable from x using all attributes from B, and using given interpretation of missing attribute values. Thus, $K_A(x)$ is the set of all cases that cannot be distinguished from x using all attributes. Also, note that the previous definition is an extension of a definition of $K_B(x)$ from [7–9]: for decision tables with only lost values and "do not care" conditions, both definitions are identical.

The *characteristic relation* $R(B)$ is a relation on U defined for $x, y \in U$ as follows

$$(x, y) \in R(B) \text{ if and only if } y \in K_B(x).$$

The characteristic relation $R(B)$ is reflexive but – in general – it does not need to be symmetric or transitive. Also, the characteristic relation $R(B)$ is known if we know characteristic sets $K_B(x)$ for all $x \in U$. In our example,

$$R(A) = \{(1, 1), (1, 8), (2, 2), (2, 8), (3, 3), (4, 4), (4, 8), (5, 4), (5, 5), (5, 8), (6, 6), (7, 7), (8, 1), (8, 2), (8, 4), (8, 8)\}$$

For decision tables, in which all missing attribute values are lost, a special characteristic relation $LV(B)$ was defined by Stefanowski and Tsoukias in [24], see also [23, 25]. Characteristic relation $LV(B)$ is reflexive, but – in general – it does not need to be symmetric or transitive.

For decision tables where all missing attribute values are "do not care" conditions a special characteristic relation $DCC(B)$ was defined by Kryszkiewicz in [14], see also, e.g., [15]. Relation $DCC(B)$ is reflexive and symmetric but – in general – is not transitive.

Obviously, characteristic relations $LV(B)$ and $DCC(B)$ are special cases of the characteristic relation $R(B)$. For a completely specified decision table, the characteristic relation $R(B)$ is reduced to IND(B).

4 Lower and Upper Approximations

For completely specified decision tables lower and upper approximations are defined using the indiscernibility relation. Any finite union of elementary sets of B is called a *B-definable set*. Let X be any subset of the set U of all cases. The set X is called concept and is usually defined as the set of all cases defined by a specific value of the decision. In general, X is not a B-definable set. However, set X may be approximated by two B-definable sets, the first one is called a *B-lower approximation* of X, denoted by $\underline{B}X$ and defined as follows

$$\{x \in U \mid [x]_B \subseteq X\}.$$

The second set is called an *B-upper approximation* of X, denoted by $\overline{B}X$ and defined as follows

$$\{x \in U \mid [x]_B \cap X \neq \emptyset\}.$$

The above way of computing lower and upper approximations, by constructing them from singletons x, will be called the *first method*. The B-lower approximation of X is the greatest B-definable set, contained in X. The B-upper approximation of X is the least B-definable set containing X.

As it was observed in [19], for complete decision tables we may use a second method to define the B-lower approximation of X, by the following formula

$$\underline{B}X = \cup\{[x]_B \mid x \in U, [x]_B \subseteq X\},$$

and the B-upper approximation of X may de defined, using the second method, by

$$\overline{B}X = \cup\{[x]_B \mid x \in U, [x]_B \cap X \neq \emptyset).$$

For Table 1 and $B = A$, A-lower and A-upper approximations are:

$$\underline{A}\{1, 2, 4, 8\} = \{1, 2\},$$
$$\underline{A}\{3, 5, 6, 7\} = \{3, 7\},$$
$$\overline{A}\{1, 2, 4, 8\} = \{1, 2, 4, 5, 6, 8\},$$
$$\overline{A}\{3, 5, 6, 7\} = \{3, 4, 5, 6, 7, 8\}.$$

For incompletely specified decision tables lower and upper approximations may be defined in a few different ways. To begin with, the definition of definability should be modified. Any finite union of characteristic sets of B is called a *B-definable* set. Following [7], we suggest three different definitions of approximations. Again, let X be a concept, let B be a subset of the set A of all attributes, and let $R(B)$ be the characteristic relation of the incomplete decision table with characteristic sets $K_B(x)$, where $x \in U$. Our first definition uses a similar idea as in the previous articles on incompletely specified decision tables [14, 15, 23–25], i.e., lower and upper approximations are sets of singletons from the universe U satisfying some properties. Thus we are defining

lower and upper approximations by analogy with the above first method, by constructing both sets from singletons. We will call these definitions *singleton*. A singleton B-lower approximation of X is defined as follows:

$$\underline{B}X = \{x \in U \mid K_B(x) \subseteq X\},$$

A singleton B-upper approximation of X is

$$\overline{B}X = \{x \in U \mid K_B(x) \cap X \neq \emptyset\}.$$

In our example presented in Table 2 let us say that $B = A$. Then the singleton A-lower and A-upper approximations of the two concepts: {1, 2, 4, 8} and {3, 5, 6, 7} are:

$$\underline{A}\{1,2,4,8\} = \{1,2,4,8\},$$
$$\underline{A}\{3,5,6,7\} = \{3,6,7\},$$
$$\overline{A}\{1,2,4,8\} = \{1,2,4,5,8\},$$
$$\overline{A}\{3,5,6,7\} = \{3,5,6,7\}.$$

Note that $\overline{A}\{3,5,6,7\} = \{3,5,6,7\}$. However, the set {3, 5, 6, 7} is not A-definable, so a set of rules, induced from {3, 5, 6, 7}, cannot cover precisely this set. In general, singleton approximations should not be used for data mining.

The second method of defining lower and upper approximations for complete decision tables uses another idea: lower and upper approximations are unions of elementary sets, subsets of U. Therefore we may define lower and upper approximations for incomplete decision tables by analogy with the second method, using characteristic sets instead of elementary sets. There are two ways to do this. Using the first way, a *subset* B-lower approximation of X is defined as follows:

$$\underline{B}X = \cup\{K_B(x) \mid x \in U, K_B(x) \subseteq X\},$$

A *subset* B-upper approximation of X is

$$\overline{B}X = \cup\{K_B(x) \mid x \in U, K_B(x) \cap X \neq \emptyset\}.$$

Since any characteristic relation $R(B)$ is reflexive, for any concept X, singleton B-lower and B-upper approximations of X are subsets of subset B-lower and B-upper approximations of X, respectively. For the same the decision presented in Table 2, the subset A-lower and A-upper approximations are:

$$\underline{A}\{1,2,4,8\} = \{1,2,4,8\},$$
$$\underline{A}\{3,5,6,7\} = \{3,6,7\},$$
$$\overline{A}\{1,2,4,8\} = \{1,2,4,5,8\},$$
$$\overline{A}\{3,5,6,7\} = \{3,4,5,6,7,8\}.$$

The second possibility is to modify the subset definition of lower and upper approximation by replacing the universe U from the subset definition by a concept X. A *concept* B-lower approximation of the concept X is defined as follows:

$$\underline{B}X = \cup\{K_B(x) \mid x \in X, K_B(x) \subseteq X\},$$

Obviously, the subset B-lower approximation of X is the same set as the concept B-lower approximation of X. A *concept* B-upper approximation of the concept X is defined as follows:

$$\overline{B}X = \cup\{K_B(x) \mid x \in X, K_B(x) \cap X \neq \emptyset\} = \cup\{K_B(x) \mid x \in X\}.$$

The concept B-upper approximation of X are subsets of the subset B-upper approximations of X. For the decision presented in Table 2, the concept A-lower and A-upper approximations are:

$$\underline{A}\{1, 2, 4, 8\} = \{1, 2, 4, 8\},$$
$$\underline{A}\{3, 5, 6, 7\} = \{3, 6, 7\},$$
$$\overline{A}\{1, 2, 4, 8\} = \{1, 2, 4, 8\},$$
$$\overline{A}\{3, 5, 6, 7\} = \{3, 4, 5, 6, 7, 8\}.$$

For complete decision tables, all three definitions of lower approximations, singleton, subset and concept, coalesce to the same definition. Also, for complete decision tables, all three definitions of upper approximations coalesce to the same definition. This is not true for incomplete decision tables, as our example shows.

5 Rule Induction

The same idea of blocks of attribute-value pairs is used in the rule induction algorithm LEM2. LEM2 explores the search space of attribute-value pairs. Its input data file is a lower or upper approximation of a concept, so its input data file is always consistent. Rules induced from the lower approximation of the concept *certainly* describe the concept, so they are called *certain*. On the other hand, rules induced from the upper approximation of the concept describe the concept only *possibly* (or *plausibly*), so they are called *possible* [3].

Rules in LERS format (every rule is equipped with three numbers, the total number of attribute-value pairs on the left-hand side of the rule, the total number of cases correctly classified by the rule during training, and the total number of training cases matching the left-hand side of the rule) induced from Table 2 using concept approximations are:
the certain rule set:

2, 3, 3
(Temperature, high) & (Headache, yes) —> (Flu, yes)

1, 2, 2
(Temperature, very_high) —> (Flu, yes)
1, 2, 2
(Temperature, normal) —> (Flu, no)
1, 2, 2
(Headache, no) —> (Flu, no)

and the possible rule set:

2, 3, 3
(Temperature, high) & (Headache, yes) —> (Flu, yes)
1, 2, 2
(Temperature, very_high) —> (Flu, yes)
2, 1, 3
(Temperature, high) & (Nausea, yes) —> (Flu, no)
1, 2, 2
(Temperature, normal) —> (Flu, no)
1, 2, 2
(Headache, no) —> (Flu, no)

6 Conclusions

Three approaches to missing attribute values are presented in a unified way. The main applied tool is a characteristic relation, a generalization of the indiscernibility relation. It is shown that all three approaches to missing attribute values may be described using the same idea of attribute-value blocks. Moreover, attribute-value blocks are useful not only for computing characteristic sets but also for computing characteristic relations, lower and upper approximations, and, finally for rule induction. Additionally, using attribute-value blocks, it is quite easy to combine a few strategies to handle missing attribute values within the same data set. Thus, the entire data mining process, starting from computing characteristic relations and ending with rule induction, may be implemented using the same simple tool: attribute-value blocks.

References

1. C. C. Chan and J. W. Grzymala-Busse: On the attribute redundancy and the learning programs ID3, PRISM, and LEM2. Department of Computer Science, University of Kansas, TR-91-14, December 1991, 20
2. S. Greco, B. Matarazzo, and R. Slowinski: Dealing with missing data in rough set analysis of multi-attribute and multi-criteria decision problems. In Decision Making: Recent Developments and Worldwide Applications, ed. by S. H. Zanakis, G. Doukidis, and Z. Zopounidis, Kluwer, Dordrecht, 2000, 295–316
3. J. W. Grzymala-Busse: Knowledge acquisition under uncertainty – A rough set approach. *Journal of Intelligent and Robotic Systems* **1**, 1988, 3–16

4. J. W. Grzymala-Busse: On the unknown attribute values in learning from examples. Proc. of the ISMIS-91, 6th International Symposium on Methodologies for Intelligent Systems, Charlotte, North Carolina, October 16–19, 1991. Lecture Notes in Artificial Intelligence, vol. 542, Springer, Berlin Heidelberg New York, 1991, 368–377
5. J. W. Grzymala-Busse: 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, ed. by R. Slowinski, Kluwer, Dordrecht, 1992, 3–18
6. J. W. Grzymala-Busse. MLEM2: A new algorithm for rule induction from imperfect data. Proceedings of the 9th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, IPMU 2002, July 1–5, Annecy, France, 243–250
7. J. W. Grzymala-Busse. Rough set strategies to data with missing attribute values. Proceedings of the Workshop on Foundations and New Directions in Data Mining, Associated with the Third IEEE International Conference on Data Mining, Melbourne, FL, November 19–22, 2003, 56–63
8. J. W. Grzymala-Busse. Characteristic relations for incomplete data: A generalization of the indiscernibility relation. Proceedings of the RSCTC'2004, the Fourth International Conference on Rough Sets and Current Trends in Computing, Uppsala, Sweden, June 1–5, 2004. Lecture Notes in Artificial Intelligence 3066, Springer, Berlin Heidelberg New York, 2004, 244–253
9. J. W. Grzymala-Busse. Data with missing attribute values: Generalization of idiscernibility relation and rule induction. *Transactions on Rough Sets*, Lecture Notes in Computer Science Journal Subline, Springer Berlin Heidelberg New York, vol. 1, 2004, 78–95
10. J. W. Grzymala-Busse. Three approaches to missing attribute values – A rough set perspective. Proceedings of the Workshop on Foundation of Data Mining, associated with the 4th IEEE International Conference on Data Mining, Brighton, UK, November 1–4, 2004, 55–62
11. J. W. Grzymala-Busse and M. Hu. A comparison of several approaches to missing attribute values in data mining. Proceedings of the 2nd International Conference on Rough Sets and Current Trends in Computing RSCTC'2000, Banff, Canada, October 16–19, 2000, 340–347
12. J. W. Grzymala-Busse and S. Siddhaye. Rough set approaches to rule induction from incomplete data. Proceedings of the IPMU'2004, the 10th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, Perugia, Italy, July 4–9, 2004, vol. 2, 923–930
13. J. W. Grzymala-Busse and A. Y. Wang: Modified algorithms LEM1 and LEM2 for rule induction from data with missing attribute values. Proceedings of the 5th International Workshop on Rough Sets and Soft Computing (RSSC'97) at the 3rd Joint Conference on Information Sciences (JCIS'97), Research Triangle Park, NC, March 2–5, 1997, 69–72
14. M. Kryszkiewicz: Rough set approach to incomplete information systems. Proceedings of the 2nd Annual Joint Conference on Information Sciences, Wrightsville Beach, NC, September 28–October 1, 1995, 194–197
15. M. Kryszkiewicz: Rules in incomplete information systems. *Information Sciences* **113**, 1999, 271–292

16. T. Y. Lin: Neighborhood systems and approximation in database and knowledge base systems. 4th International Symposium on Methodologies of Intelligent Systems (Poster Sessions), Charlotte, North Carolina, October 12–14, 1989, 75–86
17. T. Y. Lin: Chinese wall security policy – An aggressive model. Proceedings of the 5th Aerospace Computer Security Application Conference, Tucson, Arizona, December 4–8, 1989, 286–293
18. T. Y. Lin: Topological and fuzzy rough sets. In Intelligent Decision Support. Handbook of Applications and Advances of the Rough Sets Theory, ed. by R. Slowinski, Kluwer, Dordrecht, 1992, 287–304
19. Z. Pawlak: Rough sets. *International Journal of Computer and Information Sciences* **11**, 1982, 341–356
20. Z. Pawlak: Rough Sets. Theoretical Aspects of Reasoning about Data. Kluwer, Dordrecht, 1991
21. J. R. Quinlan. C4.5: Programs for Machine Learning. Morgan Kaufmann, Los Altos, CA, 1993
22. R. Slowinski and D. Vanderpooten. A generalized definition of rough approximations based on similarity. *IEEE Transactions on Knowledge and Data Engineering* **12**, 2000, 331–336
23. J. Stefanowski: Algorithms of Decision Rule Induction in Data Mining. Poznan University of Technology Press, Poznan, Poland, 2001
24. J. Stefanowski and A. Tsoukias: On the extension of rough sets under incomplete information. Proceedings of the 7th International Workshop on New Directions in Rough Sets, Data Mining, and Granular-Soft Computing, RSFDGrC'1999, Ube, Yamaguchi, Japan, November 8–10, 1999, 73–81
25. J. Stefanowski and A. Tsoukias: Incomplete information tables and rough classification. *Computational Intelligence* **17**, 2001, 545–566
26. Y. Y. Yao: Two views of the theory of rough sets in finite universes. *International Journal of Approximate Reasoning* **15**, 1996, 291–317
27. Y. Y. Yao: Relational interpretations of neighborhood operators and rough set approximation operators. *Information Sciences* **111**, 1998, 239–259
28. Y. Y. Yao: On the generalizing rough set theory. Proceedings of the 9th International Conference on Rough Sets, Fuzzy Sets, Data Mining and Granular Computing (RSFDGrC'2003), Chongqing, China, October 19–22, 2003, 44–51

MLEM2 Rule Induction Algorithms: With and Without Merging Intervals

Jerzy W. Grzymala-Busse

Department of Electrical Engineering and Computer Science, University of Kansas, Lawrence, KS 66045–7621, USA
and
Institute of Computer Science, Polish Academy of Sciences, 01-237 Warsaw, Poland
jerzy@ku.edu

Summary. The MLEM2 algorithm is a rule induction algorithm in which rule induction, discretization, and handling missing attribute values are all conducted simultaneously. In this paper two versions of the MLEM2 algorithm are compared: the first version of MLEM2 induces rules that may contain two conditions with the same numerical attribute and different intervals. The second version of MLEM2 induces rules with merged conditions associated with numerical attributes, i.e., all conditions are related to different attributes. For completeness, experiments on the original LEM algorithm with discretization as a preprocessing are also included. The performance, in terms of accuracy, for all three algorithms is approximately the same (for any two of them the difference in performance is not statistically significant).

1 Introduction

The algorithm MLEM2 (Modified Learning from Examples Module, version 2) [7] is a component of the LERS (Learning from Examples based on Rough Sets) data mining system. Rough set theory was introduced in [13], see also [14]. MLEM2 is based on LEM2 (Learning from Examples Module, version 2). LEM2 requires a preprocessing called discretization, a conversion of numerical values into intervals. Additionally, LEM2 requires preprocessing to handling missing attribute values before the main process of rule induction. On the other hand, in MLEM2 all three processes: rule induction, discretization and handling missing attribute values are conducted at the same time, i.e., the MLEM2 module induces rule sets directly from data with numerical attributes and missing attribute values. Recently, a new version of MLEM2 was implemented, with merging conditions with intervals (for simplicity, we will call it MLEM2 with merging intervals).

The data mining system LERS uses a number of discretization algorithms. The simplest method to discretize a numerical attribute is partitioning its

J.W. Grzymala-Busse: *MLEM2 Rule Induction Algorithms: With and Without Merging Intervals*, Studies in Computational Intelligence (SCI) **118**, 153–164 (2008)
www.springerlink.com

domain into equal width intervals is called *Equal Interval Width Method*. Another method of discretization, called *Equal Frequency per Interval Method*, is based on dividing the domain of a numerical attribute into subsets with approximately equal relative frequencies of attribute values. In this method the discretized attribute entropy is maximum. In our experiments we used another discretization method, based on minimum entropy as a criterion to evaluate a list of best cutpoints. This method was used together with the original LEM2 algorithm as one of our three approaches for rule induction from numerical data. The two other approaches were two different versions of MLEM2: with and without merging intervals.

Our main objective was to compare performance of these three approaches for rule induction from numerical data. As was expected, the newest version of MLEM2 produces the smallest total number of conditions in rule sets. However, performance in terms of accuracy is approximately the same for all three approaches. Using different data sets than reported in this paper, MLEM2 with merging intervals was compared with two other approaches to rule induction from numerical data: discretization based on agglomerative and divisive cluster analysis and then LEM2 in [8].

Note that results of experiments comparing the quality of rule sets, induced by ID3, in terms of accuracy, with and without dropping conditions, were published in [10]. A preliminary version of this chapter was presented at the Workshop on Foundations of Semantic Oriented Data and WEB Mining, in conjunction with the ICDM'05, Fifth IEEE International Conference on Data Mining, Houston, TX, November 27–30, 2005.

2 Discretization Algorithm Based on Minimum Entropy

The discretization method, based on minimum entropy, was suggested in [3], and is also called *Minimal Class Entropy Method*. A similar process of discretization is used in C4.5 [15].

We will present our own discretization algorithm, introduced in [2]. Discretization is a conversion of domains of numerical attributes into intervals [6]. Such intervals are defined by cutpoints, numbers limiting the intervals. Let us say the set of all cases of a data set will be denoted by U. A single cutpoint q, a number from the domain of a numerical attribute a, defines two intervals, containing two subsets S_1 and S_2 of U. For an attribute a, the conditional entropy of a cutpoint q is

$$E(a, U, q) = \frac{|S_1|}{|U|} E(S_1) + \frac{|S_2|}{|U|} E(S_2),$$

where $E(S)$ is the entropy of a subset S of U. The entropy $E(S)$ is computed in the standard way as

$$-\sum_{j=1}^{n} p_j \; log \; p_j,$$

where p_j is defined as a relative frequency of a concept C_j, equal to

$$\frac{|S \cap C_j|}{|S|}.$$

For given attribute a, the cutpoint q for which $E(a, U, q)$ is minimal is the best cutpoint. In order to induce k intervals the above procedure is applied recursively $k - 1$ times. After determining the first cutpoint q that defines a partition of U into two sets S_1 and S_2, we compute $E(a, S_1, q_1)$ and $E(a, S_2, q_2)$ for two candidate cutpoints q_1 and q_2 for S_1 and S_2), respectively. Among q_1 and q_2, we select the cutpoint with the larger entropy. Thus, the worse of sets S_1 and S_2 is partitioned.

The discretization methods presented here can be classified as either *local* or *global* [2]. Local methods are characterized by operating on only one attribute, while global methods are characterized by considering *all* attributes (rather than one) before making a decision where to induce interval cutpoints.

In global discretization, first we select the best attribute and then, for the selected attribute, we select the best cutpoint. In our approach, see [2], the best attribute was selected on the basis of the following measure

$$\mathcal{M}_{\{A^D\}^*} = \frac{\sum_{S \in \{A^D\}^*} \frac{|S|}{|U|} E(S)}{|\{A^D\}^*|}$$

where $\{A^D\}^*$ is the partition induced by the discretized attribute A^D. A candidate attribute for which $\mathcal{M}_{\{A^D\}^*}$ is maximum is selected as for re-discretization. Obviously, we need only to re-compute this measure for an attribute which was last picked for re-discretization.

3 MLEM2

In general, LERS uses two different approaches to rule induction: one is used in machine learning, the other in knowledge acquisition. In machine learning, or more specifically, in learning from cases (examples), the usual task is to learn the smallest set of minimal rules, describing the concept. To accomplish this goal, LERS uses two algorithms: LEM1 and LEM2 (LEM1 and LEM2 stand for Learning from Examples Module, version 1 and 2, respectively) [4, 5].

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

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

where $[(a, v)]$ denotes the set of all examples such that for attribute a its values are v.

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:

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

The user may select an option of LEM2 with or without taking into account attribute priorities. The procedure LEM2 with attribute priorities is presented below. The option without taking into account priorities differs from the one presented below in the selection of a pair $t \in T(G)$ in the inner loop WHILE. When LEM2 is not to take attribute priorities into account, the first criterion is ignored. In our experiments all attribute priorities were equal to each other.

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 $t \in T(G)$ with
the highest attribute priority;
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.

MLEM2, a modified version of LEM2, processes numerical attributes differently than symbolic attributes. For numerical attributes MLEM2 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 q MLEM2 creates two blocks, the first block contains all cases for which values of the numerical attribute are smaller than q, the second block contains remaining cases, i.e., all cases for which values of the numerical attribute are larger than q. The search space of MLEM2 is the set of all blocks computed this way, together with blocks defined by symbolic attributes. Starting from that point, rule induction in MLEM2 is conducted the same way as in LEM2.

Additionally, the newest version of MLEM2, with merging intervals, at the very end simplifies rules by, as its name indicates, merging intervals for numerical attributes.

4 Classification System

Rules induced from raw, training data are used for classification of unseen, testing data. The classification system of LERS is a modification of the *bucket brigade algorithm* [1, 12]. The decision to which concept a case belongs to 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 matching rules with a larger number of attribute-value pairs are considered more specific. The third factor, *support*, is defined as follows

$$\sum_{matching\ rules\ R\ describing\ C} Strength\_factor(R) * Specificity\_factor(R).$$

The concept C for which the support is the largest is a winner and the case is classified as being a member of C.

In the classification system of LERS, if complete matching is impossible, all partially matching rules are identified. These are rules with at least one attribute-value pair matching the corresponding attribute-value pair of a case. For any partially matching rule R, the additional factor, called *Matching_factor* (R), is computed. Matching_factor (R) is defined as the ratio of the number of matched attribute-value pairs of R with a case to the total number

Table 1. An example of the data set

Case	Attributes		Decision hobby
	Age	Gender	
1	32	Female	Shooting
2	27	Male	Fishing
3	45	Male	Shooting
4	63	Female	Shooting
5	35	Male	Fishing

of attribute-value pairs of R. In partial matching, the concept C for which the following expression is the largest

$$\sum_{\substack{partially\ matching \\ rules\ R\ describing\ C}} Matching\_factor(R) * Strength\_factor(R) \\ * Specificity\_factor(R)$$

is the winner and the case is classified as being a member of C.

Every rule induced by LERS is preceded by three numbers: specificity, strength, and the rule domain size (the total number of training cases matching the left-hand side of the rule).

We will illustrate the MLEM2 algorithm with rule set induction from Table 1. The data set from Table 1 contains one numerical attribute *Age* and one symbolic attribute *Gender*. Additionally, there are two concepts: [(Hobby, shooting)] = {1, 3, 4,} and [(Hobby, fishing)] = {2, 5}.

First we need to sort the numerical attribute *Age*. The sorted list is: 27, 32, 35, 45, 63. The corresponding cutpoints determined by MLEM2 are: 29.5, 33.5, 40 and 54. Thus the set of all attribute-value pair blocks (the search space for MLEM2) is:

[(Age, 27..29.5)] = {2}
[(Age, 29.5..63)] = {1, 3, 4, 5}
[(Age, 27..33.5)] = {1, 2}
[(Age, 33.5..63)] = {3, 4, 5}
[(Age, 27..40)] = {1, 2, 5}
[(Age, 40..63)] = {3, 4}
[(Age, 27..54)] = {1, 2, 3, 5}
[(Age, 54..63)] = {4}
[(Gender, female)] = {1, 4}
[(Gender, male)] = {2, 3, 5}

Let us start from the set B equal to the concept [(Hobby, shooting)]. Thus $B = G = \{1, 3, 4\}$. The set $T(G)$ of all attribute-value pairs relevant to G consists of (Age, 29.5..63), (Age 27..33.5), (Age, 33.5..63), (Age, 27..40),

(Age, 40..63), (Age, 27..54), (Age, 54..63), (Gender, female) and (Gender, male). The most relevant attribute-value pair is (Age, 29.5..63), since among all attribute-value pairs from $T(G)$ the value of

$$|[(Attribute, value)] \cap G|$$

is the largest for (Age, 29.5..63). However,

$$[(Age, 29.5..63)] \not\subseteq (Hobby, shooting)].$$

Thus we have to start the second iteration of the inner *while* loop of the MLEM2 algorithm. This time $T(G)$ is equal to the set $T(G)$ that was initially computed except (Age, 29.5..63). Four attribute-value pairs: (Age, 33.5..63), (Age, 40..63), (Age, 27..54) and (Gender, female) are the most relevant. Since there is a tie, we have to use the second criterion to break the tie: the attribute-value pair with the smallest block cardinality. There are two candidates: (Age, 40..63) and (Gender, female). We have to use the third criterion: the first candidate, i.e., (Age, 40..63). Moreover, for the set T consisting of two attribute-value pairs computed so far: (Age, 29.5..63) and (Age, 40..63)

$$[T] \subseteq [(Hobby, shooting)].$$

The next step is to go through the loop *for* that follows the inner loop *while*. In different words, we will try to minimize set T. The first test is whether

$$[(Age, 40..63)] \subseteq [(Hobby, shooting)]$$

Since this is true, our final minimal complex is (Age, 40..63). Note that we used rule minimization, a part of the LEM2 algorithm that is common for both versions of MLEM2, with and without merging intervals, so both versions of the MLEM2 algorithm will induce the same rule. However,

$$[(Age, 40..63)] \neq [(Hobby, shooting)],$$

or, $G = B - [T] \neq \emptyset$, so we have to run the MLEM2 algorithm through the next iteration of its outer *while* loop. This time $G = \{1\}$ and $T(G) = \{$(Age, 29.5..63), (Age, 27..33.5), (Age, 27..40), (Age, 27..54), and (Gender, female)$\}$. Obviously, every member of $T(G)$ is the *most* relevant. The second criterion, the minimum of |[(Attribute, value)]| returns two candidates: (Age, 27..33.5) and (Gender, female). The third criterion returns (Age, 27..33.5). Additionally,

$$[(Age, 27..33.5)] \not\subseteq (Hobby, fishing)],$$

so we have to go through the second iteration of the inner *while* loop of the MLEM2 algorithm. T(G) is equal to {(Age, 29.5..63), (Age, 27..40), (Age,

27..54), and (Gender, female)}. The second criterion will indicate that (Gender, female) is the best candidate. Thus T = {(Age, 27..33.5), (Gender, female)}. Furthermore,

$$[T] \subseteq [(Hobby, shooting)].$$

We will execute the loop *for* to minimize T. After the first attempt we have

$$[(Gender, female)] \subseteq [(Hobby, shooting)]$$

hence (Gender, female) is our second minimal complex. Additionally, for $\mathcal{T}$ = {{(Age, 40..63)}, {(Gender, female)}}, we have

$$[(Age, 40..63)] \cup [(Gender, female)] = [(Hobby, shooting)],$$

so $\mathcal{T}$ is a local covering of [(Hobby, shooting)].

Our new input set to the algorithm MLEM2 is the other concept, i.e., the set {2, 5}. The set $T(G)$ of all attribute-value pairs relevant to G is {(Age, 27..29.5), (Age, 29.5..63), (Age 27..33.5), (Age, 33.5..63), (Age, 27..40), (Age, 27..54), (Gender, male)}. The most relevant attribute-value pairs are (Age, 27..40), (Age, 27..54), and (Gender, male). The second criterion, the minimum of |[(Attribute, value)]| does not break the tie since |[(Age, 27..40)]| = |[(Gender, male)]| = 3. The last resort is to select the first pair, i.e., (Age, 27..40). However,

$$[(Age, 27..40)] \not\subseteq [(Hobby, fishing)],$$

therefore we have to run the MLEM2 algorithm through the second iteration of the inner *while* loop. This time from $T(G)$ the attribute-value pair (Age, 27..40) is excluded, and the most relevant attribute-value pair is (Gender, male). Moreover, for $T = \{(Age, 27..40), (Gender, male)\}$ we have

$$T \subseteq [(Hobby, fishing)].$$

Furthermore, this set is already minimal and

$$[T] = [(Hobby, fishing)],$$

so our local covering for [(Hobby, fishing)] is the set containing as the only element T = {(Age, 27..40), (Gender, male)}. The rule set, determined by the MLEM2 algorithm, in the LERS format, is

1, 2, 2
(Age, 40..63) –> (Hobby, shooting)
1, 2, 2
(Gender, female) –> (Hobby, shooting)
2, 2, 2
(Age, 27..40) & (Gender, male) –> (Hobby, fishing)

5 Experiments

In our research we used the same data sets that were used for experiments in [9]. All of these data set have numerical attributes and are completely specified (i.e., for every attribute and every case the corresponding attribute value is specified). These ten data sets are presented in Table 2. For experiments we used three different approaches: the original LEM2 algorithm with discretization based on entropy as preprocessing, and two versions of MLEM2 algorithms. The first version of MLEM2 was not equipped with a mechanism for merging intervals within the same rule. For example, from *pima* data set, a typical induced rule was:

6, 38, 38
(Diabetes, 0.078..0.2995) & (Pressure, 57..122) &
(Diabetes, 0.1655..2.42) & (Age, 21..38.5) &
(Pressure, 0..83) & (Glucose, 0..99.5) –> (Class, 0)

It is clear that two conditions, both associated with the same attribute *Diabetes*, namely:

(Diabetes, 0.078..0.2995) and (Diabetes, 0.1655..2.42)

can be merged into one condition:

(Diabetes, 0.1655..0.2995).

Similarly, for attribute *Pressure*, the following two conditions

(Pressure, 57..122) and (Pressure, 0..83)

can be also merged into one condition:

Pressure, 57..83).

The third way to induce rules was the newest version of the MLEM2 algorithm that is able to merge conditions with intervals. We used results of

Table 2. Data sets

Data set	Number of		
	Cases	Attributes	Concepts
Bank	66	5	2
Bricks	216	10	2
Bupa	345	6	2
Buses	76	8	2
German	1,000	24	2
Glass	214	9	6
HSV	122	11	2
Iris	150	4	3
Pima	768	8	2
Segmentation	210	19	7

Table 3. Number of rules

Data set	Discretization based on entropy and LEM2	MLEM2 without merging conditions	MLEM2 with merging conditions
Bank	10	3	3
Bricks	25	12	12
Bupa	169	73	71
Buses	3	2	2
German	290	160	159
Glass	111	33	30
HSV	62	23	23
Iris	14	9	8
Pima	252	113	116
Segmentation	108	14	14

Table 4. Number of conditions

Data set	Discretization based on entropy and LEM2	MLEM2 without merging conditions	MLEM2 with merging conditions
Bank	13	5	6
Bricks	61	40	35
Bupa	501	345	241
Buses	4	4	5
German	1,226	1,044	814
Glass	262	137	87
HSV	206	101	80
Iris	33	23	17
Pima	895	599	428
Segmentation	322	48	37

experiments for the first two approaches that were reported in [9]. The same data sets were used for experiments with the newest version of the MLEM2 algorithm with merging of intervals. Results are presented in Tables 3–5. Table 3 presents the total number of rules for all three approaches while Table 4 shows the total number of conditions in those rule sets. Note that both versions of MLEM2 were independently implemented, using different heuristics, so the number of rules may differ. Furthermore, even the total number of conditions may be – for some data sets – larger for the algorithm that was supposed to induce rules with the smallest number of conditions.

Table 5 shows accuracy for all ten data sets and all three used approaches for rule induction. Accuracy was computed using tenfold cross validation.

Table 5. Accuracy

Data set	Discretization based on entropy and LEM2	MLEM2 without merging conditions	MLEM2 with merging conditions
Bank	97	95	95
Bricks	92	92	87
Bupa	66	65	64
Buses	99	96	93
German	74	70	69
Glass	67	72	69
HSV	56	60	65
Iris	97	95	94
Pima	74	71	70
Segmentation	64	89	84

6 Conclusions

To compare our three different approaches to rule induction from numerical data the Wilcoxon matched-pairs signed rank test was used (with level of significance 5%, two-tailed test) [11]. All three approaches were compared pair wise. The total number of rules is the largest for discretization based on entropy, used as preprocessing for original data sets, and then the LEM2 algorithm for rule induction. For both versions of MLEM2 the difference in performance is – statistically – insignificant.

Similarly, for the total number of conditions in the induced rule sets, the worst result – the largest number of conditions – was induced by the first approach: discretization based on entropy and then the LEM2 algorithm for rule induction. The performance of the MLEM2 algorithm with merging intervals was better than MLEM2 without merging intervals, as expected.

Surprisingly, all three approaches show no significant difference in performance for the most important parameter: accuracy.

References

1. Booker LB, Goldberg DE, and Holland JF (1990) Classifier systems and genetic algorithms. In: Carbonell JG (ed.) *Machine learning. Paradigms and methods.* MIT, Menlo Park, CA, 235–282
2. Chmielewski MR, Grzymala-Busse JW (1996) Global discretization of continuous attributes as preprocessing for machine learning, *International Journal of Approximate Reasoning* 15: 319–331
3. Fayyad UM, Irani KB (1992) On the handling of continuous-valued attributes in decision tree generation, *Machine Learning* 8: 87–102
4. Grzymala-Busse JW (1992) LERS – A system for learning from examples based on rough sets. In: Slowinski R (ed) *Intelligent decision support. Handbook of applications and advances of the rough set theory.* Kluwer, Dordrecht, 3–18

5. Grzymala-Busse JW (1997) A new version of the rule induction system LERS, *Fundamenta Informaticae* 31: 27–39
6. Grzymala-Busse JW (2002) Discretization of numerical attributes. In: Kloesgen W, Zytkow J (eds) *Handbook of data mining and knowledge discovery*, Oxford University Press, New York, 218–225
7. Grzymala-Busse JW (2002) MLEM2: A new algorithm for rule induction from imperfect data. *Proceedings of the 9th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, IPMU 2002*, July 1–5, Annecy, France, 243–250
8. Grzymala-Busse JW (2004) Three strategies to rule induction from data with numerical attributes, *Transactions on Rough Sets*, Lecture Notes in Computer Science Journal Subline, Springer, Berlin Heidelberg New York 2: 54–62
9. Grzymala-Busse JW, Stefanowski J (2001) Three discretization methods for rule induction, *International Journal of Intelligent Systems* 16: 29–38
10. Grzymala-Busse JW, Hsiao TY (1998) Dropping conditions in rules induced by ID3. *Proceedings of the 6th International Workshop on Rough Sets, Data Mining and Granular Computing RSDMGrC'98 at the 4th Joint Conference on Information Sciences (JCIS'98)*, October 1998, Research Triangle Park, NC, 351–354
11. Hamburg M (1983) *Statistical Analysis for Decision Making*. Harcourt Brace Jovanovich, New York 546–550 and 721
12. Holland JH, Holyoak KJ, Nisbett RE (1986) *Induction. Processes of inference, learning, and discovery*. MIT, Boston
13. Pawlak Z (1982) Rough sets, *International Journal of Computer and Information Sciences* 11: 341–356
14. Pawlak Z (1991) *Rough Sets. Theoretical Aspects of Reasoning about Data*. Kluwer, Dordrecht
15. Quinlan JR (1993) *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Francisco, CA

Towards a Methodology for Data Mining Project Development: The Importance of Abstraction

P. González-Aranda[1], E. Menasalvas[1], S. Millán[2], Carlos Ruiz[1], and J. Segovia[1]

[1] Universidad Politecnica de Madrid, Madrid, Spain
pgonzalez@fi.upm.es, emenasalvas@fi.upm.es, cruiz@fi.upm.es, fsegovia@fi.upm.es
[2] Universidad del Valle. Cali, Colombia
millan@eisc.univalle.edu.co

Summary. Standards such as CRISP-DM, SEMMA, PMML, are making data mining processes easier. Nevertheless, up to date, projects are being developed more as an art than as a science making it difficult to understand, evaluate and compare results as there is no standard methodology. In this chapter, we make a proposal for such a methodology based on RUP and CRISP-DM and concentrate on the project conception phase for determining a feasible project plan.

1 Introduction

The success of Business Intelligence is usually related to a Data Mining Project. Several common pitfalls in a Business Intelligence project have been identified in [21]: lack of understanding of the complexity of the project, lack of recognizing Business Intelligence project as a cross-organizational business initiative, no iterative development method, no business analysis and no standardization activities, inappropriate project team structure and dynamics and too much reliance on disparate methods and tools. All of them being summarized as a lack of methodology for project development.

The question that arises is which methodology should be used when developing data mining projects. Unfortunately, despite the years of data mining experience in companies the answer to that question is that there is no standard methodology. In fact (see [25]) a 2004 report sets CRISP-DM as the most frequently used in 42% of companies interviewed followed by companies using their own methodology (28%).

However, defining a methodology requires first of all, knowing the activities to be developed. These have been described in CRISP-DM the standard of the Data Mining process model. But besides the activities, the lifecycle of the

P. González-Aranda et al.: *Towards a Methodology for Data Mining Project Development: The Importance of Abstraction*, Studies in Computational Intelligence (SCI) **118**, 165–178 (2008)
www.springerlink.com

project, and deliverables have to be defined. Although deliverables have been considered in CRISP-DM, they have been informally defined and consequently no comparison of output and nonformal evaluation are possible. Regarding the lifecycle, in this chapter we propose a first approach to Data Mining project phases and lifecycle based on the ones defined by RUP. We also propose a first approach towards abstraction for the project conception step.

The rest of the chapter has been organized as follows. Section 2 presents related work and review advances in Data Mining standardization and approaches to data mining methodology. We also review RUP as representative of software development methodologies. In Sect. 3 we define a data mining project. In particular we make a first approach to the definition of steps and lifecycle. In Sect. 4 we focus on the first phase of a Data Mining project, namely what we define as the project conception, to properly define a data mining project plan. Section 5 presents preliminary conclusions and outlook.

2 Related Work

In the information age when data generated and stored by modern organizations increase in an extraordinary way, data mining tasks [9] become a necessary and fundamental technology. A lot of data mining research has been focusing on the development of algorithms for performing different tasks, i.e. clustering, association and classification [1, 2, 5, 13, 15, 16, 19, 20, 24, 28, 30], and on their applications to diverse domains. One major challenge in data mining, according to [12], is getting researchers to agree on a common standard for preprocessing tasks and standards related to applying the data mining process to operational processes and systems. In this sense, the Predictive Model Markup Language (PMML) [8] provides several components (Data Dictionary, Mining Schema, Transformation Dictionary, Models) useful for producing data mining models. The Data Dictionary includes only information about type of data and range of values. Semantic information is not taken into account.

Several proposals have been developed in order to offer a guide for implementing data mining projects [7, 22, 27].

The Common Warehouse Model for Data Mining (CWM DM) [22] proposed by the Object Management Group, introduces a CWM Data Mining metamodel integrated by the following conceptual areas: a core Mining metamodel and metamodels representing the data mining subdomains of Clustering, Association Rules, Supervised, Classification, Approximation, and Attribute Importance.

The Cross-Industry Standard Process for Data Mining (CRISP-DM), was proposed in 1997 [7] in order to establish the standard data mining process. CRISP-DM steps include several processes:

- Business Understanding focuses on understanding the project objectives and requirements from the business perspective, then converting this

knowledge into data mining problem definition and a preliminary plan to achieve the objectives.

- Data Understanding starts with an initial data collection and proceeds with activities in order to get familiar with the data, to identify data quality problems, to discover first insights into the data or to detect interesting subsets to propose hypotheses for hidden information.
- Data Preparation constructs the final dataset from the initial raw data. Data preparation tasks are likely to be performed multiple times and not in any prescribed order. Tasks include table, record and attribute selection as well as transformation and cleaning of data for modelling tools.
- Modelling techniques are selected and applied and their parameters are calibrated to optimal values. There are several techniques for the same data mining problem type that have some different data requirements.
- Evaluation evaluates the model and review the steps executed to construct the model to be certain it properly achieves the business objectives.
- Deployment presents the knowledge in a way that the customer can use it. It often involves applying models within an organization's decision making processes.

At 1999 SAS Institute proposed the SEMMA [27] methodology integrated by five phases: Sample, Explore, Modify, Model and Assess. The data mining process starts by taking a representative sample of the target population to which a confidence level is associated. Then, this sample is explored and analyzed using visualization and statistical tools in order to obtain a set of significant variables that will become the input for a selected model. The selected model is analyzed. The goal of this step is to determine relationships among variables. In this phase, both statistical methods (e.g. discriminant analysis, clustering, and regression analysis) and data-oriented methods (e.g. neural networks, decision trees, association rules) can be used. The final phase in this process consists of evaluating the model and comparing it with different statistical methods and samples. On the other hand, Clementine proposes CATs [29] (Clementine Application Templates) as application specific libraries that follow the CRISP-DM standard, being each CAT stream assigned to a CRISP-DM phase.

All of the above models depend heavily on the analysts (business, domain experts, data miners) knowledge. There seems to exist a need for an intermediate level of conceptualization which can provide an interface between the experts and the clients.

According to Grossman et al. [12] "although efforts have been done to homogenize terminology and concepts among standards more work is required". A framework to develop a unified model for data mining is proposed in [10]. The goal of the model is to provide a uniform data structure for all data mining patterns and operators to manipulate them. The model is designed under a three-view architecture (Process view, model view and data view) that includes a process model and data views. The model view contains a set

of mining models with information about mining results. All these approaches and standards do not take the semantics of the data into account.

In [21] a very good approach of the advantages and disadvantages of traditional methodologies of software development when applied to Business Intelligence solutions is found. Here the authors state how old practices are good when every system had a beginning and an end and every system was designed to solve only one isolated problem for one set of business people from one line of business. However, this practices fail when integration of different departments is needed, because they do not include any cross-organizational activities necessary to sustain an enterprise-wide decision support environment. For nonintegrated system development, conventional waterfall methodology is sufficient. However, these traditional methodologies do not cover strategic planning cross-organizational business analysis. Software methodologies such as an iterative model had to be improve to deal with risks.

In RUP [17] an architecture centered model is presented in which an iterative and incremental way makes it possible to develop a software product of any scale or size. Outputs of each iteration can be components, modules of any software part that will be integrated into the next iteration in order to fulfil the final product at the end. These features make it appropriate for Data Mining projects in which requirements change as a consequence of already obtained patterns and where the outputs (patterns) of each step integrate the global solution.

3 Basis of a Data Mining Project Development Methodology

The term business refer to any activity developed in a company in the most general sense, no matter the nature and aim of such activity (commercial, governmental, education, ...). Data mining is one of the technologies that make Business Intelligence solutions [6] be implemented ("a fairly new term that incorporates a broad variety of processes and technologies to harvest and analyze specific information to help a business make sound decisions"). In fact, any business intelligence solution should include a data mining project to extract "the intelligence" of the business that will be accordingly deployed.

However, the truth is that data mining projects are being developed more as an art than as an engineering process. It does not properly meet real business needs when dealing with any kind of project. Companies really need to manage projects in the most controlled way, always trying to reduce risks without increasing costs. As there is no proper methodology to face data mining projects, several different practices from different areas are applied. This leads to failures when developing a project to getting poor results, or at least not as good as they could be.

The need for a proper method to manage data mining projects is thus clear. This method should allow managers to identify tasks and subtasks,

roles, risks, milestones as well as estimating costs, benefits, taking into account the cross-dependant nature of all the elements. Efforts presented in Sect. 2 towards a methodology have generated a "good manners" guide as well as a definition of the technical activities to be developed in the process of knowledge discovery. However, no clear result towards the management of the global process have been obtained up to date. Higher risk levels, greater efforts and higher associated costs with each task being developed is the result of a lack of understanding of the project development process due mainly to a lack of methodology. Methodologies make the development team concentrate their efforts in tasks to be developed, clearly defining roles and assignment for each participant making organization and project development a lot easier. Consequently, prior to defining the methodology, the terms we are refereing to have to be clearly understood and defined.

A project has been defined as any piece of work that is undertaken or attempted. Consequently, project management involves "the application of knowledge, skills, tools and techniques to a broad range of activities to meet the requirements of the particular project" [3]. Project management is needed to organize the process of development and to produce a project plan. The way the process is going to be developed (life cycle) and how it will be split into phases and tasks (process model), will be established. This project definition [23] exactly describes the common understanding, its extent and nature, among the key people involved in a project. Thus, any data mining project need to be defined to state the parties, goals, data and human resources, tasks, schedules, expected results, that comply the foundation upon which a successful project will be built. In general, any engineering project iterates through the following stages between inception and implementation: Justification and motivation of the project, Planning and Business Analysis, Design, Construction, Deployment. In fact in software engineering this approach has been successfully applied. Although a data mining project has components similar to those found in IT, the nature is different even some that concepts need to be modified in order to be integrated.

In fact the proposal that we make here is inspired on concepts from RUP, taking as technical tasks the ones defined in the CRISP-DM process model.

For a proper definition of the methodology, phases that will lead the project have to be defined. Phases will have iterations with intermediate products and the end of a phase will lead a deliverable. The set of activities (in our case from CRISP-DM) and the effort dedicated to them in each project phase will have to be defined. Depending on the activities involved in each phase,the roles of the team and the associated effort will have to be defined.

Figure 1 depicts development phases in the proposed methodology for data mining projects. X axis represents phases while Y axis represents the involved processes. For each phase, efforts dedicated to each activity of the process model have been represented. In each phase more than one iteration can occur and each of them may lead to an intermediate product. A phase will end having as a result deliverables. Intermediate products and deliverables will help to establish milestones in the project development plan.

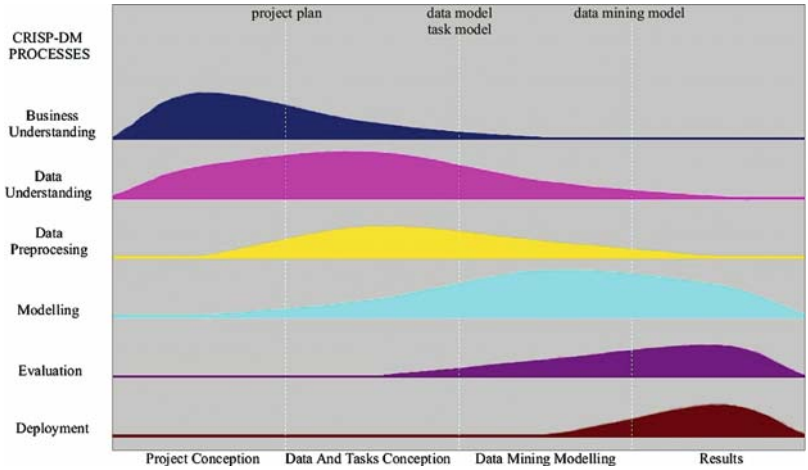


Fig. 1. Phases of a data mining methodology

Defining the proposed phases of the methodology, the different stages of RUP methodology have been taken into account. One of the main goals of the proposed methodology is to establish the activities to be carried out as well as their timing for its successful ending while preserving flexibility in the process.

The proposed phases are briefly described below:

- Project Conception establishes the main topics in the project. In order to develop a proper project plan information about business goals, data sources, risks and contingencies plans, costs and benefits, estimations and schedules, resources and results need to be gathered. A complete project plan is fundamental to achieve a proper and successful data mining project because the information reflected will be used to complete project's life cycle. Figure 1 depicts main activities in this phase, Business Understanding, Data Understanding and Data Preparation, that help data miners define business goals and data sources. This definition is basic to develop a data mining project and will be used in later phases.
- Data And Tasks Conception. The Data Model defines all the data sources and extraction, transformation, loading and integration processes involved in a data mining project. In order to define them in a formal way some metamodel must be defined and used. The Task Model defines all the data mining tasks to be done in the project. The approach here is that a task model is first defined in terms of types of problems (e.g. clustering instead of K-means, association instead of a priori, ...) and then refined in some iterations by a data mining expert. The Task Model uses the Data Model to establish the data involved in each data mining task. Considering these models, the main activities involved are Data Understanding and Preprocessing and Modelling.

- Data Mining Modelling applies data mining techniques from tasks defined in the task model and choose data sources from the data model. At this point, the task model is refined from problem types to algorithms. Some procedures to measure the satisfaction of possible algorithms is needed.
- Results establish visualization and evaluation in terms of business goals and customer satisfaction. Thus, some mechanisms must be defined in order to adequate the solution to customer needs. Sometimes this phase supposes a software project to deploy the knowledge obtained. This is a very important phase because our customer must validate the results of the project.

4 Conception of a Data Mining project

Data mining experts have made the process of translating business goals into data mining goals, automatic. When doing so, the expert does not only take into account data mining techniques but also their constraints, inputs, outputs, the order in which algorithms will be applied and dependencies between inputs and outputs. As part of the process, the expert automatically evaluates different choices depending on the intermediate results and/or inputs. The quality of the overall process will finally determine the quality of the obtained results.

However, the main goal for a company should be to make this process explicit: to generate a method to perform the required tasks in a systematic way. Ensuring the automatic generation of feasibility plans for each business goal being translated into data mining goals no matter who the person in charge of the process may be. A first step towards this method will be the definition of certain mechanisms of abstraction to obtain a model of the objectives of the project. Which is the method to be followed so that business objectives can be translated into data mining objectives? Unluckily, there is no such methodology but if we think on how to obtain it new questions will arise: How is a business objective expressed? Do we have any standard to express business objectives in a uniform way? What is a data mining goal? How are data mining goals achieved? Which are the requirements of data mining functions? Do we have a standard to establish data mining goals?

4.1 Setting Objectives

The bottom line to business success is to increase the knowledge of decision makers at every level of an organization. The process of knowledge creation and enhancement comes from information which is nothing else than data that have been collected, accessed, formatted and analyzed [18].

A data mining project arises when a given organization needs to solve a set of problems that can be addressed by means of specific knowledge discovery techniques. Independently of how good the data mining techniques can

be, a project whose requirements are poorly specified will end up with a disappointed end user [26]. In a data mining project the most critical factor is related to the clear understanding of the business goals.

Both the client and the data miner play an important role when establishing business goals. The client has to formulate his problem while the data miner tries to understand it in order to be able to translate it into data mining functions. During this task, it is relevant to keep in mind the following aspect (inspired from software engineering [26]): "Business domain as well as functional domain of the problem has to be represented and understood".

Deeply analyzing any activity of the organization (even external to it) that generate data that will be potentially used as input in a data mining project as well as the data themselves and the data mining functions will highlight important concepts that are common in any data mining project independently of the domain. However, eliciting, analyzing and graphically depicting concepts is no easy task [11] and must be developed based on a systematic method that provide all those elements that make this process less risky but more controlled.

Hence, first of all it is necessary to set the basis for a definition of elements that will make it possible to represent or abstract the business domain that is the target of the project. The goal of this abstraction is to provide a data mining project manager with a method to systematically describe the goals of the project. Moreover, once the goals are understood, they can be translated into data mining goals and then, into data mining problem types. However, not only identifying the data mining problems to be solved is enough. We should also be able to find out if the available data to be analyzed fulfil a set of general requirements or conditions. Every problem type will require different kinds of data. In the following, we will describe the different existing issues as well as their requirements.

Data Requirements Assessment

Every data mining project can be collectively described as data analysis and knowledge extraction to obtain the intelligence of the business. Data do not only represent the activities or business processes that have generated them but implicitly carry important hidden knowledge about the business. Extracting this knowledge means making decisions in an intelligent and successful way.

Though talking about intelligence, data mining does not involve deductive processes. On the contrary, it is an inductive process that analyzes the data to extract knowledge: it accepts data from different sources, manipulates them and obtains an output and patterns of knowledge, that if of good quality, will be deployed. This is the general setting of the process no matter the domain or organization we are dealing with.

In the process of data analysis and knowledge extraction, different perspectives of the data being analyzed are taken into account: data sources,

information content of the data (knowledge to explain the data), data structure and data flow. To fully understand the process, all of them must be considered.

Related to the content, data have to be enriched so that goals can be fulfilled. Data that are the source of a data mining project were never designed, captured and stored thinking they would become the input of the data mining process. Consequently, an effort is needed to transform them so that knowledge can be extracted. Any element that can be determinant when making a decision should be analyzed. In this sense, the major problem is establishing these elements as it is as equally fatal to leave one element out as it is to introduce erroneous elements. In any case, elements that can be decisive when making decisions have to do with the operations developed in the company, the internal organization of the company as well as business rules, and finally the external conditions related to the business (competitors) and general (political, social, . . . ,) events.

In order for the process to be data miner and client independent, this is to say, to be able to obtain the same goals no matter who the experts leading and developing the process are, a systematic abstract way to express this content information is needed. The first naïve approach is to conceptualize this information to discover the concepts and properties the available data represent with respect to the business. Only this way, we should be able to establish which are the business elements involved and consequently to asses whether the data comply with the requirements of each function within the data mining process.

Understanding the Data Mining Domain

As it was stated before, every data mining problem type will require different kinds of data. Depending on the project objectives different techniques should be applied and consequently, different kinds of data will be required. In order to be able to systematically determine whether project goals are feasible or not it will be necessary to automate the process of extracting data requirements from data mining problem types. In the following, we will describe the existing data mining problem types as well as their inner data requirements.

Several data mining problems classifications can be found in the literature. In [7] authors describe six kinds of problems: data description and summarization, segmentation, concept description, classification, prediction and dependency analysis. Usually the data mining project involves different problem types that together will achieve the goals of the project. In other words, information extracted from different analysis tasks must be integrated into a piece of knowledge that fits project goals.

In [4, 14] the various types of data mining algorithms such as memory-based reasoning, link analysis, decision trees, neural networks, ... are explained. Data mining common tasks are identified: classification, estimation,

prediction, affinity grouping, clustering and description. Moreover, they explain which data mining techniques are more appropriate for every type of problem.

Although in [4] which data-mining techniques are best for what types of business applications is stated, it starts with data mining objectives already identified. A further description of the problem, so that a mapping could be done between business objectives and data mining problems, is missing. This mapping will help on the one hand to see if certain business objectives are feasible or not, and, on the other, it would provide means to interpret the patterns to be obtained.

Results cannot be interpreted depending only on the function or/and technique used to obtain it. After applying any data mining function, lets take clustering as an example, a set of patterns is obtained but to evaluate their quality and consequently the success of the process, not only measures related to the patterns, clusters in this case, (number of elements, cohesion, ...) are needed but also some values to measure the results according to user expectations. The latter are a mixture of understanding the meaning of each pattern, cluster, together with the business requirements.

Hence, data mining problems cannot be analyzed to abstract common features on their own. There is a need for deeply analyzing data mining problems to fully understand its requirements at conceptual, technical and data levels. Once this work is done, the conceptualization of the data mining problems will serve the data mining project managers to define a project plan as well as to provide a basis for understanding and evaluating the results.

4.2 Planning the Project

As explained before, the focus of the article is to face the issue of how to ensure the systematic generation of rational, feasibility and controlled data mining project plans no matter who the person in charge of the process may be. The definition of certain abstraction mechanisms is a first step towards it. The question arising now, when those domains that have to be abstracted and engaged have been clearly stated, is: which are the goal elements (tools, models, documents, ...) of the abstraction? Previous to the definition of such a systematic method, there is a need to find a standard way to represent all the elements identified as relevant in the business domain to be analyzed.

Figure 2 depicts the basic steps, tools and intermediate results that underline the establishment of a systematic method to define data mining goals.

Mandatory elements that compose this information are: objectives and motivations underlying the project, scope of application of the expected results and structure, content and flow of the data to be analyzed. Besides, technical elements related to the very nature of the data mining project will have to be incorporated to the previous information. The blending and abstraction of these two pieces of information, will result not only in a model of shared

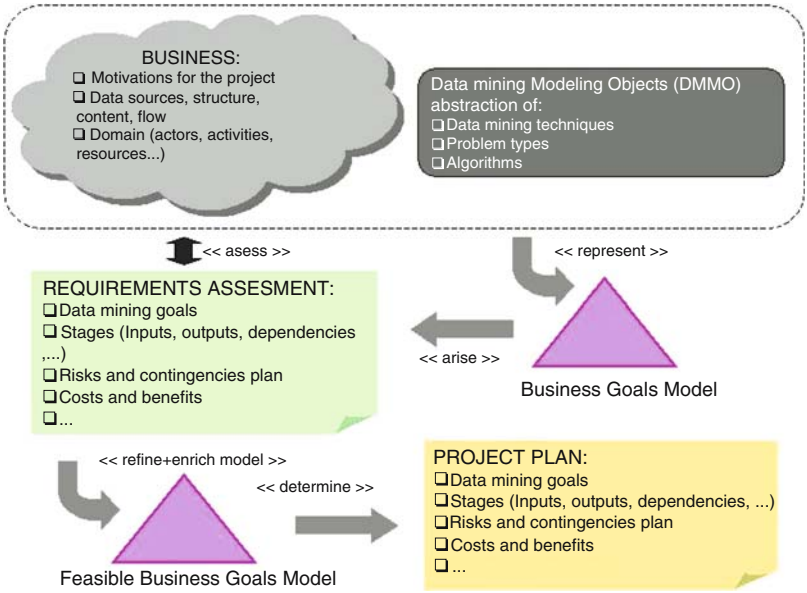


Fig. 2. Project plan definition process

understanding for client and data miner but will also be a tool to determine potential data mining goals and consequently the basis to project plans.

In Fig. 2, DMMO (Data Mining Modelling Objects) denotes the set of the compounding elements of the Modelling Language. Elements of the business domain will be abstracted using DMMO, generating the Business Objective Model. Together with this intermediate model, a document that we have called Requirements Assessment will be produced. This is a first approach to project goals in which special attention is given to critical factors (risks, constraints, information required, ...) that will depend both on the goals themselves and on the tools and techniques used to achieve them.

The requirement assessment document will be analyzed jointly with the client to enrich and refine the previous business goals model. The resulting model from this analysis is what has been called Feasible Business goals Model in the figure. In this model, goals previously identified but analyzed as not feasible have been not included, although not removed as these are conclusion of the projet by themselves. From the refined model and making use once again of DMMO the project plan will be produced almost in a automatic way as the model will represent relevant aspects both of the domain and of the tools themselves. Due to the assumed abstraction capability of DMMO, the plan will contain detail information about: techniques, tools, kind of data mining to be solved, inputs, outputs, flow of data and dependencies. Thus, risk and contingencies, cost, milestones, will be identified.

5 Discussion and Conclusion

The main reason for data mining to be developed more as and art than as a science can be found in the lack of a methodology for data mining project development. In this chapter, we have presented a first approach to such methodology and we have focused on the first phase, project conception, as the basis for the methodology. A first step towards the systematization of data mining project development is the definition of certain abstraction mechanisms to capture the project goals as well as to define how to reach them. The objective of this abstraction is to provide the manager with a method to describe goals in terms of data mining. This will help the later planning, managing and developing processes involved in every project. Deeply analyzing the target domains of data mining tasks (structure, data generation processes and data themselves) will help to identify shared concepts to every data mining project no matter what the nature of the domain could be. This way it will be possible to set the basis for defining elements to represent business goals and therefore fully find answer, to questions such as: what is intended to do?, which are the target elements to be analyzed?, what are the techniques to be applied? or which requirements must fulfil the data?

In this chapter we have also presented a first approach to a global methodology for managing data mining projects based on RUP. Steps of the methodology have been established as well as deliverables to be obtained along the process. Technical tasks of the methodology have been taken from CRISP-DM. Nevertheless, to clearly define the methodology, deliverables have to be precisely defined and an abstraction mechanism has to be found to express deliverables in a standard way. On the other hand, activities related to the management tasks have also to be defined.

Acknowledgments

The research has been partially supported by Ministerio de Educacion y Ciencia (project TIN2004-05873).

References

1. A. Chidanand and S. Weiss. Data mining with decision trees and decision rules. *Future Generation Computer Systems*, 13(2–3):197–210, 1997
2. R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A. Verkamo. Fast discovery of association rules. *In U. Fayyad et al. Advances in Knowledge Discovery and Data Mining*. MIT, Cambridge, MA, 1996
3. American Society for Quality. Six Sigma Forum. `http://www.asq.org/info/glossary/p.html`, last accessed 2005
4. M. Berry and G. Linoff. *Data Mining Techniques for Marketing, Sales and Customer Support*. Wiley, New York, 1998

5. S. Brin, R. Motwani, J.D. Ullman, and S. Tsur. Dynamic itemset counting and implication rules for market basket data. In *SIGMOD '97: Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data*, ACM, New York, NY, 255–264, 1997
6. Commerce-Database.com. Business Intelligence Definition. `http://www.commerce-database.com/business-intelligence.htm`, last accessed 2005
7. CRISP-DM Consortium. *CRISP-DM 1.0. Step-by-step data mining guide*, 1.0 edition, August 2000
8. Data Mining Group. The Predictive Model Markup Language PMML. `http://www.dmg.org`, last accessed 2005
9. U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From data mining to knowledge discovery: An overview. *In U. Fayyad et al. Advances in Knowledge Discovery and Data Mining* 1–34. MIT, Cambridge, MA, Chapter 1, 1996
10. I. Geist. A framework for data mining and kdd. In *SAC '02: Proceedings of the 2002 ACM Symposium on Applied Computing*, ACM, New York, NY, USA, 508–513, 2002
11. K. Mc Graw and K. Harbison-Briggs. *Knowledge Acquisition: Principles and Guidelines*. McGraw-Hill, New York, 1986
12. R. Grossman, M. Hornick, and G. Meyer. Data Mining Standards Initiatives. *Communication of ACM, August 2002, Vol. 45 No. 8 pp. 59–61*, 2002
13. S. Guha, R. Rastogi, and K. Shim. CURE: an efficient clustering algorithm for large databases. In *ACM SIGMOD International Conference on Management of Data*, 73–84, June 1998
14. J. Han and M. Kamber. *Data Mining: Concepts and Techniques*, 550. Morgan Kaufmann, Los Altos, CA, August 2000
15. J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In W. Chen, J. Naughton, and P.A. Bernstein, editors, *2000 ACM SIGMOD International Conference on Management of Data*, 1–12. ACM, 05 2000
16. M. Kamber, L. Winstone, W. Gong, S. Cheng, and J. Han. Generalization and decision tree induction: Efficient classification in data mining. In *Proceedings of the 1997 International Workshop Research Issues on Data Engineering (RIDE'97)*, Birmingham, England, 111–120, April 1997
17. P. Kruchten. *The Rational Unified Process: An Introduction*. Addison-Wesley, Reading, MA, 2004
18. S. Kudyba. Data Mining Efforts Increase Business Productivity and Efficiency. *Interview with Stephan Kudyba – President of Null Sigma Inc.*, 2001
19. T.Y. Lin and E. Louie. Data mining using granular computing: fast algorithms for finding association rules. In *Data Mining, Rough Sets and Granular Computing*, 23–45. Physica, Heidelberg, 2002
20. M. Mehta, R. Agrawal, and J. Rissanen. Sliq: A fast scalable classifier for data mining. In *Proceedings of International Conference on Extending Database Technology*, 18–32, 1996
21. L.T. Moss and S. Atre. *Business Intelligence Roadmap. The Complete Project Lifecycle for Decision-Support Applications*. Addison-Wesley Information Technology Series, 2004
22. Object Management Group. Common Warehouse Metamodel – Data Mining. `http://www.omg.org/cgi-bin/doc?ad/00-01-01`, March last accessed 2005

23. University of Washington. Project Definition in Project Management. http://www.washington.edu/computing/pm/define/definition.html, last accessed 2005
24. Z. Pawlak. Information systems: theoretical foundations. *Information Systems*, 6(3):205–218, 1981
25. G. Piatesky-Shapiro. Data Mining, Web Mining, and Knowledge Discovery Guide. http://www.kdnuggets.com, 2005
26. R.S. Pressman. *Software Engineering: A Practioner's Approach.* McGraw-Hill, New York, 1997
27. SAS. SEMMA – Sample, Explore, Modify, Model, Assess. http://www.sas.com/technologies/analytics/datamining/miner/semma.html, last accessed 2005
28. D. Slezak, J. Wroblewski, and M.S. Szczuka. Constructing extensions of bayesian classifiers with use of normalizing neural networks. In *Foundations of Intelligent Systems, 14th International Symposium, ISMIS 2003, Maebashi City, Japan, October 28–31, 2003, Proceedings*, volume 2871 of *Lecture Notes in Computer Science*, 408–416. Springer, Berlin Heidelberg New York, 2003
29. SPSS Corporation. CAT (Clementine Application Templates). http://www.spss.com/clementine/cats.htm, last accessed 2005
30. W. Ziarko. Variable precision Rough Set Model. *Journal of Computer Systems and Science*, 46(1):39–59, 1993

Fining Active Membership Functions in Fuzzy Data Mining

Tzung-Pei Hong[1], Chun-Hao Chen[2], Yu-Lung Wu[3], and Vincent S. Tseng[4]

[1] Department of Electrical Engineering, National University of Kaohsiung, Kaohsiung, Taiwan, ROC
tphong@nuk.edu.tw
[2] Department of Computer Science and Information Engineering, National Cheng-Kung University, Tainan, Taiwan, ROC
chchen@idb.csie.edu.tw
[3] Department of Information Management, I-Shou University, Kaohsiung, Taiwan, ROC
wuyulung@isu.edu.tw
[4] Department of Computer Science and Information Engineering, National Cheng-Kung University, Tainan, Taiwan, ROC
vincent@idb.csie.ncku.edu.tw

Summary. This chapter proposes a fuzzy data-mining algorithm for extracting both association rules and membership functions from quantitative transactions. The number of membership functions for each item is not predefined, but can be dynamically adjusted. A GA-based framework for finding membership functions suitable for mining problems is proposed. The encoding of each individual is divided into two parts. The control genes are encoded into bit strings and used to determine whether membership functions are active or not. The parametric genes are encoded into real-number strings to represent membership functions of linguistic terms. The fitness of each set of membership functions is evaluated using the fuzzy-supports of the linguistic terms in the large 1-itemsets and the suitability of the derived membership functions. The suitability of membership functions considers overlap, coverage and usage factors.

1 Introduction

Data mining is most commonly used in attempts to induce association rules from transaction data. Transaction data in real-world applications, however, usually consist of quantitative values. Designing a sophisticated data-mining algorithm able to deal with various types of data presents a challenge to workers in this research field.

Recently, fuzzy set theory has been used more and more frequently in intelligent systems because of its simplicity and similarity to human reasoning.

T.-P. Hong et al.: *Fining Active Membership Functions in Fuzzy Data Mining*, Studies in Computational Intelligence (SCI) **118**, 179–196 (2008)
www.springerlink.com

In [4], we proposed a mining approach that integrated fuzzy-set concepts with the a priori mining algorithm [1] to find interesting itemsets and fuzzy association rules in transaction data with quantitative values. In that paper, the membership functions were assumed to be known in advance. The given membership functions may, however, have a critical influence on the final mining results. This chapter thus modifies the previous algorithm and proposes a new fuzzy data-mining algorithm for extracting both association rules and membership functions from quantitative transactions.

In the past, Srikant and Agrawal proposed a mining method [7] to handle quantitative transactions by partitioning the possible values of each attribute. Hong et al. proposed a fuzzy mining algorithm to mine fuzzy rules from quantitative data [4]. They transformed each quantitative item into a fuzzy set and used fuzzy operations to find fuzzy rules. Wang and Bridges used GAs to tune membership functions for intrusion detection systems based on similarity of association rules [11]. Kaya and Alhajj [6] proposed a GA-based clustering method to derive a predefined number of membership functions for getting a maximum profit within an interval of user specified minimum support values. In this chapter, we will try to derive an unknown number of membership functions from quantitative transactions by using a divide-and-conquer genetic strategy.

2 A GA-Based Mining Framework

In this section, the fuzzy and GA concepts are used to discover both useful association rules and suitable membership functions from quantitative values. A GA-based framework for achieving this purpose is proposed in Fig. 1.

The proposed framework is divided into two phases: mining membership functions and mining fuzzy association rules. Assume the number of items is m. In the phase of mining membership functions, it maintains m populations of membership functions, with each population for an item $I_j (1 \geq j \geq m)$. Each chromosome in a population represents a possible set of membership functions for that item. Next, in the phase of mining fuzzy association rules, the sets of membership function for all the items are gathered together and used to mine the interesting rules from the given quantitative database. Our fuzzy mining algorithm proposed in [5] is adopted to achieve this purpose.

3 Chromosome Representation

Several possible encoding approaches in GAs have been described in [2, 8–10]. In this chapter, we adopt the encoding approach similar to that in [8]. Each individual is divided into two parts, control genes and parametric genes. In the first part, control genes are encoded into bit strings and used to determine whether parametric genes are active or not. In the second part, each set of

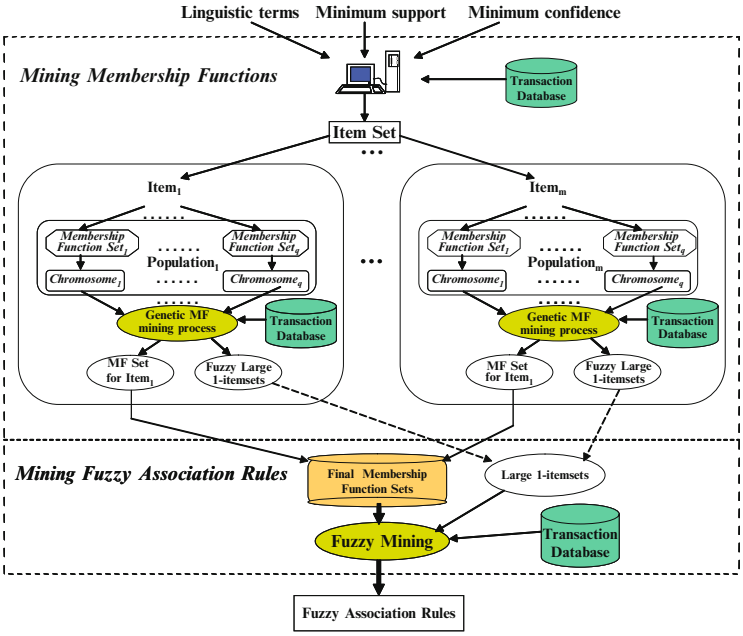


Fig. 1. The proposed GA-based framework for fuzzy mining

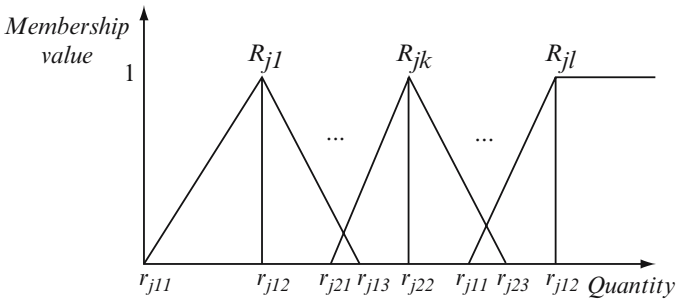


Fig. 2. The set of membership functions for item I_j

membership functions for an item is encoded as parametric genes with real-number schema.

Assume the membership functions are triangular. Three parameters are thus used to represent a membership function. Each parametric gene thus consists of three real values. Figure 2 shows an example for item I_j, where R_{jk} denotes the membership function of the k-th linguistic term and r_{jkp} indicates the p-th parameter of fuzzy region R_{jk}.

The parametric genes of item I_j can be represented as a string of r_{j11} r_{j12} r_{j13} r_{j21} r_{j22} r_{j23} $\cdots$ r_{jl1} r_{jl2} r_{jl3}, where $r_{jl3} = \infty$. The control genes of

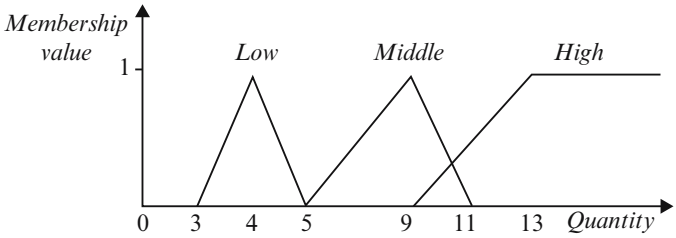


Fig. 3. An example of a possible set of membership functions for Item *milk*

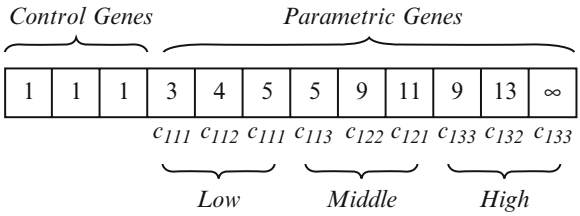


Fig. 4. The chromosome representation for the set of membership functions in Fig. 3

Item I_j can be represented as a bit string of $b_{j1}\ b_{j2} \cdots\ b_{jT}$, where T is the maximum possible number of linguistic terms. The bit b_{ji} indicates whether the i-th membership function is active or not. If $b_{ji} = 1$, the i-th membership function is active, meaning it will be used in the later fuzzy mining process. If $b_{ji} = 0$, it is inactive. All the individuals in the same population thus have the same string length. Below, an example is given to demonstrate the process of encoding membership functions.

Example 1. Assume there are four items in a transaction database: milk, bread, cookies and beverage. Also assume a possible set of membership functions for Item milk is given as shown in Fig. 3.

There are three active linguistic terms, *Low*, *Middle*, and *High*, for this item. According to the proposed encoding scheme, the individual for representing the set of membership functions in Fig. 3 is encoded as shown in Fig. 4.

In Fig. 4, the three bits in the control genes have value 1, representing the three membership functions are active. The membership function of *Low* for milk is encoded as (3, 4, 5) according to Fig. 3. Similarly, the membership functions for *Middle* and *High* are respectively encoded as (5, 9, 13) and (9, 13, ∞). The parametric genes are then the catenation of the three tuples. In another case, if the control genes of the chromosome become (1, 0, 1) during the evolution, then only two active membership functions, *Low* and *High*, will be used for the item. The proposed model can thus learn the number of labels for a variable according to the coding scheme.

4 Mining Membership Functions and Fuzzy Association Rules

4.1 Initial Population

A genetic algorithm requires a population of feasible solutions to be initialized and updated during the evolution process. As mentioned above, each individual within the population is a set of triangular membership functions for a certain item. Each membership function corresponds to a linguistic term in the item. The initial set of chromosomes is randomly generated with some constraints of forming feasible membership functions.

4.2 Fitness and Selection

In order to develop a good set of membership functions from an initial population, the genetic algorithm selects parent sets of membership functions with high fitness values for mating. An evaluation function is defined to qualify the derived sets of membership functions. Before the fitness of each set of membership functions is formally described, several related terms are first explained below. The overlap ratio of two membership functions R_{jk} and $R_{ji}(k < j)$ is defined as the overlap length divided by the minimum of the right span of R_{jk} and the left span of R_{ji}. That is,

$$overlap\_ratio(R_{jk}, R_{ji}) = \frac{overlap(R_{jk}, R_{ji})}{min(c_{jk3} - c_{jk2}, c_{ji2} - c_{ji1})},$$

where $overlap(R_{jk}, R_{ji})$ is the overlap length of R_{jk} and R_{ji}.

If the overlap length is larger than the minimum of the above two half spans, then these two membership functions are thought of as a little redundant. Appropriate punishment must then be considered in this case. Thus, the overlap factor of the membership functions for an item I_j in the chromosome C_q is defined as:

$$\sum_{\substack{k \neq i \\ R_{jk}, R_{ji} \, are \, active}} [max((\frac{overlap(R_{jk}, R_{ji})}{min(c_{jk3} - c_{jk2}, c_{ji2} - c_{ji1})}, 1) - 1)],$$

The coverage ratio of membership functions for an item I_j is defined as the coverage range of the functions divided by the maximum quantity of that item in the transactions. The more the coverage ratio is, the better the derived membership functions are. Thus, the coverage factor of the membership functions for an item I_j in the chromosome C_q is defined as:

$$coverage\_factor(C_q) = \frac{1}{\frac{range(R_{j1}, \cdots, R_{jl})}{max(I_j)}},$$

where $range(R_{j1}, R_{j2}, \cdots, R_{jl})$ is the coverage range of the active membership functions, l is the number of active membership functions for I_j, and $max(I_j)$ is the maximum quantity of I_j in the transactions.

The usage ratio of membership functions for an item I_j is defined as the number of large-1 itemsets for I_j divided by the number of active linguistic terms. Note that the maximum possible number of large-1 itemsets for an item is the number of its active linguistic terms. The more the usage ratio is, the better the derived membership functions are. Thus, the usage factor of the membership functions for an item I_j in the chromosome C_q is defined as:

$$usage\_factor(C_q) = \frac{l_{C_q}}{max(\left|L_1^{C_q}\right|, 1)},$$

where l_{C_q} is the active linguistic terms of chromosome C_q, and $max(\left|L_1^{C_q}\right|, 1)$ is the maximum of the number of large-1 itemsets and 1.

The suitability of the set of membership functions in a chromosome C_q is thus defined as $k_1 * overlap\_factor(C_q) + k_2 * coverage\_factor(C_q) + k_3 * usage\_factor(C_q)$, where k_1, k_2, k_3 are weighting factors.

The fitness value of a chromosome C_q is then defined as:

$$f(C_q) = \frac{\sum_{X \in L_1^{C_q}} fuzzy\_support(X)}{suitability(X)},$$

where $L_1^{C_q}$ is the set of large 1-itemsets obtained by using the set of membership functions in C_q, and $fuzzy\_support(X)$ is the fuzzy support of the 1-itemset X derived from C_q in the given transaction database.

The suitability factor used in the fitness function can reduce the occurrence of the two bad kinds of membership functions shown in Fig. 5, where the first one is too redundant, and the second one is too separate. It can also help generate an appropriate number of membership functions for an item.

The overlap factor in $suitable(C_q)$ is designed for avoiding the first bad case, and the coverage factor is for the second one.

Using the fuzzy-supports of the linguistic terms in the large 1-itemsets can achieve a trade-off between execution time and rule interestingness. Usually,

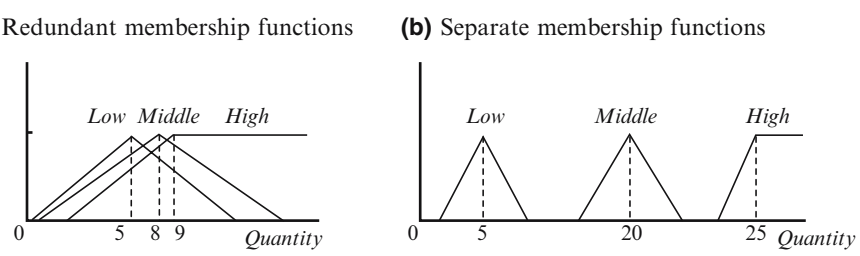


Fig. 5. Two bad sets of membership functions

a linguistic term of an item with a larger fuzzy-support in a 1-itemset will usually result in its appearance in itemsets of more items with a higher probability, which will thus usually imply more interesting association rules. The evaluation by the fuzzy supports in 1-itemsets is, however, faster than that by considering all itemsets or interesting association rules.

4.3 Genetic Operators

Genetic operators are important to the success of specific GA applications. In our approach, different crossover operators are performed for control genes and parametric genes. For control genes, the single-point crossover and the binary one-point mutation operators are used. For parametric genes, the max–min-arithmetical (MMA) crossover operator proposed in [3] and the one-point mutation for real numbers are used. The max–min-arithmetical (MMA) crossover operator proceeds as follows. Assume there are two parent chromosomes with their parametric genes as:

$$C_u^t = (c_1, \cdots, c_h, \cdots, c_z)$$
$$C_w^t = (c'_1, \cdots, c'_h, \cdots, c'_z)$$

The max–min-arithmetical (MMA) crossover operator will generate the following four candidate chromosomes from them.

1. $C_1^{t+1} = (c_{11}^{t+1}, \cdots, c_{1h}^{t+1}, \cdots, c_{1z}^{t+1})$, where $c_{1h}^{t+1} = dc_h + (1-d)'c'_h$,
2. $C_2^{t+1} = (c_{21}^{t+1}, \cdots, c_{2h}^{t+1}, \cdots, c_{2z}^{t+1})$, where $c_{2h}^{t+1} = dc'_h + (1-d)c_h$,
3. $C_3^{t+1} = (c_{31}^{t+1}, \cdots, c_{3h}^{t+1}, \cdots, c_{3z}^{t+1})$, where $c_{3h}^{t+1} = min(c_h, c'_h)$,
4. $C_4^{t+1} = (c_{41}^{t+1}, \cdots, c_{4h}^{t+1}, \cdots, c_{4z}^{t+1})$, where $c_{4h}^{t+1} = max(c_h, c'_h)$,

where the parameter d is either a constant or a variable whose value depends on the age of the population. The best two chromosomes of the four candidates are then chosen as the offspring.

The one-point mutation operator for real numbers will create a new fuzzy membership function by adding a random value ϵ (may be negative) to one parameter of an existing linguistic term, say R_{jk}. Assume that r_{jkp} represents a parameter of R_{jk}. The parameter of the newly derived membership function may be changed to $r_{jkp} + \epsilon$ by the mutation operation. Mutation at a parameter of a fuzzy membership function may, however, disrupt the order of the resulting fuzzy membership functions. These fuzzy membership functions then need rearrangement according to their values. An example is given below to demonstrate the mutation operation.

Example 2. Continuing from Example 1, assume the mutation point is set at c_{122} and the random value ϵ is set at 3. The mutation process is shown in Fig. 6.

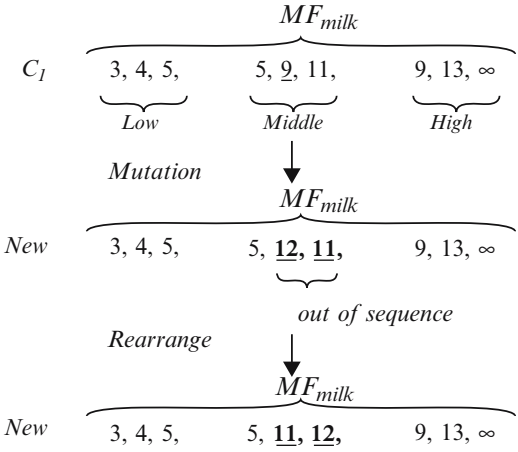


Fig. 6. A mutation operation

5 The Proposed Mining Algorithm

According to the above description, the proposed algorithm for mining both membership functions and fuzzy association rules is described below.

The proposed mining algorithm:

INPUT: A body of n quantitative transaction data, a set of m items, a maximum possible number T of linguistic terms, a support threshold α, a confidence threshold λ, and a population size P.

OUTPUT: A set of fuzzy association rules with its associated set of membership functions.

STEP 1: Randomly generate m populations, each for an item; Each individual in a population represents a possible set of membership functions for that items.

STEP 2: Encode each set of membership functions into a string representation in the way mentioned above.

STEP 3: Calculate the fitness value of each chromosome in each population by the following substeps:

STEP 3.1: For each transaction datum D_i, $i = 1$ to n, and for each item I_j, $j = 1$ to m, transfer the quantitative value $v_j^{(i)}$ into a fuzzy set $f_j^{(i)}$ represented as:

$$\left(\frac{f_{j1}^{(i)}}{R_{j1}}, \frac{f_{j2}^{(i)}}{R_{j2}}, \cdots, \frac{f_{jl}^{(i)}}{R_{jl}} \right),$$

using the corresponding membership functions represented by the chromosome, where R_{jk} is the k-th fuzzy region (term) of

item I_j, $f_{jl}^{(i)}$ is $v_j^{(i)}$'s fuzzy membership value in region R_{jk}, and $l(=|I_j|)$ is the number of active linguistic terms for I_j.

STEP 3.2: For each item region R_{jk}, calculate its scalar cardinality on the transactions as follows:

$$count_{jk} = \sum_{i=1}^{n} f_{jk}^{(i)}.$$

STEP 3.3: For each R_{jk}, $1 \leq j \leq m$ and $1 \leq k \leq |I_j|$, check whether its $count_jk$ over n is larger than or equal to the minimum support threshold α. If R_{jk} satisfies the above condition, put it in the set of large 1-itemsets (L_1). That is:

$$L_1 = \{R_{jk} | count_{jk}/n \geq \alpha, 1 \leq j \leq m \; and \; 1 \leq k \leq |I_j|\}.$$

STEP 3.4: Set the fitness value of the chromosome as the sum of the fuzzy supports (the scalar cardinalities / n) of the fuzzy regions in L_1 divided by suitability(C_q). That is:

$$f(C_q) = \frac{\sum_{X \in L_1} fuzzy\_support(X)}{suitability(C_q)}.$$

STEP 4: Execute crossover operations on each population.

STEP 5: Execute mutation operations on each population.

STEP 6: Using the selection criteria to choose individuals in each population for the next generation.

STEP 7: If the termination criterion is not satisfied, go to Step 3; otherwise, do the next step.

STEP 8: Gather the sets of membership functions, each of which has the highest fitness value in its population.

The sets of the best membership functions gathered from each population are then used to mine fuzzy association rules from the given quantitative database. Our fuzzy mining algorithm proposed in [5] is then adopted to achieve this purpose. It first transforms each quantitative value into a fuzzy set of linguistic terms using the derived membership functions. It then calculates the scalar cardinality of each linguistic term on all the transaction data. The mining process based on fuzzy counts is then performed to find fuzzy association rules. The details of the fuzzy mining algorithm [5] are described as follows.

The algorithm for mining fuzzy association rules:

INPUT: A set of n quantitative transaction data, each with m item values, a set of membership functions, a predefined minimum support threshold α , a predefined confidence threshold λ , and the large 1-itemset L_1 from the phase of mining membership functions.

OUTPUT: A set of fuzzy association rules.

STEP 1: IF L_1 is not null, then do the next step; otherwise, exit the algorithm.

STEP 2: Set $r = 1$, where r is used to represent the number of items kept in the current large itemsets.

STEP 3: Join the large itemsets L_r to generate the candidate set C_{r+1} in a way similar to that in the a priori algorithm except that two regions (linguistic terms) belonging to the same attribute can not simultaneously exist in an itemset in C_{r+1}. Restated, the algorithm first joins L_r and L_r under the condition that r-1 items in the two itemsets are the same and the other one is different. It then keeps in C_{r+1} the itemsets which have all their sub-itemsets of r items existing in L_r and do not have any two items R_{jp} and R_{jq} $(p \neq q)$ of the same attribute R_j.

STEP 4: Do the following substeps for each newly formed $(r+1)$-itemset s with items $(s_1, s_2, \ldots, s_{r+1})$ in C_{r+1}:

STEP 4.1: Calculate the fuzzy value of each transaction data $D^{(i)}$ in s as $f_s^{(i)} = f_{s_1}^{(i)} \wedge f_{s_2}^{(i)} \wedge, \ldots, \wedge f_{s_{r+1}}^{(i)}$, where $f_{s_j}^{(i)}$ is the membership value of $D^{(i)}$ in region s_j. If the minimum operator is used for the intersection, then:

$$f_s^{(i)} = Min_{j=1}^{r+1} f_{s_j}^{(i)}$$

STEP 4.2: Calculate the scalar cardinality $count_s$ of s in the transactions as:

$$count_s = \sum_{i=1}^{n} f_s^{(i)}.$$

STEP 4.3: If $count_s$ is larger than or equal to the predefined minimum support value α, put s in L_{r+1}.

STEP 5: IF L_{r+1} is null, then do the next step; otherwise, set $r = r + 1$ and repeat Steps 2–4.

STEP 6: Construct association rules for each large q-itemset s with items $(s_1, s_2, \ldots, s_q)$, $q \geq 2$, using the following substeps:

STEP 6.1: Form each possible association rule as follows:

$$s_1 \wedge \ldots \wedge s_{k-1} \wedge s_{k+1} \wedge \ldots \wedge s_q \rightarrow s_k$$

where k=1 to q.

STEP 6.2: Calculate the confidence values of all association rules using:

$$\frac{\sum_{i=1}^{n} f_s^{(i)}}{\sum_{i=1}^{n} f_{s_1}^{(i)} \wedge \ldots \wedge f_{s_{k-1}}^{(i)}, f_{s_{k+1}}^{(i)} \wedge \ldots \wedge f_{s_q}^{(i)}}$$

STEP 7: Output the association rules with confidence values larger than or equal to the predefined confidence threshold λ.

6 An Example

In this section, an example is given to illustrate the proposed mining algorithm. Assume there are four items in a transaction database: milk, bread, cookies and beverage. The data set includes the six transactions shown in Table 1.

Assume the maximum possible number (T) of fuzzy regions for each item is set at 4. The actual number of membership functions of each item will be derived by the proposal mining algorithm. For the data shown in Table 1, the proposed algorithm proceeds as follows.

STEP 1: Four populations are randomly generated, each for one item. Assume the population size is 10 in this example. Each population then includes 10 individuals. Each individual in the first population is a set of membership functions for item *milk*. Similarly, an individual in the other populations is a set of membership functions respectively for *bread*, *cookies*, and *beverage*.

STEP 2: Each set of membership functions for an item is encoded into a chromosome according to the proposed representation. Assume the ten individuals in each of the four populations are randomly generated as show in Table 2.

STEP 3: The fitness value of each chromosome is then calculated by the following substeps. Take the chromosome C_1 in $Population_3$ as an example. The membership functions in C_1 for *cookies* are represented as (1 1 1 1, 0 3 5, 3 5 10, 6 13 16, 15 20 20).

STEP 3.1: The quantitative value of each item in each transaction datum is transformed into a fuzzy set according to the active membership functions represented by that chromosome. Take the first item in transaction $T1$ as an example. The contents of $T1$ include (*milk*, 5), (*bread*, 10), (*cookies*, 7), and (*beverage*, 7). The amount "7" of item cookies is then converted into the fuzzy set:

$$\left(\frac{0}{cookies.Low}+\frac{0.6}{cookies.LowMiddle}+\frac{0.14}{cookies.MiddleHigh}+\frac{0}{cookies.High}\right)$$

by using the membership functions in C_1 in $Population_3$. The results for all the transactions by using chromosome C_1 in $Population_3$ are shown in Table 3, where the notation *item.term* is called a fuzzy region.

Table 1. Six transactions in this example

TID	Items
T1	(milk, 5); (bread, 10); (cookies, 7); (beverage, 7)
T2	(milk, 7); (bread, 14); (cookies, 12)
T3	(bread, 15); (cookies, 12); (beverage, 10)
T4	(milk, 2); (bread, 5); (cookies, 5)
T5	(bread, 9)
T6	(milk, 13); (beverage, 12)

Table 2. The ten chromosomes in each of the four populations

	$Population_1(milk)$		$Population_2(bread)$
C_1	0 1 1 1, 0 3 4, 3 4 5, 5 9 11, 9 11 11	C_1	1 0 1 1, 0 6 12, 8 12 16, 6 14 18, 12 18 18
C_2	1 1 0 1, 3 5 10, 6 13 16, 14 16 18, 15 20 20	C_2	1 1 0 1, 0 4 6, 4 6 10, 6 10 14, 8 14 14
C_3	0 1 1 1, 0 3 6, 3 6 10, 6 13 16, 12 20 20	C_3	0 0 1 1, 0 4 5, 4 5 9, 8 10 15, 9 16 16
C_4	1 0 1 1, 0 4 8, 4 8 12, 6 13 16, 12 20 20	C_4	1 1 1 1, 0 3 8, 5 10 12, 10 15 16, 17 20 20
C_5	0 1 1 1, 0 4 6, 4 5 6, 8 12 16, 15 20 20	C_5	1 1 1 1, 0 3 8, 5 10 12, 10 15 16, 17 20 20
C_6	1 1 0 1, 0 10 20, 8 12 18,12 18 20, 9 20 20	C_6	0 0 1 1, 0 5 10, 3 8 13, 5 10 15, 10 15 15
C_7	0 1 1 1, 0 3 6, 3 6 10, 6 13 16, 12 15 15	C_7	1 0 1 1, 0 5 10, 4 6 10, 8 10 15, 15 20 20
C_8	0 0 1 1, 0 5 10, 5 10 15, 7 15 18, 15 20 20	C_8	0 1 1 1, 0 3 6, 3 6 9, 6 9 12, 9 12 12
C_9	1 1 1 1, 0 3 8, 3 8 12, 9 15 18, 16 20 20	C_9	1 1 1 1, 0 8 10, 8 10 12, 10 15 20, 15 20 20
C_{10}	0 0 1 1, 0 5 8, 3 5 8, 6 13 16, 12 20 20	C_{10}	0 0 1 1, 0 5 8, 2 10 16, 8 15 20, 10 20 20
	$Population_3(cookies)$		$Population_4(beverage)$
C_1	1 1 1 1, 0 3 5, 3 5 10, 6 13 16, 15 20 20	C_1	1 1 1 1, 0 4 5, 4 5 6, 8 12 16, 15 20 20
C_2	0 1 1 1, 0 3 6, 3 6 10, 6 13 16, 12 20 20	C_2	0 0 1 1, 0 10 15, 8 12 15, 6 9 15, 9 15 15
C_3	0 0 1 1, 0 4 8,4 8 12, 6 13 16, 12 20 20	C_3	0 0 1 1, 0 3 6, 3 6 10, 6 13 16, 12 15 15
C_4	1 0 1 1, 0 5 10, 4 8 12, 8 12 16, 15 20 20	C_4	0 0 1 1, 0 3 6, 3 6 9, 6 9 12, 9 12 12
C_5	1 1 1 1, 0 3 8, 5 8 12, 10 12 16, 12 16 16	C_5	0 1 1 1, 0 8 10, 8 10 15, 10 15 20, 15 20 20
C_6	0 1 1 1, 0 5 10,4 8 10, 5 10 15, 10 15 15	C_6	1 1 1 1, 0 2 4, 2 10 16, 8 15 20, 10 20 20
C_7	1 1 1 1, 0 4 6, 4 6 10, 8 10 15, 15 20 20	C_7	0 1 1 1, 0 3 5, 3 5 10, 6 13 16, 15 20 20
C_8	1 1 0 1, 0 6 10, 6 10 15,10 12 14, 10 15 15	C_8	0 0 1 1, 0 3 6, 3 6 10, 6 13 16, 12 20 20
C_9	0 1 1 1, 0 2 5, 2 5 7, 8 10 12, 14 16 16	C_9	1 1 0 1, 0 4 8, 6 13 16,13 16 20, 12 20 20
C_{10}	1 0 1 1, 0 5 10, 8 10 14, 5 10 15, 10 15 15	C_{10}	1 0 1 1, 0 5 10, 8 10 12, 5 10 15, 10 15 15

Table 3. The fuzzy sets transformed by using chromosome C_1 in $Population_3$

TID	Transformed Fuzzy set
$T1$	$\left(\frac{0.6}{cookies.LM} + \frac{0.14}{cookies.MH}\right)$
$T2$	$\left(\frac{0.86}{cookies.MH}\right)$
$T3$	$\left(\frac{0.86}{cookies.MH}\right)$
$T4$	$\left(\frac{1}{cookies.LM}\right)$
$T5$	Null
$T6$	Null

Table 4. The counts of the fuzzy regions for item *cookies* when using C_1

$Population_3$	$Count$
$cookies.Low$	0.00
$cookies.LowMiddle$	1.60
$cookies.MiddleHigh$	1.86
$cookies.High$	0.00

STEP 3.2: The scalar cardinality of each fuzzy region in the transactions is calculated as the count value. Take the fuzzy region $cookies.LM$ as an example. Its scalar cardinality = (0.6 + 0.0 + 0.0 + 1 + 0.0 + 0.0) = 1.6. The counts of the fuzzy regions for item *cookies* using C_1 are shown in Table 4.

STEP 3.3: The count of any fuzzy region is checked against the predefined minimum support value α. Assume in this example, α is set at 0.25. Since both the count value of $cookies.LowMiddle$ and $cookies.MiddleHigh$ is larger than 0.25*6 (= 1.5), $cookies.LowMiddle$ and $cookies.MiddleHigh$ is then put in L_1.

STEP 3.4: Two large 1-itemset, $cookies.LowMiddle$ and $cookies.Middle\ High$, are derived from the membership functions of C_1 in $Population_3$. The fuzzy support of $cookies.LowMiddle$ and $cookies.MiddleHigh$ are 1.6/6 (= 0.266) and 1.86/6 (= 0.31). The suitability of C_1 is calculated as $overlap_factor(C_1) + coverage_factor(C_1) + usage_factor(C_1) = 3$ (= (0 + 0 + 0) + 1 + 2). The fitness value of C_1 is thus (0.266 + 0.31)/3 (= 0.192). The fitness values of all the chromosomes in the four populations are calculated with their results shown in Table 5.

STEP 4: The crossover operator is executed on the populations. Take C_1 and C_5 in $Population_3$ as an example. Assume for the control genes, the one-point crossover operator selects the first number as the crossover point and for the parametric genes, d is set at 0.35.The following four candidate offspring chromosomes are generated:

C_1: 1 1 1 1, 0 3 5, 3 5 10, 6 13 16, 15 20 20
C_5: 1 1 1 1, 0 3 8, 5 8 12, 10 12 16, 12 16 16

Table 5. The fitness values of all the chromosomes in the four initial populations

$Population_1$	f	$Population_2$	f
C_1	0	C_1	0.286
C_2	0.084	C_2	0.104
C_3	0	C_3	0.177
C_4	0.057	C_4	0.200
C_5	0	C_5	0
C_6	0.043	C_6	0.253
C_7	0	C_7	0.070
C_8	0	C_8	0.242
C_9	0	C_9	0.183
C_{10}	0	C_{10}	0.074
$Population_3$	f	$Population_4$	f
C_1	0.192	C_1	0.049
C_2	0.073	C_2	0.075
C_3	0.077	C_3	0.065
C_4	0.240	C_4	0
C_5	0.066	C_5	0.044
C_6	0.044	C_6	0.062
C_7	0	C_7	0.058
C_8	0.065	C_8	0.060
C_9	0	C_9	0.060
C_{10}	0.214	C_{10}	0.083

Table 6. The fitness value of the four candidate offspring chromosomes

chromosome	f	*chromosome*	f
C_1^{t+1}	0.066	C_3^{t+1}	0.209
C_2^{t+1}	0	C_4^{t+1}	0

1) C_1^{t+1} : 1 1 1 1, 0 3 6.95, 4.3 6.95 11.3, 8.6 12.35 16, 13.05 17.4 17.4
2) C_2^{t+1} : 1 1 1 1, 0 1.95 3.25, 1.95 3.25 6.5, 3.9 8.45 10.4, 9.75 13 13
3) C_3^{t+1} : 1 1 1 1, 0 3 5, 3 5 10, 6 12 16, 12 16 16
4) C_4^{t+1} : 1 1 1 1, 0 3 8, 5 8 12, 10 13 16, 15 20 20

The fitness value of the above four candidates are then evaluated, with results shown in Table 6.

In Table 6, the best two of the four candidate offspring chromosomes are chosen. Thus C_1^{t+1} and C_3^{t+1} are chosen.

STEP 5: The *mutation operator* is executed to generate possible offspring. The operation is the same as the traditional one except that rearrangement may need to be done.

STEPs 6–8: The best ten chromosomes in each population are then selected as the next generation. The same procedure is then executed until the termination criterion is satisfied. The best chromosome (with the highest

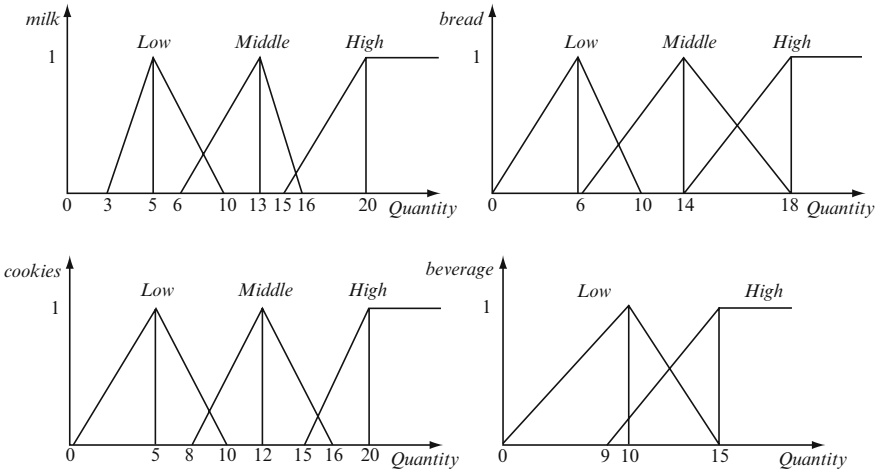


Fig. 7. The final set of membership functions

fitness value) is then output as the membership functions for deriving fuzzy association rules.After the evolutionary process terminates, the final set of membership functions for each item is shown in Fig. 7.

After the membership functions are derived, the fuzzy mining method proposed in [5] is then used to mine fuzzy association rules from the quantitative database.

7 Experimental Results

In this section, experiments made to show the performance of the proposed approach are described. They were implemented in Java on a personal computer with Intel Pentium 4 2.00 GHz and 256 MB RAM. 64 items and 10,000 transactions were used in the experiments. In each data set, the numbers of purchased items in transactions were first randomly generated. The purchased items and their quantities in each transaction were then generated. An item could not be generated twice in a transaction. The initial population size P is set at 50, the crossover rate p_c is set at 0.8, and the mutation rate p_m is set at 0.01. The parameter d of the crossover operator is set at 0.35 according to [3] and the minimum support α is set at 400.

After 500 generations, the final membership functions are apparently much better than the original ones. For example, the initial membership functions of some four items among the 64 items are shown in Fig. 8.

In Fig. 8, the membership functions have the bad types of shapes that are defined in the previous section. After 500 generations, the final membership functions for the same four items are shown in Fig. 9. It is easily seen that the membership functions in Fig. 9 is better than those in Fig. 8. The two bad

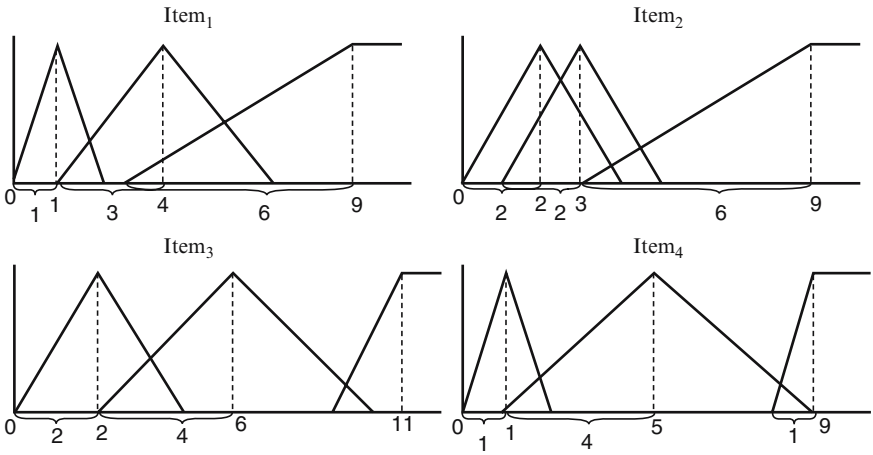


Fig. 8. The initial membership functions of some four items

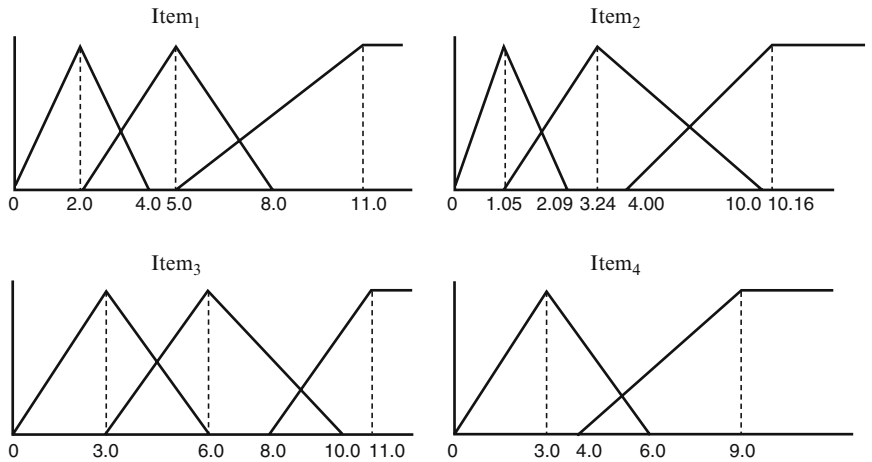


Fig. 9. The final membership functions of some four items after 500 generations

kinds of membership functions don't appear in the final results. The adopted fitness function thus works.

The average fitness values of the chromosomes in $population_1$ along with different numbers of generations are shown in Fig. 10. As expected, the curve gradually goes upward, finally converging to a certain value. The other populations have similar behavior.

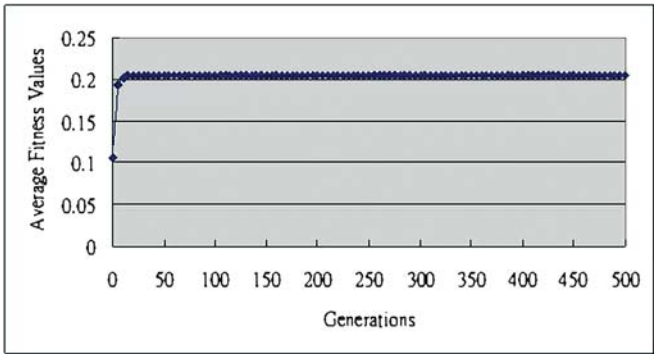


Fig. 10. The average fitness values along with different numbers of generations in $population_1$

8 Conclusion and Future Works

In this chapter, we have proposed a GA-based fuzzy data-mining algorithm for extracting both association rules and membership functions from quantitative transactions. The number of membership functions for each item is not predefined, but can be dynamically adjusted. Since the fitness of each set of membership functions is evaluated by the fuzzy-supports of the linguistic terms in the large 1-itemsets and the suitability of the derived membership functions, the derivation process can easily be done by the divide-and-conquer strategy. The experimental results show that the proposed fitness function works. Our approach can reduce human experts' intervention during the mining process, thus saving much acquisition time. In the future, we will continuously attempt to enhance the GA-based mining framework for more complex problems.

Acknowledgement

The authors would like to thank Mr. Chien-Shing Chen for his help in making the experiments. This research was supported by the National Science Council of the Republic of China under contract NSC93-2213- E-390-001.

References

1. Agrawal R, Srikant R (1994) Fast algorithm for mining association rules. The International Conference on Very Large Databases, pp. 487–499
2. Cordon O, Herrera F, Villar P (2001) Generating the knowledge base of a fuzzy rule-based system by the genetic learning of the data base. IEEE Transactions on Fuzzy Systems, Vol. 9, No. 4 pp. 667–674

3. Herrera F, Lozano M, Verdegay J L (1997) Fuzzy connectives based crossover operators to model genetic algorithms population diversity. Fuzzy Sets and Systems, Vol. 92, No. 1, pp. 21–30
4. Hong T P, Kuo C S, Chi S C (1999) Mining association rules from quantitative data. Intelligent Data Analysis, Vol. 3, No. 5, pp. 363–376
5. Hong T P, Kuo C S, Chi S C (2001) Trade-off between time complexity and number of rules for fuzzy mining from quantitative data. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, Vol. 9, No. 5, pp. 587–604
6. Kaya M, Alhajj R (2003) A clustering algorithm with genetically optimized membership functions for fuzzy association rules mining. The IEEE International Conference on Fuzzy Systems, pp. 881–886
7. Srikant R, Agrawal R (1996) Mining quantitative association rules in large relational tables. The 1996 ACM SIGMOD International Conference on Management of Data, pp. 1–12, Montreal, Canada
8. Tang K S, Man K F, Liu A F, Kwong S (1998) Mining fuzzy memberships and rules using hierarchical genetic algorithms. IEEE Transactions on Industrial Electronics, Vol. 45, No. 1 pp. 162–69
9. Wang C H, Hong T P, Tseng S S (1998) Integrating fuzzy knowledge by genetic algorithms. IEEE Transactions on Evolutionary Computation, Vol. 2, No. 4, pp. 138–149
10. Wang C H, Hong T P, Tseng S S (2000) Integrating membership functions and fuzzy rule sets from multiple knowledge sources. Fuzzy Sets and Systems, Vol. 112, pp. 141–154
11. Wang W, Bridges S M (2000) Genetic algorithm optimization of membership functions for mining fuzzy association rules. The International Joint Conference on Information Systems, Fuzzy Theory and Technology, pp. 131–134

A Compressed Vertical Binary Algorithm for Mining Frequent Patterns

J. Hdez. Palancar, R. Hdez. León, J. Medina Pagola, and A. Hechavarría

Advanced Technologies Application Center (CENATAV), 7a ♯ 21812 e/ 218 y 222, Rpto. Siboney, Playa, C.P. 12200, Ciudad de la Habana, Cuba
jpalancar@cenatav.co.cu, rhernandez@cenatav.co.cu, jmedina@cenatav.co.cu, ahechavarria@cenatav.co.cu

Summary. A new algorithm named Compressed Binary Mine (CBMine) for mining association rules and frequent patterns is presented in this chapter. Its efficiency is based on a compressed vertical binary representation of the database. CBMine was compared with several a priori implementations, like Bodon's a priori algorithm, and MAFIA, another vertical binary representation method. The experimental results have shown that CBMine has significantly better performance, especially for sparse databases.

1 Introduction

Mining association rules in transaction databases have been demonstrated to be useful and technically feasible in several application areas [14,18,21] particularly in retail sales, and it becomes every day more important in applications that use document databases [11, 16, 17]. Although research in this area has been going on for more than one decade; today, mining such rules is still one of the most popular methods in knowledge discovery and data mining.

Various algorithms have been proposed to discover large itemsets [2, 3, 6, 9, 11, 19]. Of all of them, a priori has had the biggest impact [3], since its general conception has been the base for the development of new algorithms to discover association rules.

Most of the previous algorithms adopt an a priori-like candidate set generation-and-test approach. However, candidate set generation is still costly, especially when there are many items, when the quantity of items by transaction is high, or the minimum support threshold is quite low. These algorithms need to scan the database several times to check the support of each candidate, and it is a time-consuming task for very sparse and huge databases. The weak points of the a priori algorithm are these aspects: the candidate generation and the count of each candidate support.

Performance of the algorithm could be significantly improved if we find a way to reduce the computational cost of the tasks above mentioned.

J.H. Palancar et al.: *A Compressed Vertical Binary Algorithm for Mining Frequent Patterns*, Studies in Computational Intelligence (SCI) **118**, 197–211 (2008)
www.springerlink.com

Although in [3] the way in which the transactions are represented is not mentioned by the authors, this aspect influences decisively in the algorithm. In fact, it has been one of the elements used by other authors, including us, in the formulation of new algorithms [4, 6, 7, 9, 19].

We face the problem in the following way:

- Other authors represent the transaction database as sorted lists (or array-based), BTree, Trie, etc., using items that appear in each transaction; others use horizontal or vertical binary representations. We will use a compressed vertical binary representation of the database.
- The efficiency count of each candidate's support in this representation can be improved using logical operations, which are much faster than working with non-compact forms.

A new algorithm suitable for mining association rules in databases is proposed in this chapter; this algorithm is named as CBMine (Compressed Binary Mine).

The discovery of large itemsets (the first step of the process) is computationally expensive. The generation of association rules (the second step) is the easiest of both. The overall performance of mining association rules depends on the first step; for this reason, the comparative effects of the results that we present with our algorithm covers only the first step.

In the next section we give formal definitions about association rules and frequent itemsets. Section 3 is dedicated to related work. Section 4 contains the description of CBMine algorithm. The experimental results are discussed in the Sect. 5.

The new algorithm shows significantly better performance than several algorithms, like Bodon's a priori algorithms, and in sparse databases than MAFIA, and in a general way it is applicable to those algorithms with an a priori-like approach.

2 Preliminaries

Let be $I = \{i_1, i_2, \ldots, i_n\}$ a set of elements, called items (we prefer to use the term elements instead of literals [2, 3]). Let D be a set of transactions, where each transaction T is a set of items, so that $T \subseteq I$. An association rule is an implication of the form $X \Rightarrow Y$, where $X \subset I$, $Y \subset I$, and $X \cap Y = \emptyset$. The association rule $X \Rightarrow Y$ holds in the database D with certain quality and a support s, where s is the proportion of transactions in D that contain $X \cup Y$. Some quality measures have been proposed, although these are not considered in this work.

Given a set of transactions D, the problem of mining association rules is to find all association rules that have a support greater than or equal to the user-specified minimum (called *minsup*) [3]. For example, beer and disposable diapers are items so that beer $\Rightarrow$ diaper is an association rule mined from the database if the co-occurrence rate of beer and disposable diapers (in the same transaction) is not less than *minsup*.

The first step in the discovery of association rules is to find each set of items (called *itemset*) that has co-occurrence rate above the minimum support. An itemset with at least the minimum support is called a *large* itemset or a *frequent* itemset. In this chapter, as in others, the term frequent itemset will be used. The size of an itemset represents the number of items contained in the itemset, and an itemset containing k items is called a k-itemset. For example, beer, diaper can be a frequent 2-itemset. If an itemset is frequent and no proper superset is frequent, we say that it is a *maximally* frequent itemset.

Finding all frequent itemsets has received a considerable amount of research effort in all these years because it is a very resource-consuming task. For example, if there is a frequent itemset with size l, then all $2^l - 1$ non-empty subsets of the itemset have to be generated.

The set of all subsets of I (the powerset of I) naturally forms a lattice, called the *itemset lattice* [10, 22]. For example, consider the lattice of subsets of $I = \{i_1, i_2, i_3, i_4\}$, shown in Fig. 1 (the empty set has been omitted). Each maximal frequent itemset of the figure is in bold face and in an ellipse.

Due to the *downward closure* property of itemset support – meaning that any subset of a frequent itemset is frequent – there is a border, so that all frequent itemsets lie below the border, while all infrequent itemsets lie above it. The border of frequent itemsets is shown with a bold line in Fig. 1.

An optimal association mining algorithm must only evaluate the frequent itemsets traversing the lattice in some way. This one can be done considering an equivalence class approach. The equivalence class of an itemset a, expressed as $E(a)$, is given as:

$$E(a) = \{b : |a| = k, |b| = k, Prefix_{k-1}(b) = Prefix_{k-1}(a)\}, \qquad (1)$$

where $Prefix_k(c)$ is the prefix of size k of c, i.e., its k first items in a lexicographical order.

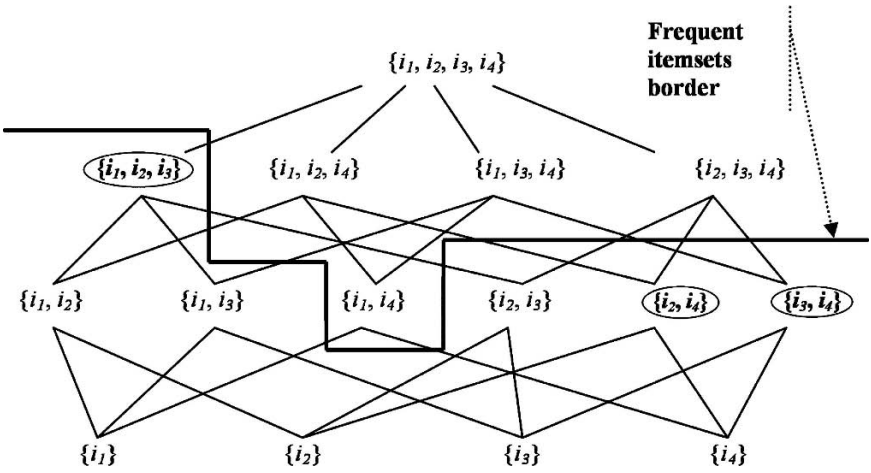


Fig. 1. Lattice of subsets of $I = \{i_1, i_2, i_3, i_4\}$

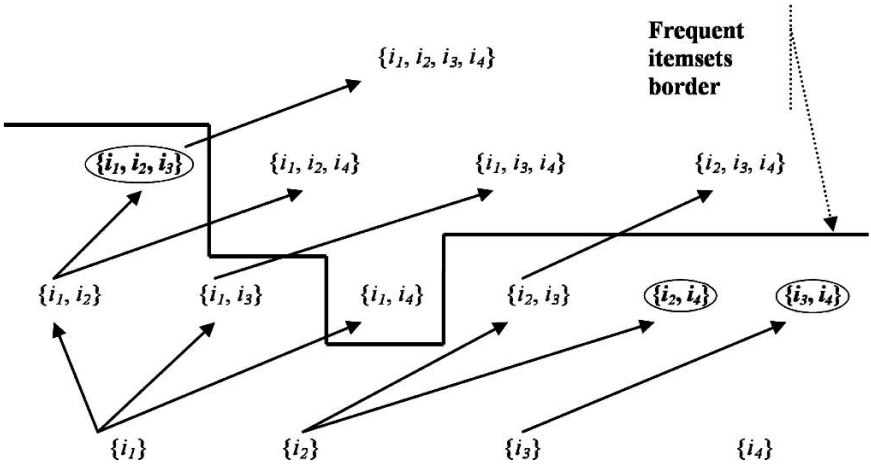


Fig. 2. Search forest of subsets of $I = \{i_1, i_2, i_3, i_4\}$

Assuming equivalence classes, the itemset lattice of Fig. 1 can be structured in a forest, shown in Fig. 2, clustering itemsets of same equivalence classes.

In order to traverse the itemset space, a convenient strategy should be chosen. Today's common approaches employ either breath-first search or depth-first search. In a breadth strategy the support values of all (k-1)-itemsets are determined before counting the support values of k-itemsets, and in a depth one recursively descends following the forest structure defined through equivalence classes [10].

The way itemsets are represented is decisive to compute their supports. Conceptually, a database is a two-dimensional matrix where the rows represent the transactions and the columns represent the items. This matrix can be implemented in the following four different formats [20]:

- Horizontal item-list (HIL): The database is represented as a set of transactions, storing each transaction as a list of item identifiers (item-list).
- Horizontal item-vector (HIV): The database is represented as a set of transactions, but each transaction is stored as a bit-vector (item-vector) of 1's and 0's to express the presence or absence of the items in the transaction.
- Vertical tid-list (VTL): The database is organized as a set of columns with each column storing an ordered list (tid-list) of only the transaction identifiers (TID) of the transactions in which the item exists.
- Vertical tid-vector (VTV): This is similar to VTL, except that each column is stored as a bit-vector (tid-vector) of 1's and 0's to express the presence or absence of the items in the transactions.

Many association rule mining algorithms have opted for a list-based (horizontally or vertically) layout (see Fig. 3) since, in general, this format takes less space than the bit-vector approach. In other way, it could be noticed that

TID	Item-lists
1	1 2 3 5
2	2 3 4 5
3	3 4 5
4	1 2 3 4 5

HIL

Item-vectors
1 1 1 0 1
0 1 1 1 1
0 0 1 1 1
1 1 1 1 1

HIV

Tid-lists				
1	2	3	4	5
1	1	1	2	1
4	2	2	3	2
	4	3	4	3
		4		4

VTL

Tid-vectors				
1	2	3	4	5
1	1	1	0	1
0	1	1	1	1
0	0	1	1	1
1	1	1	1	1

VTV

Fig. 3. Examples of database layouts

computing the supports of itemsets is simpler and faster with the vertical layout (VTL or VTV) since it involves only the intersections of tid-lists or tid-vectors.

In a general point of view, rule mining algorithms employ a combination of a traverse strategy (breadth-first or depth-first) and a form of the database layout. Examples of algorithms for horizontal mining with a breadth strategy previously presented are a priori, a prioriTID, DIC [3] and with a depth strategy are different version applying FP-Trees [1]. Other algorithms considering a VTL layout are Partition, with a breadth strategy [10], and Eclat, with a depth strategy [22].

In this chapter we evaluate a compressed form of the VTV layout, improving the performance of the itemset generation, applicable to those algorithms with an a priori-like approach.

The problem of mining frequent itemsets was first introduced by Agrawal et al. [2]. To achieve efficient mining frequent patterns, an antimonotonic property of frequent itemsets, called the a priori heuristic, was formulated in [3]. The a priori heuristic can dramatically prune candidate itemsets.

A priori is a breadth-first search algorithm, with a HIL organization, that iteratively generates two kinds of sets: C_k and L_k. The set L_k contains the large itemsets of size k (k-itemsets). Meanwhile, C_k is the set of candidate k-itemsets, representing a superset of L_k. This process continues until a null set L_k is generated.

The set L_k is obtained scanning the database and determining the support for each candidate k-itemset in C_k. The set C_k is generated from L_{k-1} with the following procedure:

$$C_k = \{c | Join(c, L_{k-1}) \wedge Prune(c, L_{k-1})\} \tag{2}$$

Algorithm: Apriori
Input: Database
Output: Large itemsets

```
1)  L1 = {large 1-itemsets};
2)  for ( k = 2; Lk-1 ≠ ∅; k++ ) do begin
3)    Ck = apriori_gen(Lk-1);  // New candidates
4)    forall transaction t ∈ D do begin
5)      Ct = subset(Ck, t);  // Candidates in t
6)      forall candidate c ∈ Ct do
7)        c.count++;
8)      end
9)    Lk = {c ∈ Ck | c.count ≥ minsup};
10)   end
11) end
12) Answer = ∪k Lk;
```

Fig. 4. Pseudocode of a priori Algorithm

where:

$$Join(\{i_1, \ldots, i_k\}, L_{k-1}) \equiv \langle \{i_1, .., i_{k-2}, i_{k-1}\} \in L_{k-1} \wedge \{i_1, .., i_{k-2}, i_k\} \in L_{k-1} \rangle, \tag{3}$$

$$Prune(c, L_{k-1}) \equiv \langle \forall s[s \subset c] \wedge |s| = k-1 \rightarrow s \in L_{k-1} \rangle. \tag{4}$$

Observe that the Join step (3) takes two (k-1)-itemsets of a same equivalence class to generate a k-itemsets.

The a priori algorithm is presented in Fig. 4.

The procedure a priori_gen used in step 3 is described in (2).

3 Related Work

The vertical binary representations (VTV) and the corresponding support counting method have been investigated by other researchers [7, 8, 10, 12, 22].

Zaki et al. proposed several algorithms using vertical binary representations in 1997 [22]. Their improvements are obtained clustering the database and applying an a priori-like method with simple tid-vectors.

Gardarin et al. proposed two breadth-first search algorithms using vertical binary representations named N-BM and H-BM in 1998 [8]. N-BM considers simple (uncompressed) vertical binary representations for itemsets. Meanwhile, H-BM uses, besides, an auxiliary bit-vector, where each bit represents a group of bits of the original bit-vector. In order to save the memory, every 1-itemset has both, while every large itemset keeps only the auxiliary bit-vector. H-BM first performs the AND between auxiliary bit-vectors and only non-zero groups are considered to the final count. However, as in large itemsets

only auxiliary bit-vectors are stored; the bit values of the items considered in the itemset need to be checked.

Burdick et al. proposed a depth-first search algorithm using vertical binary representation named MAFIA in 2001 [7]. They use bit-vectors, and it is an efficient algorithm; but only for finding maximal frequent itemsets, and especially in dense databases. Mining only maximal frequent itemsets has the following deficiency: From them we know that all its subsets are frequent, but we do not know the exact value of the supports of these subsets. Therefore, we can not obtain all the possible association rules from them.

Shenoy et al. proposed another breath-first search algorithm, called VIPER, using vertical representations. Although VIPER used a compressed binary representation on disk, when these compressed vectors are processed in memory they are converted right away into tid-lists, not considering advantages of boolean operations over binary formats [20].

Many other researchers have proposed other vertical binary algorithms, although the above mentioned are to the best of our knowledge the most representatives.

The method we present in this chapter, CBMine, obtains all frequent itemsets faster than these well-known a priori and vertical binary implementations, outperforming them considerably, especially for sparse databases.

4 CBMine Algorithm

A new method applied to a priori-like algorithms, named CBMine (Compressed Binary Mine), is analyzed in this section.

4.1 Storing the Transactions

Let us call the itemset that is obtained by removing infrequent items from a transaction the filtered transaction. The size of the filtered transactions is declared to be "substantially smaller than the size of database". Besides, all frequent itemsets can be determined even if only filtered transactions are available.

The set of filtered transactions can be represented as an m x n matrix where m is the number of transactions and n is the number of frequent items (see Fig. 5 for an 8x5 matrix). We can denote the presence or absence of an item in each transaction by a binary value (1 if it is present, else 0).

This representation has been considered as a logical view of the data. Nevertheless, some researchers have employed it for counting the support for an item and for generating the set of 1-frequent itemsets [15].

To reduce I/O cost and speed up the algorithm, the filtered transactions could be stored in main memory instead of on disk. Although this is a reasonable solution, any data structure could require a considerable – and probably a prohibitive – memory space for large databases.

Tid	Items
1	1 2 3 5
2	2 3 4 5
3	3 4 5
4	1 2 3 4 5
5	4 5
6	2 3 4
7	2 4 5
8	1 2 4 5

⇨

1 2 3 4 5
1 1 1 0 1
0 1 1 1 1
0 0 1 1 1
1 1 1 1 1
0 0 0 1 1
0 1 1 1 0
0 1 0 1 1
1 1 0 1 1

Fig. 5. Horizontal layout of the database

1 2 3 4 5
1 1 1 0 1
0 1 1 1 1
0 0 1 1 1
1 1 1 1 1
0 0 0 1 1
0 1 1 1 0
0 1 0 1 1
1 1 0 1 1

⇨

Item	Tid-vector	Array
1	1 0 0 1 0 0 0 1	{0x91}
2	1 1 0 1 0 1 1 1	{0xD7}
3	1 1 1 1 0 1 0 0	{0xF4}
4	0 1 1 1 1 1 1 1	{0x7F}
5	1 1 1 1 1 0 1 1	{0xFB}

Fig. 6. Vertical binary representation of a transaction database (word size = 8)

Considering the standard binary representation of the filtered transactions, we propose to represent these transactions vertically and store them in main memory as an array of integer numbers (a VTV organization). It should be noticed that these numbers are not defined by row but by column (see Fig. 6). The reasons for this orientation will be explained later on.

If the maximum number of transactions were not greater than a word size, the database could be stored as a simple set of integers; however, a database is normally much greater than a word size, and in many cases very much greater. For that reason, we propose to use a list of words (or integers) to store each filtered item.

Let T be the binary representation of a database, with n filtered items and m transactions. Taking from T the columns associated to frequent items, each item j can be represented as a list I_j of integers (integer-list) of word size w, as follows:

$$I_j = \{W_{1,j}, \ldots, W_{q,j}\}, q = \lceil m/w \rceil, \tag{5}$$

where each integer of the list can be defined as:

$$W_{s,j} = \sum_{r=1}^{min(w,m-(q-1)*w)} 2^{(w-r)} * t_{((s-1)*w+r),j}. \tag{6}$$

The upper expression $min(w, m-(q-1)*w)$ is included to consider the case in which the transaction number $(s-1)*w+r$ doesn't exist due to the fact that it is greater than m.

This binary representation for items, as noted Burdick et al., naturally extends to itemsets [7]. Suppose we have an integer-list A_X for an itemset X, the integer-list $A_{X\cup\{j\}}$is simply the bitwise AND of the integer-lists A_X and I_j. If an itemset has a single item then its integer-list is I_j.

The weakness of a the vertical binary representation is the memory spending, especially in sparse databases. An alternative representation is considering only non-null integers. This compressed integer-lists could be represented as an array CA of pairs $<s, B_s>$ with $1 \leq s \leq q$ and $B_s \neq 0$.

4.2 Algorithm

CBMine is a breadth-first search algorithm with a VTV organization, considering compressed integer-lists for itemset representation.

This algorithm iteratively generates a prefix list PL_k. The elements of this list have the format: $<Prefix_{k-1}, CA_{Prefix_{k-1}}, Suffixes_{Prefix_{k-1}}>$, where $Prefix_{k-1}$ is a (k-1)-itemset, $CA_{Prefix_{k-1}}$ is the corresponding compressed integer-list, and $Suffixes_{Prefix_{k-1}}$ is the set of all suffix items j of k-itemsets extended with the same $Prefix_{k-1}$, where j is lexicographically greater than every item in the prefix and the k-itemsets extended are frequent . This representation not only reduces the required memory space to store the integer-lists but also eliminates the Join step described in (3).

CBMine guarantees a significant memory reduction. Other algorithms with VTV representation have an integer-list for each itemset, meanwhile CBMine has an integer-list only for each equivalent class (see Fig. 7).

The Prune step (4) is optimized generating PL_k as a sorted list by the prefix field and, for each element, by the suffix field.

In order to determine the support of an itemset with a compressed integer-list CA, the following expression is considered:

$$Support(CA) = \sum_{<s,B_s>\in CA} BitCount(B_s), \tag{7}$$

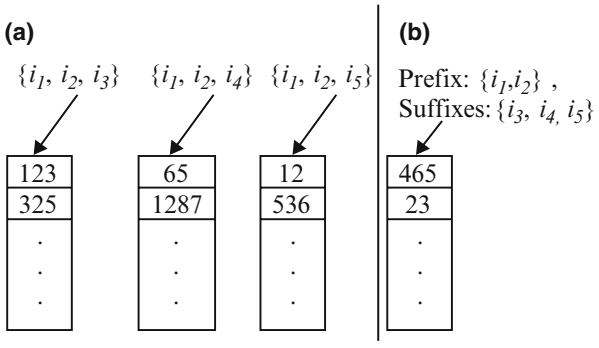


Fig. 7. (**a**) Others algorithms with VTV representations, (**b**) CBMine

where $BitCount(B_s)$ represents a function that calculates the Hamming Weight of each B_s.

Although this algorithm uses compressed integer-lists of non-null integers (CA) for itemset representation, in order to improve the efficiency, we maintain the initial integer-lists (including the null integers) $I_j = \{W_{1,j}, \ldots, W_{q,j}\}$ associated with each large 1-itemset j. This consideration allows directly accessing in I_j the integer position defined in CA.

The above consideration allows defining the formulae (8). Notice that this function represents a significant difference and improvement respect other methods.

$$CompAnd(CA, I_j) = \{<s, B_s^{'}>:<s, B_s> \in CA, \quad B_s^{'} = B_s and W_{s,j}, B_s^{'} \neq 0\}. \tag{8}$$

It could also be noticed that the cardinality of CA is reduced with the increment of the size of the itemsets due to the downward closure property. It allows the improvement of the above processes (7) and (8).

The CBMine algorithm is presented in Fig. 8.

The step 2 of the pseudocode shown in Fig. 8 (the process for $k = 2$) is performed in a similar way to that for $k \geq 3$, except the Prune procedure because it is unnecessary in this case. This procedure Prune, used in step 9, is similar to (4). Notice that this algorithm only scans the database once in the first step.

Algorithm: CBMine
Input: Database
Output: Large itemsets

```
1)  L2 = {large 1-itemsets}; // Scanning the database
2)  PL2 = {< Prefix1, CA_Prefix1, Suffixes_Prefix1 >};
3)  for ( k = 3; PL_k-1 ≠ ∅; k++ ) do
4)    forall <Prefix, CA, Suffixes> ∈ PL_k-1 do
5)      forall item j ∈ Suffixes do begin
6)        Prefix' = Prefix ∪ {j};
7)        CA' = CompAnd(CA, I_j);
8)        forall (j' ∈ Suffixes) && (j' > j) do
9)          if Prune(Prefix' ∪ {j'}, PL_k-1) &&
10)             Support(CompAnd(CA', I_j')) ≥ minsup
11)         then Suffixes'= Suffixes' ∪ {j'};
12)     if Suffixes' ≠ ∅
13)     then PL_k = PL_k ∪ {<Prefix', CA', Suffixes'>};
14)     end
15) Answer = ∪_k L_k; // L_k is obtained from PL_k
```

Fig. 8. Pseudocode of CBMine algorithm

5 Experimental Results

Here we present the experimental results of an implementation of the CBMine algorithm. It was compared with the performance of two a priori implementations made by Ferenc Bodon (Simple and Optimized algorithms) [5], and MAFIA [7].

Four known databases were used: T40I10D100K, T10I4D100K, generated from the IBM Almaden Quest research group, Chess and Pumsb*, prepared by Roberto Bayardo from the UCI datasets and PUMSB (see Table 1).

Our tests were performed on a PC with a 2.66 GHz Intel P4 processor and 512 MB of RAM. The operating system was Windows XP. Running times were obtained using standard C functions. In this chapter, the runtime includes both CPU time and I/O time.

Table 2 and the following graphics present the test results of the a priori implementations, MAFIA and CBMine with these databases. Each test was

Table 1. Database characteristics

	T10I4D100K	T40I10D100K	Pumsb*	Chess
AvgTS	10.1	39.54	50	37
MaxItems	870	942	2.087	75
Transactions	100,000	100,000	49,046	3,196

Table 2. Performance results (in s)

Databases	*minsup*	CBMine	Simple-a priori	Optimized-a priori	MAFIA
T10I4D100K	0.0004	17	32	20	52
	0.0003	25	49	39	64
	0.0002	51	198	86	104
	0.0001	135	222	192	287
T40I10D100K	0.0500	3	3	3	8
	0.0400	5	6	6	11
	0.0300	6	7	7	16
	0.0100	17	31	20	38
	0.0090	28	95	57	56
	0.0085	32	104	66	67
Pumsb*	0.7	2	2	1	2
	0.6	2	5	1	2
	0.5	2	11	5	2
	0.4	2	28	24	2
	0.3	23	106	47	11
Chess	0.9	0	3	2	0
	0.8	0	8	2	0
	0.7	1	45	3	1
	0.6	2	92	22	2
	0.5	16	163	53	7

carried out three times; the tables and graphics show the averages of the results. Bodon's a priori implementations were versions on April 26th, 2006 [5]. MAFIA implementation was a version 1.4 on February 13th, 2005; it was run with "-fi" option in order to obtain all the frequent itemsets [13].

CBMine beats the other implementations in almost all the times. It performs best results independently of the support threshold in sparse databases (T10I4D100K and T40I10D100K) (see Figs. 9 and 10 respectively). Nevertheless, we have verified that MAFIA beats CBMine for low thresholds in less sparse databases (Pumsb* and Chess) (see Figs. 11 and 12 respectively).

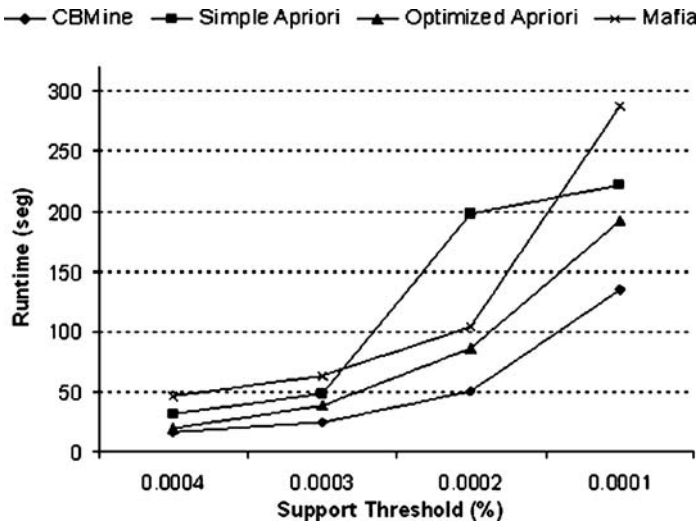


Fig. 9. T10I4D100K database

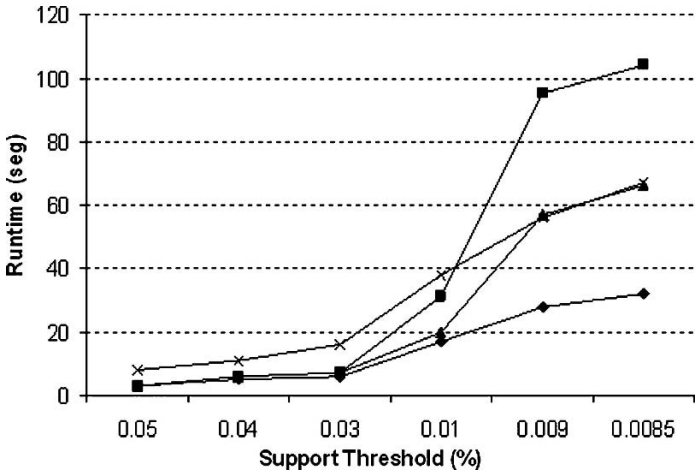


Fig. 10. T40I10D100K database

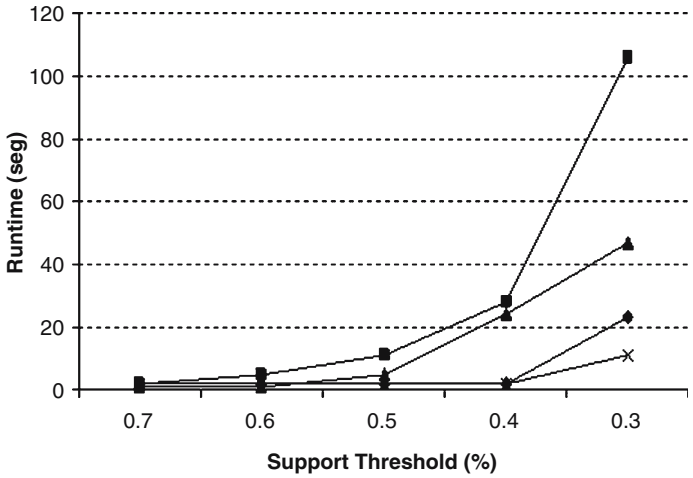


Fig. 11. Pumsb* database

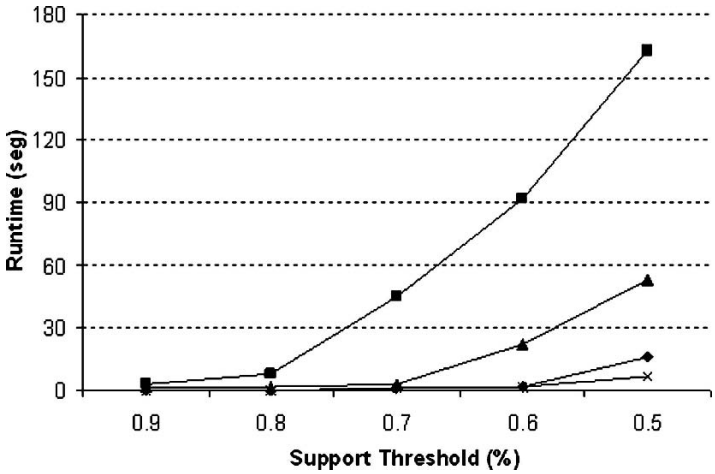


Fig. 12. Chess database

6 Conclusions

The discovery of frequent objects (itemsets, episodes, or sequential patterns) is one of the most important tasks in data mining. The ways databases and its candidates are stored cause a crucial effect on running times and memory requirements.

In this chapter we have presented a compressed vertical binary approach for mining several kinds of databases. Our experimental results show that the inclusion of this representation in a priori-like algorithms makes them more efficient and scalable.

We presented a method that obtains frequent itemset faster than other well-known a priori implementations and vertical binary implementations, outperforming them considerably, especially for sparse databases.

There are many other issues to be analyzed using a vertical compressed binary approach. This is our goal, and these issues will be included in further chapters.

References

1. Fast algorithms for frequent itemset mining using fp-trees. *IEEE Transactions on Knowledge and Data Engineering*, 17(10):1347–1362, 2005. Member-Gosta Grahne and Student Member-Jianfei Zhu
2. Agrawal R., Imielinski T., and Swami A. N. Mining association rules between sets of items in large databases. In Buneman P. and Jajodia S. editors, *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pages 207–216. Washington DC, 26–28 1993
3. Agrawal R. and Srikant R. Fast algorithms for mining association rules. In Bocca J. B., Jarke M., and Zaniolo C. editors, *Proceedings of the 20th International Conference on Very Large Data Bases, VLDB*, pages 487–499. Morgan Kaufmann, San Fransisco, CA, 12–15 1994
4. Bodon F. Surprising results of trie-based fim algorithms. In Goethals B., Zaki M. J., and Bayardo R. editors, *Proceedings of the IEEE ICDM Workshop on Frequent Itemset Mining Implementations (FIMI'04)*, volume 126 of *CEUR Workshop Proceedings*, Brighton, UK, 1 November 2004
5. Bodon F. Trie-based apriori implementation for mining frequent itemsequences. In Goethals B., Nijssen S., and Zaki M. J. editors, *Proceedings of ACM SIGKDD International Workshop on Open Source Data Mining (OSDM'05)*, pages 56–65. Chicago, IL, USA, August 2005
6. Brin S., Motwani R., Ullman J. D., and Tsur S. Dynamic itemset counting and implication rules for market basket data. In Peckham J. editor, *SIGMOD 1997, Proceedings of ACM SIGMOD International Conference on Management of Data*, pages 255–264. ACM, Tucson, Arizona, USA, May 13–15, 1997, 05 1997
7. Burdick D., Calimlim M., and Gehrke J. Mafia: A maximal frequent itemset algorithm for transactional databases. In *Proceedings of the Seventeenth International Conference on Data Engineering*, pages 443–452. Washington DC, USA, 2001. IEEE Computer Society
8. Gardarin G., Pucheral P., and Wu F. Bitmap based algorithms for mining association rules, in: Actes des journèes Bases de Donnèes Avances (BDA'98), Hammamet, Tunisie, 1998
9. Han J., Pei J., Yin Y., and Mao R. Mining frequent patterns without candidate generation: A frequent-pattern tree approach. *Data Mining and Knowledge Discovery*, 8(1):53–87, 2004
10. Hipp J., Güntzer U., and Nakhaeizadeh G. Algorithms for association rule mining – a general survey and comparison. *SIGKDD Explorations*, 2(1):58–64, July 2000
11. Holt J. D. and Chung S. M. Multipass algorithms for mining association rules in text databases. *Knowledge Information System*, 3(2):168–183, 2001

12. Lin T. Y. Data mining and machine oriented modeling: A granular computing approach. *Applied Intelligence*, 13(2):113–124, 2000
13. Calimlim M. and Gehrke J. Himalaya data mining tools: Mafia. http://himalaya-tools.sourceforge.net, May 2006
14. Fayyad U. M., Piatetsky-Shapiro G., and Smyth P. From data mining to knowledge discovery: An overview. In Fayyad U. M., Piatetsky-Shapiro G., Smyth P., and Uthurusamy R. editors, *Advances in Knowledge Discovery and Data Mining*, pages 1–34. AAAI, Menlo Park, CA, 1996
15. Gopalan R. P. and Sucahyo Y. G. High performance frequent patterns extraction using compressed fp-tree. In *Proceedings of the SIAM International Workshop on High Performance and Distributed Mining*, Orlando, USA, 2004
16. Feldman R. and Hirsh H. Finding associations in collections of text In *Machine Learning and Data Mining: Methods and Applications*, pages 223–240. Wiley, New York, 1998
17. Feldman R., Dagen I., and Hirsh H. Mining text using keyword distributions. *Journal of Intelligent Information Systems*, 10(3):281–300, 1998
18. Chen M. S., Han J., and Yu P. S. Data mining: An overview from a data-base perspective. *IEEE Transactions on Knowledge and Data Engineering*, 8(6):866–883, 1996
19. Savasere A., Omiecinski E., and Navathe S. B. An efficient algorithm for mining association rules in large databases. In *The VLDB Journal*, pages 432–444, 1995
20. Shenoy P., Haritsa J. R., Sundarshan S., Bhalotia G., Bawa M., and Shah D. Turbo-charging vertical mining of large databases. In *Proceedings of 2000 ACM SIGMOD International Conference on Management of Data*, pages 22–33, 2000
21. Cheung D. W., Han J., Ng V. T., and Wong C. Y. Maintenance of discovered association rules in large databases: An incremental updating technique. In *Proceedings of the Twelfth IEEE International Conference on Data Engineering*, pages 106–114. IEEE, New Orleans, LA, 1996
22. Zaki M. J., Parthasarathy S., Ogihara M., and Li W. New algorithms for fast discovery of association rules. Technical Report TR651, 1997

Naïve Rules Do Not Consider Underlying Causality

Lawrence J. Mazlack

Applied Artificial Intelligence Laboratory, University of Cincinnati,
Cincinnati, OH 45221-0030, USA
mazlack@uc.edu

Summary. Naïve association rules may result if the underlying causality of the rules is not considered. The greatest impact on the decision value quality of association rules may come from treating association rules as causal statements without understanding whether there is, in fact, underlying causality. A complete knowledge of all possible factors (i.e., states, events, constraints) might lead to a crisp description of whether an effect will occur. However, it is unlikely that all possible factors can be known. Commonsense understanding and reasoning accepts imprecision, uncertainty and imperfect knowledge. The events in an event/effect complex may be incompletely known; as well as, what constraints and laws the complex is subject to. Usually, commonsense reasoning is more successful in reasoning about a few large-grain sized events than many fine-grained events. A satisficing solution would be to develop large-grained solutions and only use the finer-grain when the impreciseness of the large-grain is unsatisfactory.

1 Introduction

One of the cornerstones of data mining is the development of association rules. Association rules greatest impact is in helping to make decisions. One measure of the quality of an association rule is its relative decision value. Association rules are often constructed using simplifying assumptions that lead to naïve results and consequently naïve and often wrong decisions. Perhaps the greatest area of concern about the decision value is treating association rules as causal statements without understanding whether there is, in fact, underlying causality.

Causal reasoning occupies a central position in human reasoning. It plays an essential role in human decision-making. Considerable effort over thousands of years has been spent examining causation. Whether causality exists at all or can be recognized has long been a theoretical speculation of scientists and philosophers. Serious questions have been asked whether commonsense perceptions of the world match the underlying reality. They run from the implications of Zeno's paradox [42] and Plato's cave [23] to Einstein's relativity

L.J. Mazlack: *Naïve Rules Do Not Consider Underlying Causality*, Studies in Computational Intelligence (SCI) **118**, 213–229 (2008)
www.springerlink.com

theory and modern string theory. An introduction to some of these issues may be found in Mazlack [19].

At the same time, people operate on the commonsense belief that causality exists. Causal relationships exist in the commonsense world; for example:

> When a glass is pushed off a table and breaks on the floor

it might be said that

> Being pushed from the table *caused* the glass to break.

Although,

> Being pushed from a table is not a ***certain*** *cause* of breakage; sometimes the glass bounces and no break occurs; or, someone catches the glass before it hits the floor.

Counterfactually, usually (but not always),

> *Not* falling to the floor prevents breakage.

Sometimes,

> A glass breaks when an errant object hits it, even though it does not fall from the table.

Positive causal relationships can be described as: *If* α *then* β (or, $\alpha \rightarrow \beta$). For example:

> When an automobile driver fails to stop at a red light and there is an accident it can be said that the failure to stop was the accident's *cause*.

However, negating the causal factor does not mean that the effect does not happen; sometimes effects can be *overdetermined.* For example:

> An automobile that did not fail to stop at a red light can still be involved in an accident; another car can hit it because the other car's brakes failed.

Similarly, simple negation does not work; both because an effect can be overdetermined and because negative statements are weaker than positive statements as the negative statements can become *overextended.* It cannot be said that $\neg\alpha \rightarrow \neg\beta$, for example:

> Failing to stop at a red light is not a ***certain*** *cause* of no accident occurring; sometimes no accident at all occurs.

Some describe events in terms of *enablement* and use counterfactual implication whose negation is implicit; for example [21]:

> Not picking up the ticket *enabled* him to miss the train.

There is a multiplicity of definitions of *enable* and *not-enable* and how they might be applied. To some degree, logic notation definitional disputes are involved. These issues are possibly germane to general causality theory. However, it is not profitable to the interests of this paper to consider notational issues; this paper is concerned with the less subtle needs of data analysis.

Negative causal relationships are less sure; but often stated; for example, it is often said that:

Not walking under a ladder prevents bad luck.

Or, usually (but not always),

Stopping for a red light avoids an accident.

In summary, it can be said that the knowledge of at least some causal effects is imprecise for both positive and negative descriptions. Perhaps, complete knowledge of all possible factors might lead to a crisp description of whether an effect will occur. However, it is also unlikely that it may be possible to fully know, with certainty, all of the elements involved. Consequently, the extent or actuality of missing elements may not be known. Additionally, some well described physics as well as neuro-biological events appear to be truly random [5]; and some mathematical descriptions randomly uncertain. If they are, there is no way of avoiding causal imprecision.

Coming to a precise description of what is meant by causality is difficult. There are multiple and sometimes conflicting definitions. For an introductory discussion of these issues, see Mazlack [19]. Recognizing many things with absolute certainty is problematic. As this is the case, our causal understanding is based on a foundation of inherent uncertainty and incompleteness. Consequently, causal reasoning models must accommodate inherent ambiguity. For an introductory discussion of this, see Mazlack [17].

It may well be that a precise and complete knowledge of causal events is not possible or at least uncertain. On the other hand, we have a commonsense belief that causal effects exist in the real world. If we can develop models tolerant of imprecision, it would be useful. Also, to some degree, the degree of importance that some of these items have decreases as grain size increases.

2 Satisficing

People do things in the world by exploiting commonsense *perceptions* of cause and effect. Manipulating perceptions has been explored [41] but is not the focus of this paper. The interest here is how perceptions affect commonsense causal reasoning, granularity, and the need for precision.

When trying to precisely reason about causality, complete knowledge of all of the relevant events and circumstances is needed. In commonsense, every day reasoning, approaches are used that do not require complete knowledge. Often, approaches follow what is essentially a *satisficing* [37] paradigm. The use of

non-optimal mechanisms does not necessarily result in ad hocism; Goodrich [7] states:

> "Zadeh [40] questions the feasibility (and wisdom) of seeking for optimality given limited resources. However, in resisting naïve optimizing, Zadeh does not abandon the quest for justifiability, but instead resorts to modifications of conventional logic that are compatible with linguistic and fuzzy understanding of nature and consequences."

Commonsense understanding of the world tells us that we have to deal with imprecision, uncertainty and imperfect knowledge. This is also the case with scientific knowledge of the world. An algorithmic way of handling imprecision is needed to computationally handle causality. Models are needed to algorithmically consider causes and effects. These models may be symbolic or graphic. A difficulty is striking a good balance between precise formalism and commonsense imprecise reality.

3 Complexes

When events happen, there are usually other related events. The entire collection of events can be called a complex. The events can be called the elements of the complex.

A "mechanism" [38] or a "causal complex" [11,12] is a collection of events whose occurrence or non-occurrence results in a consequent event happening. Hobbs' causal complex is the *complete* set of events and conditions necessary for the causal effect (consequent) to occur. Hobbs suggests that human casual reasoning that makes use of a causal complex does not require precise, complete knowledge of the complex. (Different workers may use the terms "mechanism" and "causal complex" differently; they are used here as these author's use them.)

Each complex, taken as a whole, can be considered to be a granule. Larger complexes can be decomposed into smaller complexes; going from large-grained to small-grained. For example, when describing starting an automobile, A large-grained to small-grained, nested causal view would start with

> When an automobile's ignition switch is turned on, this *causes* the engine to start.

But, it would not happen if a large system of other nested conditions were not in place.

> There has to be available fuel. The battery has to be operational. The switch has to be connected to the battery so electricity can flow through it. The wiring has to connect the switch to the starter and ignition system (spark plugs, etc.). The engine has to be in good working order; and so forth.

Turning the ignition switch on is one action in a complex of conditions required to start the engine. One of the events might be used to represent the collection of equal grain sized events; or, a higher level granule might be specified with the understanding that it will invoke a set of finer-grained events. In terms of nested granules, the largest grained view is: turning on the switch is the sole causal element; the complex of other elements represents the finer-grains. These elements in turn could be broken down into still finer-grains; for example, "available fuel" could be broken down into:

fuel in tank, operating fuel pump, intact fuel lines, and so forth.

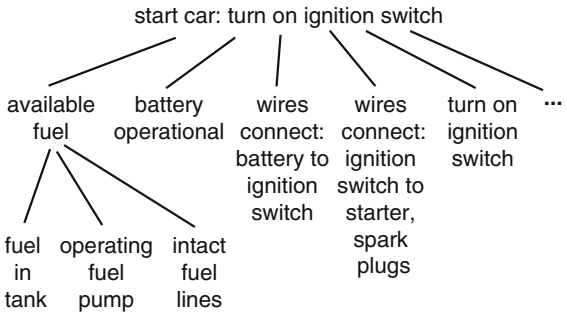


Fig. 1. Nested causal complex

Sometimes, it is enough to know what happens at a large-grained level; at other times it is necessary to know the fined grained result. For example, if

> Bill believes that turning the ignition key of his automobile *causes* the automobile to start.

It is enough if

> Bill engages an automobile mechanic when his automobile does not start when he turns the key on.

However,

> The automobile mechanic needs to know a finer-grained view of an automobile's causal complex than does Robin.

Instead of being concerned with all of the fined grained detail, a better approach may be to incorporate granulation using rough sets and/or fuzzy sets to soften the need for preciseness. And then accept impreciseness in the description. Each complex can be considered to be a granule. Larger complexes can be decomposed into smaller complexes. Thus, going from large-grained to small-grained.

Hobbs [11] uses first order logic to describe his causal complexes. Pearl [25] develops probabilistic causal networks of directed graphs (DAGs).

The causal complexes explicitly considered by Hobbs and Pearl have a required structure that may be overly restrictive for commonsense causal understanding, namely:

- If *all* of the events in the causal complex appropriately happen, then the effect will occur
- There is nothing in the causal complex that is irrelevant to the effect

These requirements are probably too precise and extensive to be realized in a commonsense world. Sometimes, only some of the events need to happen. For example,

> Someone may be able to save more money:
> - If their taxes are lowered or
> - If they earn more money.

Either even may lead to greater savings. However,

> Neither may result in increased savings if they also have to pay a large divorce settlement.

So, if all of the events happen, the effect may happen. If some of the events happen, the effect may happen. In the commonsense world, we rarely whether all of the events are in a complex are necessary. For eample,

> A man may want to attract the attention of a woman. He may do a large number of things (e.g., hair, clothes, learn to dance, etc.). If he does attract the woman, he may never know which things were relevant and which were not

An issue is how to distinguish between what is in a complex and what is not. Another issue is how to distinguish between the things that deserve to be called "causes" and those that do not. Hobbs suggests that a consideration of causal complexes can be divided into:

- Distinguishing what events are in a causal complex from those outside of it. [16, 22, 25, 35, 38]
- Within a causal complex, recognizing the events that should be identified as causes from those that are not [31].

Nested granularity may be applied to causal complexes. A complex may be several larger grained elements. In turn, each of the larger grained elements may be a complex of more fine grained elements. Recursively, in turn, these elements may be made up still finer grained elements. In general, people are more successful in applying commonsense reasoning to a few large grain sized events than the many fine grained elements that might make up a complex.

A question concerning complexes is: To what extent can we increase the causal grain size and still have useful causal information? Conversely, can

we start with a large-grained causal event and then derive the finer-grained structure? Can we measure and/or control the imprecision involved in changing grain size? If we start with a large-grained structure and resolve it, will our computational complexity burdens be reduced?

Complexes often may be best handled on a black-box, large grained basis. That is, it can be recognized that a complex exists; but we do not necessarily need to deal with the details internal to the complex.

4 Recognizing Causality is of Interest in Many Domains

Recognizing causality is of interest in many areas. Of particular interest to this paper are areas where the analysis is non-experimental. The world is taken as it is and not subject to experimentation. In the computational sciences, data mining is of concern. An area not well known to people working in the computational sciences is economics.

Perhaps, the applied area that has the greatest history of attempting to deal with causality and non-observational data is economics. Econometrics is distinguished from statistics by econometrics interest in establishing causation [13]. How and if causality can be recognized has been a significant area of discussion. Some of this discussion mirrors discussion that has gone on in the computational sciences. Hoover provides a good entry to the discussion of causality in economics.

Hume [15, p. 165], as a philosopher, suggested that causal statements are really about constant conjunction and time ordering. However, when speaking as an economist, Hume [14, p. 304] was less insistent on causal ordering: "it is of consequence to know the principle whence any phenomenon arises, and to distinguish between a cause and a concomitant effect." The issue of causal ordering is also often of importance to those modeling causality in data discovery.

Data mining analyzes data previously collected; it is non-experimental. There are several different data mining products. The most common are *conditional rules* or *association rules*. Conditional rules are most often drawn from induced trees while association rules are most often learned from tabular data.

IF Age $<$ 20
THEN vote frequency is: often
with {belief = high}

IF *Age is old*
THEN Income $<$ $10,000
with {belief = 0.8}

Fig. 2. Conditional rules

Customers who
buy ***beer*** *and* ***sausage***
also tend to buy ***hamburger***
with {*confidence = 0.7*}
in {*support = 0.15*}

Customers who
buy strawberries
also tend to buy ***whipped cream***
with {*confidence = 0.8*}
in {*support = 0.2*}

Fig. 3. Association rules

At first glance, conditional and association rules seem to imply a causal or cause-effect relationship. That is:

> A customer's purchase of both sausage and beer ***causes*** the customer to also buy hamburger.

Unfortunately, all that is discovered is the *existence* of a statistical relationship between the items. They have a degree of joint occurrence. The *nature* of the relationship is not identified. Not known is whether the presence of an item or sets of items causes the presence of another item or set of items; or the converse, or some other phenomenon causes them to occur together.

Purely accidental relationships do not have the same decision value, as do causal relationships. For example,

> IF it is true that buying both ***beer*** and ***sausage*** somehow causes someone to *buy* ***beer***,
> - THEN: A merchant might profitably put ***beer*** (or the likewise associated ***sausage***) on sale
> - AND at the same time: Increase the price of ***hamburger*** to compensate for the sausages' reduce sale price.

On the other hand, knowing that

> ***Bread*** and ***milk*** are often purchased in the same store visit

may not be as useful decision making information as both products are commonly purchased on every store visit. A knowledge of frequent co-occurrences of *bread* and *milk* purchases might lead us to placing the *bread* and *milk* at opposite ends of the store to force shoppers to visit more of the store and consequently make more impulse buying decisions. However, there is a limit to how often when such a physical distance distribution can be reasonably effected. What is most valuable is knowledge of true causal relationships.

Tangentially, what might be of interest is discovering if there is a causal relationship between the purchase of *bananas* and something else. (It turns out that *bananas* are the most frequently purchased food item at Wal-Mart [20]).

When typically developed, rules do not *necessarily* describe causality. The association rule's confidence measure is simply an estimate of conditional probability. The association rule's support indicates how often the joint occurrence happens (the joint probability over the entire data set). The strength of any causal dependency may be very different from that of a possibly related association value. In all cases

confidence $\geq$ causal dependence

All that can be said is that associations describe the strength of joint co-occurrences.

Sometimes, the association might be causal; for example, if

Someone eats salty peanuts and then drinks beer.

or

Someone drinks beer and then becomes inebriated.

there may be a causal relationship. On the other hand, if

A rooster grows and then the sun rises.

or

Someone wears a 'lucky' shirt and then wins a lottery.

there may not be a causal relationship. Recognizing true causal relationships would greatly enhances the decision value of data mining results.

The most popular market basket association rule development method identifies rules of particular interest by screening for joint probabilities (associations) above a specified threshold.

4.1 Association Rules Without an Underlying Causal Basis Can Lead to Naïve Decisions

Association rules are used is to aid in making retail decisions. However, simple association rules may lead to errors. Errors might occur; either if causality is recognized where there is no causality; or if the direction of the causal relationship is wrong [18,33]. Errors might occur; either if causality is recognized where there is no causality; or if the direction of the causal relationship is wrong. For example, if

A study of past customers shows that 94% are sick.

- Is it the following rule?
  Our customers are sick, so they buy from us.
- Is it the following complementary rule?
  If people use our products, they are likely to become sick.
- Is there no relationship between product purchase and illness?

Consequently, from a decision making viewpoint, it is not enough to know that

People both buy our products and are sick.

What is needed is knowledge of what causes what, if anything at all.

If causality is not recognized, the naïve application of association rules can result in bad decisions [33]. This can be seen in an example from Mazlack [18]:

Example:
At a particular store, a customer buys:
- *hamburger 33%* of the time
- *hot dogs* 33% of the time
- both *hamburger* and *hot dogs* 33% of the time
- *sauerkraut* only if *hot dogs* are also purchased

This would produce the binary transaction matrix:

	hamburger	hot dog	sauerkraut
t_1	1	1	1
t_2	1	0	0
t_3	0	1	1

Fig. 4. Binary transaction matrix for hamburger, hot dog, and sauerkraut purchases

This in turn would lead to the associations (confidence):
- (*hamburger, hot dog*) = 0.5
- (*hamburger, sauerkraut*) = 0.5
- (*hot dog, sauerkraut*) = 1.0

All of the support levels are adequately high for this application.

If the merchant:
- Reduced price of *hamburger* (as a sale item)
- Raised price of *sauerkraut* to compensate (as the rule *hamburger fi sauerkraut* has a high confidence.
- The offset pricing compensation would not work, as the sales of sauerkraut would not increase with the sales of *hamburger*. Most likely, the sales of *hot dogs* (and consequently, *sauerkraut*) would likely decrease as buyers would substitute *hamburger* for *hot dogs*.

4.2 Association Rules That Do Not Take into Account Quantities Can Result in Misleading Causal Inferences

Association rules are often formed by reducing all values to binary zeros and ones. This is an early technique that was and is used in data mining when analyzing market basket data. However, it is essentially flawed. Quantities do matter; some data co-occurrences are conditioned on there being a sufficiency

Actual basket:		*Binary basket:*	
Beer	Wine	Beer	Wine
6	0	1	0
0	1	0	1
12	0	1	0
0	3	0	1
24	4	1	1
24	5	1	1
48	2	1	1

Fig. 5. Beer, wine transactions: quantified and binary

of a co-occurring attribute. Also, some relationships may be non-linear based on quantity [18]

Example:

Situation: Customers frequently buy either wine or beer for themselves in varying amounts. However, when buying for a party, they often purchase both beer and wine and they usually purchase in larger quantities.

Missed rule: When at least 24 beers purchased, wine is also purchased;

Otherwise, there is no relationship between beer and wine.

Naïvely constructing an association rule on non-quantified, binary data, in this example, would find a rule that misleadingly represents the situation; i.e.,

Misleading rule: When beer is purchased, wine is also purchased
{confidence = 0.6}
{support = 0.43}

This rule is misleading because it naïvely implies that purchase probabilities are uniform; in fact, they are not. Under one set of conditions, *beer* and *wine* are *never* purchased together under one set of conditions; and, under another set of conditions they are *always* purchased together.

In neither case is there a direct causal relationship. In the quantified rule case, the larger quantities of beer and wine are caused by a third factor (a party).

5 Describing Causality

In some ways, someone may object to this paper, as it does not offer much in the way of solutions. It mostly identifies needs. Part of a reply is that there is limited space and time. Another is that recognizing a need is the first step to finding a solution. Another is that both recognizing and defining causality is still a very complex and difficult problem, even after over 3,000 years of effort.

Various causality descriptions and discovery tools have been suggested. It may eventually turn out that different subject domains may have different

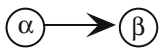


Fig. 6. Diagram indicating β that is causally dependent on α

methodological preferences. This section is intended to give a selective, non-complete, taste.

5.1 Intuitive Graph Based Approaches

Different aspects of causality have been examined. As in Fig. 6,the idea of "positive" causation ($\alpha \rightarrow \beta$) is at the core of commonsense causal reasoning. Often a positive causal relationship is represented as a network of nodes and branches [17]. In part because of their intuitive appeal, there have been many approaches to recognizing causality that use graphs.

There are a number of different books describing various aspects of causal graphs. Among them are: Gammerman [6], Glymour [8], Hausman [9], Pearl [25], Shafer [29], Spirtes [39].

5.2 Directed Graphs

Various graph based Bayesian based methods have been suggested to recognize causality. Probably the best known is the class of methods based on Directed Acyclic Graphs (DAGs). The most fully developed approach is Pearl [25]. Silverstein [33, 34] followed a similar aproach.

Pearl [24] and Spirtes [39] claim that it is possible to infer causal relationships between two variables from associations found in observational (nonexperimental) data without substantial domain knowledge. Spirtes claims that directed acyclic graphs could be used if (a) the sample size is large and (b) the distribution of random values is faithful to the causal graph. Robins [26] argues that their argument is incorrect. Lastly, Scheines [27] only claims that in some situations will it be possible to determine causality. Their discussion is tangential to the focus of this paper; going deeply into their discussion is outside this paper's scope. It is enough to note that these methods are possibly the most thoroughly developed methods of computational causal analysis.

From the commonsense causal reasoning view, the various directed graph methods have similar liabilities, specifically. Mazlack [19] discusses and lists and discusses some of the problems.

5.3 Negation and Counterfactuals

Negation or counterfactuals ($\neg\alpha \rightarrow \neg\beta$) also have a place, although it may result in reasoning errors. For example, the rule:

If a person drinks *wine*, they may become inebriated.

cannot be simply negated to

If a person ***does not*** drink *wine*, they will ***not*** become inebriated.

One reason is that effects can be *overdetermined;* that is: more than one item can cause an effect. If so, eliminating one cause does not necessarily eliminate the effect. In this case:

A person may also drink *beer* or *whiskey* to excess and become inebriated.

Events that do not happen can similarly be overdetermined. From a commonsense reasoning view, it is more likely that things do not happen than they do. For example, Oritz [21] says that it is not true that

His closing the barn door caused the horse not to escape.

because the horse might not have attempted to escape even if the door was open. Therefore, a false counterfactual is:

If he had not closed the barn door, the horse would have escaped.

Similarly, for example, the rule

If a person smokes, they will get cancer.

cannot be simply negated to

If a person **does not** smoke, they will **not** get cancer.

Again, effects can be overdetermined. In this case,

People who do not smoke may also get cancer.

Another idea that is sometimes involved in causal reasoning is *causal uncorrelatedness* [28] where if two variables have no common cause they are causally uncorrelated. This occurs if there are no single events that cause them to both change.

Similarly, Dawid [4] focuses on the negative; i.e., when α does not affect β. Dawid speaks in terms of *unresponsiveness* and *insensitivity.* If βgs *unresponsive* to α if whatever the value of α might be set to, the *value* of β will be unchanged. In parallel, if β is *insensitive* to α if whatever the value α may be set, the *uncertainty* about β will be unaffected. Along the same vein, Shoham [31,32] distinguishes between *causing, enabling,* and *preventing.* The enabling factor is often considered to be a causal factor. Shoham distinguished between background (enabling) conditions and foreground conditions. The background (enabling) conditions are inferred by default. For example [32]:

"If information is present that the key was turned and nothing is mentioned about the stated about the state of the battery, then it is inferred that the motor will start, because the battery is assumed, by default to be alive.

Given this distinction, causing is taken to refer to the foreground conditions where enabling and preventing refer to the background conditions (in this example, turning the key causes the motor to start, the live battery enables it, the dead battery prevents it)."

Other ideas that are sometimes involved in causal reasoning are *causal uncorrelatedness* [28] where if two variables share no common cause they are causally uncorrelated. This occurs if there are no single events that cause them to both change. Similarly, causal independence occurs when speaking about probabilities.

5.4 Observational and Non-Observational Data

Statistics is the traditional tool used to discover causality when handling experimental data. The standard method in the experimental sciences of recognizing causality is to perform randomized, controlled experiments. This produces experimental data. Depending on their design, randomized experiments may remove reasons for uncertainty whether or not a relationship is casual.

However, the data of greatest interest in the computational sciences, particularly data mining, is non-experimental. This is because analysis is performed on large quantities of warehoused data. In this domain, traditional statistical methods are either not useful an/or are often too computationally complex.

Even if some experimentation is possible, the amount of experimentation in contrast to the amount of data to be mined will be small. This said; some work has been done using chi-squared testing to reduce the search space [33, 34].

Several areas can only wholly (economics, sociology) or partially develop non-experimental data. In these areas, investigators can either abandon the possibility of discovering causal relationships; or, claim that causality does not exist. There continue to be efforts to discover causal relationships areas where only non-observational data is available. Among the books considering causality in non-experimental data are: Asher [1], Blalock [2], Berry [3], Hilborn [10], Shipley [30].

6 Epilogue

One of the corner stones of data mining is the development of association rules. Association rules greatest impact is in helping to make decisions. One measure of the quality of an association rule is its relative decision value. Association rules are often constructed using simplifying assumptions that lead to naïve results and consequently naïve and often wrong decisions. Perhaps the greatest area of concern is treating association rules as causal statements without understanding whether there is, in fact, underlying causality.

Causal relationships exist in the commonsense world. Knowledge of at least some causal effects is imprecise. Perhaps, complete knowledge of all possible

factors might lead to a crisp description of whether an effect will occur. However, in our commonsense world, it is unlikely that all possible factors can be known. In commonsense, every day reasoning, we use approaches that do not require complete knowledge.

People recognize that a complex collection of elements causes a particular effect, even if the precise elements of the complex are unknown. They may not know what events are in the complex; or, what constraints and laws the complex is subject to. Sometimes, the details underlying an event are known to a fine level of detail, sometimes not. Generally, people are more successful in reasoning about a few large-grain sized events than many fine-grained events. Perhaps, this can transfer over to computational models of causality.

A lack of complete, precise knowledge should not be discouraging. People do things in the world by exploiting our commonsense *perceptions* of cause and effect. When trying to precisely reason about causality, we need complete knowledge of all of the relevant events and circumstances. In commonsense, every day reasoning, we use approaches that do not require complete knowledge. Often, approaches follow what is essentially a *satisficing* paradigm.

Instead of being concerned with all of the fined grained detail, a better approach may be to incorporate granulation using rough sets and/or fuzzy sets to soften the need for preciseness. And then accept impreciseness in the description. Each complex can be considered to be a granule. Larger complexes can be decomposed into smaller complexes. Thus, going from large-grained to small-grained.

Regardless of causal recognition and representation methodologies, it is important to decision making to understand when association rules have a causal foundation. This avoids naïve decisions and increases the perceived utility of rules with causal underpinnings.

References

1. Asher, H [1983] *Causal Modeling,* Sage, Newbury Park, California
2. Blalock, H [1964] *Causal Inferences in Nonexperimental Research,* W.W. Norton, New York
3. Berry, W [1984] *Nonrecursive Causal Models,* Sage, Newbury Park, California
4. Dawid, A [1999] "Who Needs Counterfactuals" in *Causal Models and Intelligent Data Management* (ed) A. Gammerman Springer, Berlin Heidelberg New York
5. Freeman, W [1995] *Societies Of Brains*, Lawrence Erlbaum, 1995
6. Gammerman, A (ed) [1999] *Causal Models and Intelligent Data Management,* Springer, Berlin Heidelberg New York
7. Goodrich, M, Stirling, W, Boer, E [2000] "Satisficing Revisited," *Minds and Machines,* v 10, 79–109
8. Glymour, C [2001] *The Mind's Arrows,* MIT Press (Bradford), London
9. Hausman, D [1988] *Causal Asymmetries,* Cambridge University Press, Cambridge, U.K.
10. Hilborn, R, Mangel, M [1997] *The Ecological Detective: Confronting Models With Data,* Princeton University Press, Princeton, NJ

11. Hobbs, J [2001] "Causality," *Proceedings, Common Sense 2001, Fifth Symposium on Logical Formalizations of Commonsense Reasoning*, New York University, New York, May, 145–155
12. Hobbs, J [2005] "Toward a Useful Notion of Causality for Lexical Semantics", *Journal of Semantics*, Vol. 22, pp. 181–209
13. Hoover, K [2003] "Lost Causes," *HES Conference,* Presidential Address, Durham, NC
14. Hume, D [1742/1985] "Essays: Moral, Political, And Literary", in , *Liberty Classics* (ed) E. Miller, Indianapolis, 1985
15. Hume, D [1777/1902] "An Enquiry Concerning Human Understanding," in *Enquiries Concerning Human Understanding And Concerning The Principles Of Morals* (ed) L. Selby-Bigge, 2nd edition, Clarendon, Oxford, 1902
16. Lewis, D [1973] *Counterfactuals*, Harvard University Press, Cambridge University Press
17. Mazlack, L [2003a] "Commonsense Causal Modeling In The Data Mining Context," IEEE ICDM FDM Workshop Proceedings, Melbourne, Florida, November 19–22, 2003
18. Mazlack, L [2003b] "Causality Recognition For Data Mining In An Inherently Ill Defined World," 2003 BISC FLINT-CIBI International Joint Workshop On Soft Computing For Internet And Bioinformatics, December, 2003
19. Mazlack, L [2004] "Granular Causality Speculations," NAFIPS 2004, June, Banff
20. Nelson, E [1998, October 6] "Why WalMart sings, 'Yes, we have bananas'," The Wall Street Journal, B1
21. Ortiz, C [1999a] "A Commonsense Language For Reasoning About Causation And Rational Action," Artificial Intelligence, v 108, n1–2, p 125–178
22. Oritz, C [1999b] "Explanatory Update Theory: Applications Of Counterfactual Reasoning To Causation," Artificial Intelligence, v 108, n 1–2, 125–178
23. Plato [380 B.C.] *Republic,* Book VII, paragraph 514–515, G.M.A. Grube (translator), Hacket, Indianapolis, Indiana, 1992, 186–187
24. Pearl, J, Verma, T [1991] "A Theory Of Inferred Causation," *Principles Of Knowledge Representation And Reasoning: Proceedings Of The Second International Conference,* Morgan Kaufmann, 441–452
25. Pearl, J [2000] *Causality*, Cambridge University Press, New York, NY
26. Robins, R, Wasserman, L [1999], "On The Impossibility Of Inferring Causation From Association Without Background Knowledge," in (eds) C. Glymour, G. Cooper, *Computation, Causation, and Discovery* AAAI Press/MIT Press, Menlo Park, 305–321
27. Scheines, R, Spirtes, P, Glymour, C, Meek, C [1994] *Tetrad II: Tools For Causal Modeling*, Lawrence Erlbaum, Hillsdale, NJ
28. Shafer, G [1999] "Causal Conjecture," in *Causal Models and Intelligent Data Management* (ed) A. Gammerman Springer, Berlin Heidelberg New York
29. Shafer, G [1996] *The Art of Causal Conjecture*, MIT Press, Cambridge, Massachusetts
30. Shipley, B [2000] *Cause and Correlation in Biology,* Cambridge University Press, Cambridge, UK
31. Shoham, Y [1990] "Nonmonotonic Reasoning And Causation," *Cognitive Science,* v14, 213–252
32. Shoham, Y [1991] "Remarks On Simon's Comments," *Cognitive Science*, v15, 301–303

33. Silverstein, C, Brin, S, Motwani, R [1998a] "Beyond Market Baskets: Generalizing Association Rules To Dependence Rules," Data Mining And Knowledge Discovery, v 2, 39–68
34. Silverstein, C, Brin, S, Motwani, R, Ullman, J [1998b] "Scalable techniques for mining causal structures," *Proceedings 1998 VLDB Conference,* New York, NY, August 1998, 594–605
35. Simon, H [1952] "On The Definition Of The Causal Relation," *The Journal Of Philosophy,* v 49, 517–528. Reprinted in Herbert A. Simon, *Models Of Man*, John Wiley, New York, 1957
36. Simon, H [1953] "Causal ordering And Identifiability," Reprinted in Herbert A. Simon, *Models Of Man*, John Wiley, New York, 1957
37. Simon, H [1955] "A Behavioral Model Of Rational Choice," *Quarterly Journal Economics,* v 59, 99–118
38. Simon, H [1991] "Nonmonotonic Reasoning And Causation: Comment," *Cognitive Science,* v 15, 293–300
39. Spirtes, P, Glymour, C, Scheines, R [1993] *Causation, Prediction, and Search,* Springer, Berlin Heidelberg New York
40. Zadeh, L [1998] "Maximizing Sets And Fuzzy Markov Algorithms," *IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews,* v 28, 9–15
41. Zadeh, L [1999] "From computing With Numbers To Computing With Words - From Manipulation Of Measurements To Manipulation Of Perceptions," *IEEE Transactions on Circuits and Systems,* v 45, n 1, 108–119
42. Zeno (of Elea) [449 B.C.] *Paradox*, in *The Presocratic Philosophers: A Critical History with a Selection of Texts* (eds) G. Kirk and J. Raven, Cambridge, Cambridge University Press (1957)

Inexact Multiple-Grained Causal Complexes

Lawrence J. Mazlack

Applied Artificial Intelligence Laboratory, University of Cincinnati, Cincinnati, OH 45221-0030, USA
mazlack@uc.edu

Summary. Causality can be imprecise as well as granular. Complete knowledge of all possible causal factors could lead to crisp causal understanding. However, knowledge of at least some causal effects is inherently inexact and imprecise. It is unlikely that complete knowledge of all possible factors can be known for many subjects. It may not be known what events are in the complex; or, what constraints and laws the complex is subject to. Consequently, causal knowledge is inherently incomplete and inexact. Whether or not all of the elements are precisely known, people recognize that a complex of elements usually causes an effect. Causal complexes are groupings of finer-grained causal relations into a larger-grained causal object. Commonsense world understanding deals with imprecision, uncertainty and imperfect knowledge. Usually, commonsense reasoning is more successful in reasoning about a fewer large-grained events than many fine-grained events. However, the larger-grained causal objects are necessarily more imprecise than some of their components. A satisficing solution might be to develop large-grained solutions and then only go to the finer-grain when the impreciseness of the large-grain is unsatisfactory.

1 Introduction

Causal reasoning plays an essential role in human decision-making, both formal and informal (commonsense). For thousands of years, philosophers and mathematicians formally explored questions of causation. As science became more specialized, economists, physicians, cognitive scientists, physicists, psychologists, and others joined as investigators. Computer scientists, with notable exceptions, have only been sporadically interested; perhaps, the definitions are too imprecise and deeper inspection only increases the definitional uncertainty; also, the computational tractability is unclear.

Whether causality exists at all or can be recognized if it exists has long been a theoretical speculation of both scientists and philosophers. At the same time, commonsense belief in causality is the foundation of human decision-making.

L.J. Mazlack: *Inexact Multiple-Grained Causal Complexes*, Studies in Computational Intelligence (SCI) **118**, 231–249 (2008)
www.springerlink.com

1.1 Granularity

Causality is often granular. This is true both with commonsense reasoning as well as for more formal mathematical and scientific approaches. At a very fine-grained level, the physical world itself may be granular [31]. Commonsense perceptions of causality are often large-grained while the underlying causal structures may be fine-grained.

Larger-grained causal objects are often more imprecise than some of the components that are collected into the larger-grained object. Some components of a larger-grained causal object may be precisely known, while others maybe be somewhat imprecise, and others unknown. The larger the grain, the greater is the likelihood that there might be missing or unknown components. How to evaluate the impreciseness of a larger-grained causal object when the impreciseness of the underlying cascade of components is not clear. Perhaps, some form of type-II fuzzy [17] manipulation might be helpful.

1.2 Reasoning

Commonsense understanding of the world tells us that we have to deal with imprecision, uncertainty and imperfect knowledge. This is also the case with scientific knowledge of the world. A difficulty is striking a good balance between precise formalism and commonsense imprecise reality.

Causal relationships exist in the commonsense world; for example:

> When a glass is pushed off a table and breaks on the floor

it might be said that

> Being pushed from the table *caused* the glass to break.

Although,

> Being pushed from a table is not a ***certain*** *cause* of breakage; sometimes the glass bounces and no break occurs; or, someone catches the glass before it hits the floor.

Counterfactually, usually (but not always),

> *Not* falling to the floor prevents breakage.

Sometimes,

> A glass breaks when an errant object hits it, even though it does not fall from the table.

Positive causal relationships can be described as: *if* α *then* β (or, $\alpha \rightarrow \beta$). For example:

> When an automobile driver fails to stop at a red light and there is an accident; it can be said that the failure to stop was the accident's *cause.*

However, negating the causal factor does not mean that the effect does not happen; sometimes effects can be *overdetermined.* For example:

> An automobile that did not fail to stop at a red light can still be involved in an accident; another car can hit it because the other car swerved and hit it while the car was stopped.

Similarly, negating a causal description often does not work. This is often because an effect can be overdetermined. Also, negative statements are weaker than positive statements as the negative statements can become *overextended.* For example, it cannot be said that $\neg\alpha \quad \rightarrow \quad \neg\beta$, for example:

> Failing to stop at a red light is not a ***certain*** *cause* of an accident occurring; sometimes no accident at all occurs. (There may be no other cars; the other cars brake in time; etc.)

Negative causal relationships are less sure as negative statements are easily overextend; but often stated; for example, it is often said that:

> Not walking under a ladder prevents bad luck.

Or, usually (but not always),

> Stopping for a red light avoids an accident.

Some describe events in terms of *enablement* and use counterfactual implication whose negation is implicit; for example [18]:

> Not picking up the ticket *enabled* him to miss the train.

There is a multiplicity of definitions of enable and not-enable and how they might be applied. The focus of this chapter lies elsewhere.

1.3 Complexes of Elements

In causal reasoning, commonsense reasoning may recognize that a complex collection of elements can be involved causally in a particular effect, even if the precise elements of the complex are unknown. It may not be known what events are in the complex; or, what constraints and laws the complex is subject to. Sometimes, the details underlying an event are known, sometimes not. For example:

> A traffic jam might be observed in mid-town Manhattan. A reasonable supposition might be made that the jam was caused by a complex collection of elements. It might be possible to conjecture what were some of the causes; but, it is unlikely to be able to precisely know what were the complete set of actual causes.

Causal complexes may be described using nested granularity. A complex may have several larger-grained elements. In turn, each of the larger-grained elements may be made up of a complex of more fine-grained elements. Recursively, in turn, these elements may be made up still finer-grained elements. In general, people are more successful in applying commonsense reasoning to a few large-grain sized events than to many fine-grained elements.

> Somebody waiting for a dinner companion in mid-town Manhattan who observed a traffic jam might presume that they might be late because their taxi would most likely be delayed. Only the large-grained knowledge of the traffic jam would be needed; knowing the finer-grained details of the jam would probably not be needed.

When using large-grained commonsense reasoning, people do not always need to know the extent of the underling complexity. This is also true for situations not involving commonsense reasoning; for example:

> When designing an electric circuit, designers are rarely concerned with the precise properties of the materials used; instead, they are concerned with the devices functional capabilities and take the device as a larger-grained object.

Complexes often may be best handled on a black box, large-grained basis. It may be recognized that a fine-grained complex exists; but it is not necessary to deal with the finer-grained details internal to the complex.

1.4 Satisficing

The knowledge of at least some causal effects is imprecise for both positive and negative descriptions. Perhaps, complete knowledge of all possible factors might lead to a crisp description of whether an effect will occur. However, it is also unlikely that it may be possible to fully know, with certainty, all of the elements involved.

People do things in the world by exploiting commonsense *perceptions* of cause and effect. Manipulating perceptions has been explored [33] but is not the focus of this chapter. The interest here is how perceptions affect commonsense causal reasoning, granularity, and the need for precision.

When trying to precisely reason about causality, complete knowledge of all of the relevant events and circumstances is needed. In commonsense, every day reasoning, approaches are used that do not require complete knowledge. Often, approaches follow what is essentially a *satisficing* [1955] paradigm. The use of non-optimal mechanisms does not necessarily result in undesired ad hocism; Goodrich [4] states:

> Zadeh [32] questions the feasibility (and wisdom) of seeking for optimality given limited resources. However, in resisting naive optimizing,

> Zadeh does not abandon the quest for justifiability, but instead resorts to modifications of conventional logic that are compatible with linguistic and fuzzy understanding of nature and consequences.

2 Complexes

When events happen, there are usually other related events. The entire collection of events is called a *complex*. The events can be called the elements of the complex.

A "mechanism" [30] or a "causal complex" [8] is a collection of events whose occurrence or non-occurrence results in a consequent event happening. Hobbs' causal complex is the *complete* set of events and conditions necessary for the causal effect (consequent) to occur. Hobbs suggests human casual reasoning that makes use of a causal complex does not require precise, complete knowledge of the complex.

Each complex, taken as a whole, can be considered to be a granule. Larger complexes can be decomposed into smaller complexes; going from large-grained to small-grained. For example, when describing starting an automobile, A large-grained to small-grained, nested causal view (Fig. 1) would start with

> When an automobile's ignition switch is turned on, this *causes* the engine to start.

But, it would not happen if a large system of other nested conditions were not in place.

> There has to be available fuel. The battery has to be good. The switch has to be connected to the battery so electricity can flow through it. The wiring has to connect the switch to the starter and ignition system (spark plugs, etc.). The engine has to be in good working order; and so forth.

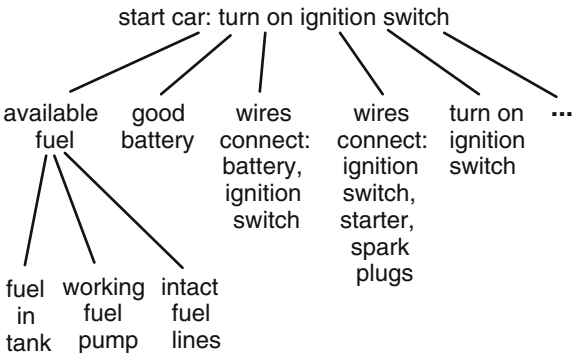


Fig. 1. Nested causal complex

Turning the ignition switch on is one action in a complex of conditions required to start the engine. One of the events might be used to represent the collection of equal grain sized events; or, a higher-level granule might be specified with the understanding that it will invoke a set of finer-grained events.

In terms of nested granules, the largest-grained event is the sole causal element: *Turn on the ignition switch.* The complex of other elements represents the finer-grains. These elements in turn could be broken down into still finer-grains; for example, "available fuel" can be broken down into:

> fuel in tank, working fuel pump, intact fuel lines, ...

Sometimes, it is enough to know what happens at a large-grained level; at other times it is necessary to know the fined grained result. For example, if

> Bill believes that turning the ignition key of his automobile *causes* the automobile to start.

It is enough if

> Bill engages an automobile mechanic when his automobile does not start when he turns the key on.

However,

> The automobile mechanic needs to know a finer-grained view of an automobile's causal complex than does Robin.

Instead of being concerned with all of the fined grained detail, a better approach may be to incorporate granulation using rough sets and/or fuzzy sets to soften the need for preciseness. If needed, larger complexes can be decomposed into smaller complexes.

Hobbs [8] uses first order logic to describe his causal complexes. Pearl [20] develops probabilistic causal networks of directed graphs (DAGs). The causal complexes explicitly considered separately by Hobbs and Pearl have a required structure that may be overly restrictive for commonsense causal understanding, namely:

- If *all* of the events in the causal complex appropriately happen, then the effect will occur
- There is nothing in the causal complex that is irrelevant to the effect

These requirements are probably too precise and extensive to be realized in a commonsense world. Sometimes, only some of the events need to happen. For example,

> Someone may be able to save more money:
> - If their taxes are lowered or
> - If they earn more money.

Either even may lead to greater savings. However,

> Neither may result in increased savings if they also have to pay a large divorce settlement.

If all of the events happen, the effect may happen. If some of the events happen, the effect may happen. In the commonsense world, we rarely know whether all of the events are in a complex are necessary. For example,

> A man may want to attract the attention of a woman. He may do a large number of things (e.g., hair, clothes, learn to dance, etc.). If he does attract the woman, he may never know which things were relevant and which were not

A way of restructuring a causal graph by increasing the grain size can be seen in Fig. 2a, b. In Fig. 2a, the idea is that solid, middle-sized, spherical things all roll are replaced with the concept class representation "ball" of Fig. 2b. However, the price of collapsing the variables is a decrease in precision.

An issue is how to distinguish between what is in a complex and what is not. Another issue is how to distinguish between the things that deserve to be called "causes" and those that do not. Hobbs [8] suggests that a consideration of causal complexes can be divided into:

- Distinguishing what events are in a causal complex from those outside of it [12, 18, 20, 27, 30].
- Within a causal complex, recognizing the events that should be identified as causes from those that are not [13, 24].

A major question concerning complexes is: To what extent can we increase the causal grain size and still have useful causal information? Conversely, can we start with a large-grained causal event and then derive the finer-grained structure? Can we measure and/or control the imprecision involved

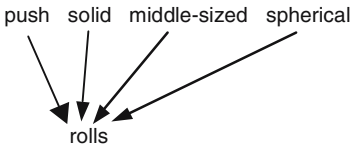


Fig. 2a. Variables (*solid, middle-sized, spherical*) with common effect [5]

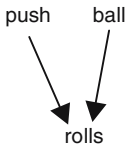


Fig. 2b. Variables with a common effect (*solid, middle-sized, spherical*) collapsed into the concept "ball" [5]

in changing grain size? If we start with a large-grained structure and resolve it, will our computational complexity burdens be reduced?

3 Defining Causality

Coming to a precise description of what is meant by causality and causal reasoning is difficult. There are multiple and sometimes conflicting definitions. Zadeh [34] suggested that a precise, formal definition might not be possible. Pearl [21] replied: "For me, the adequacy of a definition lies not in abstract argumentation but in whether the definition leads to useful ways of solving concrete problems." Regardless of arguments regarding the specificity of causality, people have a commonsense belief that there are causal relationships. Satisfactorily and explicitly specifying them is difficult, as is the description of the associated impreciseness.

Friedman [3] argues that any cause that we isolate is never the whole cause and that every direct cause itself has its own direct causes, so that networks of causation spread synchronically across the economy and diachronically back into the mists of time. If this is true, granules must necessarily be imprecise as separation trough truncation from a network would be required.

Granger [6] defined causality depends on one-way, time ordered conception of causality. In contrast, Simon [27, 28] provides an analysis of causality that does not rely on time order. Some believe [7, p 1] that causal relations are mostly indicated by asymmetric relationships. An abbreviated list of the relationships that Hausman's [7] elements of causal relationships is:

- *Time-order:* Effects do not come before causes – This corresponds with commonsense understanding. Unfortunately, it is at variance with Einsteinium space–time. This raises the question: If there is a commitment to commonsense reasoning, what should be done when commonsense reasoning differs from scientific understanding?
- *Probabilistic Independence*
- *Agency* or *manipulability:* Causes can be used to manipulate their effects, but effects cannot be used to manipulate their causes. Effects of a common cause cannot be used to manipulate one another.
- *Counterfactual dependence:* Effects counterfactually depend on their causes, while causes do not counterfactually depend on their effects
- *Overdetermination:* Effects over determine their causes, while causes rarely overdetermine their effects
- *Invariance:* Dependent variables in an equation are effects of the independent variables
- *Screening-off:* Causes screen off their effects
- *Robustness:* The relationship between cause and effect is invariant with respect to the frequency of the cause

- *Connection dependence:* If the connection between cause and effect were broken, only the effect would be affected

Counterfactuals or negation ($\neg\alpha \quad \rightarrow \neg\beta$) have a place; although they may result in less certainty in reasoning. For example:

> If a person drinks *wine*, they may become inebriated.

cannot be simply negated to

> If a person *does not* drink *wine*, they will *not* become inebriated.

One reason is that effects can be *overdetermined*; that is: more than one item can cause an effect. Eliminating one cause does not necessarily eliminate the effect. In this case:

> A person may also drink *beer* or *whiskey* to excess and become inebriated.

Events that do *not* happen can similarly be overdetermined. From a commonsense reasoning view, it is more likely that things do not happen than they do. For example, [19] states that it is not true that

> His closing the barn door caused the horse not to escape.

because the horse might not have attempted to escape even if the door was open. Therefore, a false counterfactual is:

> If he had not closed the barn door, the horse would have escaped.

Similarly, for example, the rule

> If a person smokes, they will get cancer.

cannot be simply negated to

> If a person does not smoke, they will not get cancer.

Again, effects can be overdetermined. In this case,

> People who do not smoke may also get cancer.

So far, this discussion has been on possible overdetermination; that is, the potential causes do not co-occur (the occur independently of each other). The other case is when potential causes happen at the same time. For example, when two rocks shatter a window at the same time, what causes the window to shatter [23]? Is the throwing of each individual rock a cause of the window shattering, or is it a collective cause (all the "throwings" combined). Lewis [12] calls this *redundant causation*; that is, whenever there are multiple actual distinct events, $c_1, c_2, ...c_n$, event such that each c_j without the other c_s would cause an *event*. *Preemption* (asymmetric redundancy) occurs whenever

just one of the c_s actually causes the *event*; *overdetermination* (symmetric redundancy) occurs whenever both of thec_s are causally on par with respect to the *event*.

Other ideas that are sometimes involved in causal reasoning are *causal uncorrelatedness* [22] where if two variables have no common cause they are causally uncorrelated. This occurs if there are no single events that cause them to both change.

Similarly, [1] focuses on the negative; i.e., when α does not affect β. Dawid speaks in terms of *unresponsiveness* and *insensitivity*. If β is *unresponsive* to α whatever the value of α might be set to, the *value* of β will be unchanged. In parallel, if β is *insensitive* to α if whatever the value α may be set, the *uncertainty* about β will be unaffected. Along the same vein, Shoham [24, 25] distinguishes between *causing, enabling,* and *preventing.* The enabling factor is considered to be a causal factor. Shoham distinguished between background (enabling) conditions and foreground conditions. The background (enabling) conditions are inferred by default. Causing refers to foreground conditions where enabling and preventing refer to the background conditions; in the automobile example:

> Turning the key causes the motor to start, the live battery enables it, and the dead battery prevents it.

Another idea that is sometimes involved in causal reasoning is *causal uncorrelatedness* [22] where if two variables share no common cause they are causally uncorrelated. This occurs if there are no single events that cause them to both changes. Similarly, causal independence occurs when speaking about probabilities.

4 Many Areas Would Like to Recognize Causality

Recognizing causality is of interest in many areas, included are: computational sciences, economics, philosophy, cognitive science, medicine. Of particular interest to this chapter are areas where analysis is non-experimental. In the computational sciences, data mining is of concern.

Perhaps, the applied area that has the greatest history of attempting to deal with causality and non-observational data is economics. Econometrics is distinguished from statistics by econometrics interest in establishing causation [9]. How and if causality can be recognized has been a significant area of discussion. Some of this discussion mirrors discussion that has gone on in the computational sciences. Hoover [9] provides a good entry to the discussion of causality in economics.

Hume [11, p 165], as a philosopher, suggested that causal statements are really about constant conjunction and time ordering. However, when speaking as an economist, Hume [10, p 304] was less insistent on causal ordering: "it is of consequence to know the principle whence any phenomenon arises,

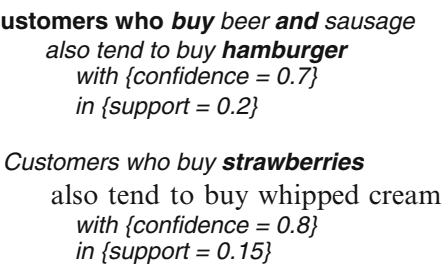


Fig. 3. Association rules

and to distinguish between a cause and a concomitant effect." The issue of causal ordering is also often of importance to those modeling causality in data discovery.

Data mining analyzes non-experimental data previously collected. There are several different data mining products. The most common are *conditional rules* or *association rules*. Conditional rules are most often drawn from induced trees while association rules are most often learned from tabular data.

At first glance, association rules (Fig. 3) seem to imply a causal or cause-effect relationship. That is:

> A customer's purchase of both sausage and beer *causes* the customer to also buy hamburger.

But, all that is discovered is the *existence* of a statistical relationship between the items. They have a degree of joint occurrence. The *nature* of the relationship is not identified. Not known is whether the presence of an item or sets of items causes the presence of another item or set of items, or if some other phenomenon causes them to jointly occur.

The information does not have a good decision value unless the degree of causality is known. Purely accidental relationships do not have the same decision value, as do causal relationships. For example,

> IF it is true that buying both ***beer*** and ***sausage*** somehow causes someone to *buy* ***beer***,
>
> - Then: A merchant might profitably put ***beer*** (or the likewise associated ***sausage***) on sale
> - And at the same time: Increase the price of ***hamburger*** to compensate for the sale price.

On the other hand, knowing that

> ***Bread*** and ***milk*** are often purchased together.

may not be useful information as both products are commonly purchased on every store visit.

When typically developed, rules do not *necessarily* describe causality. Sometimes, the association might be causal; for example, if

Someone eats salty peanuts, then drinks beer.

or

Someone drinks beer, then becomes inebriated.

there may be a causal relationship. On the other hand, if

A rooster grows, then the sun rises.

or

Someone wears a 'lucky' shirt, then wins a lottery.

there may not be a causal relationship. Recognizing true causal relationships would greatly enhances the decision value of data mining results.

4.1 Not Considering Causality Can Lead to Poor Decisions

Association rules are used is to aid in making retail decisions. However, simple association rules may lead to errors. Errors might occur; either if causality is recognized where there is no causality; or if the direction of the causal relationship is wrong [16, 26]. Errors might occur; either if causality is recognized where there is no causality; or if the direction of the causal relationship is wrong. For example, if

A study of past customers shows that 94% are sick.

- Is it the following rule?
  Our customers are sick, so they buy from us.
- Is it the following complementary rule?
  If people use our products, they become sick.
  From a decision-making viewpoint, it is not enough to know that
  People both buy our products and are sick.
  What is needed is knowledge of what causes what, if at all.

4.2 Inherently Uncertain Recognition

Recognizing many things with absolute certainty is problematic. As this is the case, our causal understanding is based on a foundation of inherent uncertainty and incompleteness. Consequently, causal reasoning models must accommodate inherent ambiguity. Some possible factors are [15]:

- Quantum physics
- Chaos theory
- Observer interference

- Space–time
- Gödel's theorem
- Arithmetic indeterminism
- Turing halting problem

Additionally, some well-described physics as well as neuro-biological events appear to be truly random [2]; as well as the mathematical descriptions that are randomly uncertain. If they are, there is no way of avoiding causal imprecision.

It may well be that a precise and complete knowledge of causal events is not possible or at least uncertain. On the other hand, we have a commonsense belief that causal effects exist in the real world. If we can develop models tolerant of imprecision, it would be useful. Also, to some degree, the degree of importance that some of these items have decreases as grain size increases.

4.3 Granular Space–Time

One of the key principles of space–time is that of *background independence*. This principle says that the geometry of space–time is not fixed. Instead, the geometry is an evolving, dynamical quantity. A closely related principle is *diffeomorphism invariance*. This principle implies that unlike theories prior to general relativity, one is free to choose any set of coordinates to map space–time and express the equations. A point in space–time is defined only by what physically happens at it, not by its location according to some special set of coordinates (no coordinates are special).

Modern physics has developed a theory that entails that space and time are granular [31]. This is an extension of quantum theory. Quantum mechanics require that certain quantities, such as the energy of an atom, can only come in specific, discrete units. Over the last few years, theory has evolved concerning quantum gravity and quantum space–time. This area of endeavor is sometimes called *loop quantum gravity.* (The term *loop* arises from how some computations in the theory involve small loops marked out in space–time.) The work is concerned with quantum theory of the structure of space–time at the smallest size scales.

What concerns us in this chapter is that there are apparently limits on fine grain size. These limits apply to areas, volumes, and time [31]. There is a non-zero minimum area (about one square Planck length, or 10^{-66} cm^2) and a discrete series of allowed quantum areas. Similarly, there is a non-zero absolute minimum volume (about one cubic Planck length, or 10^{-99} cm^2) and it restricts the set of larger volumes to a discrete series of numbers. Time is also discrete; it comes in "clicks" of approximately the Planck time. Time does not exist between the clicks; there is no "in between," in the same way that there is no water between adjacent molecules of water.

This information should influence how we think about causality. If the universe is fundamentally granular, causal descriptions need to somehow deal

with granularity. How to do this is unclear. Rough sets might be the best way of handling the granularity of causal complexes. Similarly, they seem to be a good tool to initially approach the granularity of space–time.

5 Representation

Different aspects of causality have been examined. The idea of "positive" causation of something somehow participating in the causation something else ($\alpha \rightarrow \beta$) (Fig. 4) is at the core of commonsense causal reasoning. Often a positive causal relationship is represented as a network of nodes and branches [14]. The states α, β are connected by edges that indicate that one state was the cause for the other state to come into being. The edges may be conditioned (probability, possibility, random, etc.)

Various causality descriptions and discovery tools have been suggested. It may eventually turn out that different subject domains may have different methodological preferences. This section is intended to give a selective, non-complete, taste.

Various kinds of graphs and models can be used to represent causality. Causal Bayes nets have recently received significant attention [18]. Sometimes, they are referred to simply as "graphical causal models." This is misleading, as causal Bayes nets are a subset of all possible graphic causal models. A significant difference is that a general graphic causal model allows feedback and a causal Bayes net does not. There are other significant restrictions on causal Bayes net models, included are independence conditions including various Markoff conditions. Elsewhere, the author has discussed Markoff causal graphic representations as well as the some of the unhappy limitations of using causal Bayes nets models and other models meeting some of the Markoff conditions [14]. In part, this is because "Causal Bayes nets and graphic causal models more generally, are surely an incomplete representation of the variety and wealth of causal constructions we use in science and everyday life..." Glymour [5, p 1–2]. This chapter introduces the more useful: commonsense, general imprecise graphic causal models.

One class of causal models needed for commonsense causal relationships are various kinds of cycles, including mutual causal dependencies. Often they are particularly suitable for increases in granulation. (They are not well served by causal Bayes nets.)

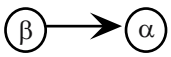


Fig. 4. Diagram indicating that α is causally dependent on β

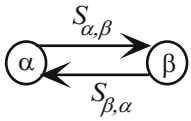


Fig. 5. Mutual causal dependency

5.1 Mutual Causal Dependencies

Mutual dependencies occur when two nodes are directly or mutually dependent on each other.

Figure 5 represents mutual dependencies; i.e., $\alpha \rightarrow \beta$ as well as $\beta \rightarrow a$. It suggests that they might do so with different strengths. $S_{i,j}$ represents the strength of the causal relationship from i to j. Often, it would seem that the strengths would be best represented by an approximate belief function. There would appear to be two variations:

- *Case 1: Different causal strengths for the same activity, occurring simultaneously:*
  For example, α could be *short men* and β could be *tall women*. If $S_{\alpha,\beta}$ meant the strength of desire for a social meeting that was caused in *short men* by the sight of *tall women*, it might be that $S_{\alpha,\beta} > S_{\beta,\mathrm{s}}$.
- *Case 2: No cumulative effect: Different causal strengths for symmetric activities, occurring at different times:*
  It would seem that if there were causal relationships in market basket data, there would often be imbalanced dependencies. For example, if a customer first buys strawberries, there may be a reasonably good chance that she will then buy whipped cream. Conversely, if she first buys whipped cream, the subsequent purchase of strawberries may be less likely. How to represent time precedence is unclear.
- *Case 3: Cumulative effect (feedback): Different causal strengths for symmetric activities, occurring at different times:*
  There are many cases of feedback where there is a cumulative effect; i.e., the strength of the relationships increase or decrease. For example, α could be *significant other's lack of interest* and β could be *depression*. Often (but not always), How to represent the change in intensity in this simple model is not clear.

5.2 General Cycles

General cycles can occur. Non-cyclic elements can influence cyclic elements. Depending on the conditioning of the cyclic nodes, the causal path might remain within the cycle, or it might branch out. As in Sect. 5.1, there may or may not be a cumulative effect (feedback).

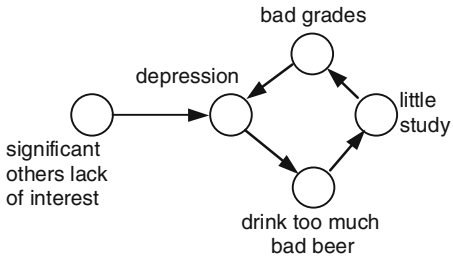


Fig. 6a. Cyclic causal dependency

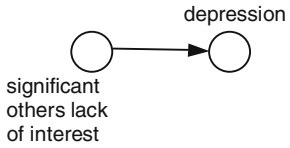


Fig. 6b. Larger grained representation of Fig. 6a

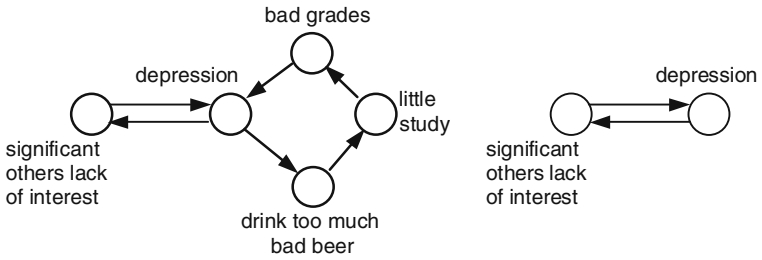


Fig. 6c. Linked cyclic causal complexes

Figure 6a represents a cycle that has external input. Possibly the cycle might be collapsed to an imprecise larger grained single node labeled "depression" as in Fig. 6b. Reasonably, applying our common sense understanding of human relationships, a return arrow could also go from "depression" to "significant other's lack of interest" as in Fig. 6c. This would create another cumulative cycle. The two cycles affect each other. How to represent the interaction of two cycles, let alone two cumulative linked cycles is unclear.

Two union of two large-grained complexes can be inconsistent. Shoham [24] provides the following example:

- Taking the engine out of a car makes it lighter
- Making a car lighter makes it go faster
- So, taking the engine out of the car makes it go faster

The union of the first two causal complexes is inconsistent. It ignores the presumption that the car has a working engine.

6 Conclusions

Whether causality can be recognized at all has long been a theoretical speculation of scientists and philosophers. At the same time, in our daily lives, we operate on the commonsense belief that causality exists.

Causal relationships exist in the commonsense world. Knowledge of all the possible causal factors of an event might lead to crisp causal understanding. However, knowledge of at least some causal effects is inherently inexact and imprecise. It is also unlikely that complete knowledge of all possible factors can be known for many subjects. Consequently, causal knowledge is inherently incomplete and inexact.

Commonsense world understanding deals with imprecision, uncertainty and imperfect knowledge. Even if the precise elements are unknown, people recognize that a complex of elements usually causes a particular effect. Causal complexes are groupings of finer-grained causal relations into a larger-grained causal object. It may not be known what events are in the complex; or, what constraints and laws the complex is subject to. Potentially, commonsense reasoning can work with imprecise causal complexes.

The details underlying an event may or may not be precisely known. Instead of being concerned with all of the fined grained detail, a better approach may be to incorporate granulation using rough sets and/or fuzzy sets to soften the need for preciseness. And then accept impreciseness in the description. Each complex can be considered to be a granule. Larger complexes can be decomposed into smaller, finer-grained complexes.

Larger-grained causal objects are often more imprecise than some of the components that are collected into the larger-grained object. Some components of a larger-grained causal object may be precisely known, while others maybe somewhat imprecise, and others unknown. The larger the grain, the greater is the likelihood that there might be missing or unknown components. How to evaluate the impreciseness of a larger-grained causal object when the impreciseness of the underlying cascade of components is not clear.

Usually, commonsense reasoning is more successful in reasoning about a fewer large-grained events than many fine-grained events. However, the larger-grained causal objects are necessarily more imprecise than some of their components. A satisficing solution might be to develop large-grained solutions and then only go to the finer-grain when the impreciseness of the large-grain is unsatisfactory.

Causality is often imprecise and granular. Methods and theories accommodating both imprecision and granularity need to be developed and refined.

References

1. Dawid, A. (1999) Who Needs Counterfactuals, in *Causal Models and Intelligent Data Management*, A. Gammerman (ed.), Springer, Berlin Heidelberg New York
2. Freeman, W. (1995) *Societies Of Brains*, Lawrence Erlbaum, Hillsdale, NJ

3. Friedman, M. (1949) The Marshallian Demand Curve, *Journal of Political Economy*, vol. 57, 463–495
4. Goodrich, M., Stirling, W., Boer, E. (2000)Satisficing Revisited, *Minds and Machines*, vol. 10, 79–109
5. Glymour, C. (2001) *The Mind's Arrows, Bayes Nets and Graphical Causal Models in Psychology*, MIT, Cambridge, MA
6. Granger, C. (1969) Investigating Causal Relations by Econometric Models and Cross-Spectral Methods, *Econometrica*, vol. 37, 424–438
7. Hausman, D. (1998) *Causal Asymmetries*, Cambridge University Press, Cambridge, UK
8. Hobbs, J. (2001) Causality, *Proceedings, Common Sense 2001, Fifth Symposium on Logical Formalizations of Commonsense Reasoning*, New York University, New York, May, 145–155
9. Hoover, K. (2003) "Lost Causes," *HES Conference*, Presidential Address, Durham, North Carolina
10. Hume, D. (1742/1985) *Essays: Moral, Political, and Literary*, E. Miller (ed.), Liberty Classics, Indianapolis, 1985
11. Hume, D. (1777/1902) An Enquiry Concerning Human Understanding, in *Enquiries Concerning Human Understanding and Concerning the Principles of Morals*, L. Selby-Bigge (ed.) 2nd edition, Clarendon Press, Oxford, 1902
12. Lewis, D. (1986) Causation, *Philosophical Papers II*, Oxford University Press, Oxford, 159–213
13. Mackie, J. (1965) Causes and Conditions, American Philosophical Quarterly 2 (1965): 245–64 in E. Sosa, M. Tooley, (eds.), Causation 33, Oxford University Press, 33–55
14. Mazlack, L. (2004a) Causal Satisficing and Markoff Models in the Context of Data Mining, *NAFIPS 2004 Proceedings*, June, Banff
15. Mazlack, L. (2004b) Granular Causality Speculations, *NAFIPS 2004 Proceedings*, June, Banff
16. Mazlack, L. (2004c) Association Rules Without Understanding the Underlying Causality Can Lead to Naïve Decisions, *Proceedings of the Third International Conference on Machine Learning and Applications (ICMLA'04)*, Louisville, December, 2004
17. Mendel, J. (2000) *Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions*, Prentice Hall, NJ
18. Ortiz, C. (1999a) A Commonsense Language for Reasoning About Causation and Rational Action, *Artificial Intelligence*, vol. 108, no. 1–2, 125–178
19. Ortiz, C. (1999b) A Commonsense Language for Reasoning About Causation and Rational Action, *Artificial Intelligence*, vol. 111, no. 2, 73–130
20. Pearl, J. (2000) *Causality*, Cambridge University Press, New York
21. Pearl, J. (2001) From a web page supporting Pearl's 2000 book, in reply to a question, Date: September 28, 2001; From: Sampsa Hautaniemi, NIH; Subject: Zadeh's 'Causality is Undefinable', http://bayes.cs.ucla.edu/BOOK-2K/hautaniemi.html
22. Shafer, G. (1999) Causal Conjecture, in *Causal Models and Intelligent Data Management*, A. Gammerman (ed.), Springer, Berlin Heidelberg New York
23. Schaffer, J. (2003) Overdetermining Causes, *Philosophical Studies*, vol. 114, 23–45
24. Shoham, Y. (1990) Nonmonotonic Reasoning and Causation, *Cognitive Science*, vol. 14, 213–252

25. Shoham, Y. (1991) Remarks on Simon's Comments, *Cognitive Science*, vol. 15, 301–303
26. Silverstein, C., Brin, S., Motwani, R. (1998) Beyond Market Baskets: Generalizing Association Rules to Dependence Rules, Data Mining and Knowledge Discovery, vol. 2, 39–68
27. Simon, H. (1952) On the Definition of the Causal Relation, *The Journal of Philosophy*, vol. 49, 517–528. Reprinted in H.A. Simon, *Models Of Man*, Wiley, New York, 1957
28. Simon, H. (1953) Causal ordering and Identifiability, Reprinted in H.A. Simon, *Models of Man*, Wiley, New York, 1957
29. Simon, H. (1955) A Behavior Model of Rational Choice,*Quarterly Journal of Economics*, vol. 59, 99–118
30. Simon, H. (1991) Nonmonotonic Reasoning and Causation: Comment, *Cognitive Science*, vol. 15, 293–300
31. Smolin, L. (2004) Atoms of Space and Time, *Scientific American*, January, 2004, 66–75
32. Zadeh, L. (1998) Maximizing Sets and Fuzzy Markov Algorithms, *IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews*, vol. 28, 9–15
33. Zadeh, L. (1999) From Computing with Numbers to Computing with Words – from Manipulation of Measurements to Manipulation of Perceptions, *IEEE Transactions on Circuits and Systems*, vol. 45, no. 1, 108–119
34. Zadeh, L. (2001) Causality is Undefinable – Towards a Theory of Hierarchical Definability, (abstract) *Proceedings of the Tenth IEEE International Conference on Fuzzy Systems*, December, Melbourne, Australia, 67–68

Does Relevance Matter to Data Mining Research?

Mykola Pechenizkiy[1,2], Seppo Puuronen[2], and Alexey Tsymbal[3,4]

[1] Information Systems Group, Department of Computer Science, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, The Netherlands
m.pechenizkiy@tue.nl
[2] Department of Computer Science and Information Systems, University of Jyväskylä, P.O. Box 35, FIN-40351, Jyväskylä, Finland
mpechen@cs.jyu.fi,sepi@cs.jyu.fi
[3] Department of Computer Science, Trinity College Dublin, Dublin 2, Ireland
[4] Corporate Technology Division, Siemens AG, Günther-Scharowsky-Str. 1, 91058 Erlangen, Germany
alexey.tsymbal@siemens.com

Summary. Data mining (DM) and knowledge discovery are intelligent tools that help to accumulate and process data and make use of it. We review several existing frameworks for DM research that originate from different paradigms. These DM frameworks mainly address various DM algorithms for the different steps of the DM process. Recent research has shown that many real-world problems require integration of several DM algorithms from different paradigms in order to produce a better solution elevating the importance of practice-oriented aspects also in DM research. In this chapter we strongly emphasize that DM research should also take into account the relevance of research, not only the rigor of it. Under relevance of research in general, we understand how good this research is in terms of the utility of its results. This chapter motivates development of such a new framework for DM research that would explicitly include the concept of relevance. We introduce the basic idea behind such framework and propose one sketch for the new framework for DM research based on results achieved in the information systems area having some tradition related to the relevance aspects of research.

1 Introduction

Data mining (DM) and knowledge discovery are intelligent tools that help to accumulate and process data and make use of it [13]. DM bridges many technical areas, including databases, statistics, machine learning, and human-computer interaction. The set of DM processes used to extract and verify patterns in data is the core of the knowledge discovery process [40]. These processes include data cleaning, feature transformation, algorithm and parameter selection, and evaluation, interpretation and validation (Fig. 1).

M. Pechenizkiy et al.: *Does Relevance Matter to Data Mining Research?*, Studies in Computational Intelligence (SCI) **118**, 251–275 (2008)
www.springerlink.com

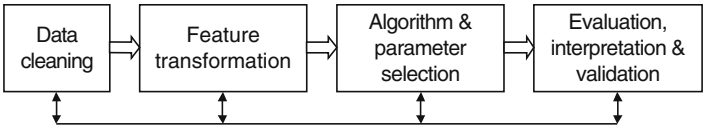


Fig. 1. Data mining process (adapted from [40])

The idea of learning from data is far from being new. However, likely due to the developments in the database management field and due to the great increase of data volumes being accumulated in databases the interest in DM has become very intense. Numerous DM algorithms have recently been developed to extract knowledge from large databases. Currently, most research in DM focuses on the development of new algorithms or improvement in the speed or the accuracy of the existing ones [30].

Relatively little has been published about theoretical frameworks of DM. A few theoretical approaches to DM were considered in [30]. A motivation for DM foundations development, and requirements for a theoretical DM framework were also considered in [30]: a theoretical framework should be simple and easy to apply; it should contribute to DM algorithms and DM systems development; it should be able to model typical DM tasks like clustering, classification and rule discovery; and it should recognize that DM is an iterative and interactive process, where a user has to be involved.

In this chapter (in Sect. 2) we consider (1) several existing foundations-oriented frameworks for DM based on statistical, data compression, machine learning, philosophy of science, and database paradigms and (2) the most well-known process-oriented frameworks, including Fayyad's [13], CRISP-DM [6], and Reinartz's [37] frameworks. We consider their advantages and limitations analyzing what these approaches are able to explain in the DM process and what they do not. We believe that a reader will notice that each one of the considered foundations-oriented DM frameworks is limited mainly to address one particular type of DM algorithms or describe certain view on the nature of DM. Process-oriented frameworks try to emphasize the issues of integration, iteration, and interactivity in DM. However, none of the frameworks stress the importance of *relevance* in DM research, i.e. they do not emphasize that relevant and applicable results from real world point of view will be achieved. In empirical type of research, relevance usually appears to be associated with utility in practical applications. The so-called "richness of worldly realism" [31] associated with relevance is opposed to "tightness of control" [31] so that at the same level of knowledge they form an iso-epistemic curve representing the fundamental trade-off [23].

In design-science type of research relevance of research is often associated with the consideration of some business need(s), and related environment [19].

In this chapter we try to analyze whether relevance matters to DM research from both perspectives.

We need to acknowledge that some work has been done with regard to the study of *interestingness* of discovered patterns in the context of association rules mining (for example [38]). Yet, even within this particular area, it has been fairly noticed in [5] that there is no consensus on how the interestingness of discovered patterns should be measured, and that most of DM research avoids this thorny way reducing *interestingness* to *accuracy* and *comprehensibility*.

Disregarding the relevance issues, DM frameworks ignore also the issues of DM artifact development and DM artifact use. Here and in the following text by DM *artifact* we mean either "hard/technical" artifacts like DM model, DM technique or its instantiation, collection of DM techniques that are part of DM system or DM embedded solution, or "soft/social" artifacts like some organizational, operational, ethical and methodological rules that focus on different considerations of risks, costs, etc.

In Sect. 3 we first refer to the traditional information system (IS) framework presented in [9] that is widely known in the IS community and is a synthesis of many other frameworks considered before it. This framework takes into account both the use and development aspects beside the technical ones in the IS area. Further we consider more detailed IS frameworks from the use and development perspectives.

In Sect. 4 we introduce our sketch for the new framework for DM research based on the material included in Sects. 2 and 3. We strongly emphasize the relevance aspect of DM research, trying not to neglect the rigor. This means that beside the technological aspects also the organizational and human aspects should be equally taken into account. Thus, our framework for DM research suggests a new turning point for the whole DM research area.

We conclude briefly in Sect. 5 with a short summary and further research topics.

Some materials presented in this chapter are the results of our earlier work [34–36].

2 Review of Some Existing Theoretical Frameworks for DM

It this section we review basic existing foundations-oriented frameworks for DM based on different paradigms, originating from statistics, machine learning, databases, philosophy of science, and granular computing and the most well-known process-oriented frameworks, including Fayyad [13], CRISP-DM [6], and Reinartz's [37] frameworks. We present our conclusions for these groups of DM frameworks and then analyze the state of art in DM research in general.

2.1 Foundations(Theory)-Oriented Frameworks

Frameworks of this type are based mainly on one of the following paradigms: (1) *the statistical paradigms*; (2) *the data compression paradigm* – "compress the dataset by finding some structure or knowledge for it"; (3) *the machine learning paradigm* – "let the data suggest a model" that can be seen as a practical alternative to the statistical paradigms "fit a model to the data"; (4) *the database paradigm* – "there is no such thing as discovery, it is all in the power of the query language" [21]; and also *the inductive databases paradigm* – "locating interesting sentences from a given logic that are true in the database" [3].

The Statistical Paradigms

Generally, it is possible to consider the task of DM from the statistical point of view, emphasizing the fact that DM techniques are applied to larger datasets than it is commonly done in applied statistics [17]. Thus the analysis of appropriate statistical literature, where strong analytical background is accumulated, would solve most DM problems. Many DM tasks naturally may be formulated in the statistical terms, and many statistical contributions may be used in DM in a quite straightforward manner [16].

According to [7] there exist two basic statistical paradigms that are used in theoretical support for DM. The first paradigm is so-called "Statistical experiment". It can be seen from three perspectives: Fisher's version that uses the inductive principle of maximum likelihood, Neyman–E.S. Pearson–Wald's version that is based on the principle of inductive behavior, and the Bayesian version that is based on the principle of maximum posterior probability. An evolved version of the "Statistical experiment" paradigm is the "Statistical learning from empirical process" paradigm [39]. Generally, many DM tasks can be seen as the task of finding the underlying joint distribution of variables in the data. Good examples of this approach would be a Bayesian network or a hierarchical Bayesian model, which give a short and understandable representation of the joint distribution. DM tasks dealing with clustering and/or classification fit easily into this approach.

The second statistical paradigm is called "Structural data analysis" and can be associated with singular value decomposition methods, which are broadly used, for example, in text mining applications.

A deeper consideration of DM and statistics can be found in [14]. Here, we only want to point out that the volume of the data being analyzed and the different educational background of researchers are not the most important issues that constitute the difference between the areas. DM is an applied area of science and limitations in available computational resources is a big issue when applying results from traditional statistics to DM. An important point here is that the theoretical framework of statistics is not concerned much about data analysis as an iterative process that generally includes several

steps. However, there are people (mainly with strong statistical background) who consider DM as a branch of statistics, because many DM tasks may be perfectly represented in terms of statistics.

The Data Compression Paradigm

The data compression approach to DM can be stated in the following way: compress the dataset by finding some structure or knowledge within it, where knowledge is interpreted as a representation that allows coding the data using a fewer amount of bits. For example, the minimum description length (MDL) principle [32] can be used to select among different encodings accounting for both the complexity of a model and its predictive accuracy.

Machine learning practitioners have used the MDL principle in different interpretations to recommend that even when a hypothesis is not the most empirically successful among those available, it may be the one to be chosen if it is simple enough. The idea is in balancing between the consistency with training examples and the empirical adequacy by predictive success as it is, for example, with accurate decision tree construction. Bensusan [2] connects this to another methodological issue, namely that theories should not be ad hoc, that is they should not simply overfit all the examples used to build it. Simplicity is the remedy for being ad hoc both in the recommendations of the philosophy of science and in the practice of machine learning.

The data compression approach has also connections with the rather old Occam's razor principle that was introduced in the fourteenth century. The most commonly used formulation of this principle in DM is "when you have two competing models which make exactly the same predictions, the one that is simpler is better".

Many (if not all) DM techniques can be viewed in terms of the data compression approach. For example, association rules and pruned decision trees can be viewed as ways of providing compression of parts of the data. Clustering can also be considered as a way of compressing the dataset. There is a connection with the Bayesian theory for modeling the joint distribution – any compression scheme can be viewed as providing a distribution on the set of possible instances of the data.

The Machine Learning Paradigm

The machine learning (ML) paradigm, "let the data suggest a model", can be seen as a practical alternative to the statistical paradigm "fit a model to the data". It is certainly reasonable in many situations to fit a small dataset to a parametric model based on a series of assumptions. However, for applications with large volumes of data under analysis the ML paradigm may be beneficial because of its flexibility with a nonparametric, assumption-free nature.

We would like to focus here on the constructive induction approach. Constructive induction is a learning process that consists of two intertwined

phases, one of which is responsible for the construction of the "best" representation space and the second concerns generating hypotheses in the found space [33]. Constructive induction methods are classified into three categories: data-driven (information from the training examples is used), hypothesis-driven (information from the analysis of the form of intermediate hypothesis is used) and knowledge-driven (domain knowledge provided by experts is used) methods. Any kind of induction strategy (implying induction, abduction, analogies and other forms of non-truth preserving and non-monotonic inferences) may potentially be used. However, the focus here is usually on operating higher-level data-concepts and theoretical terms rather than pure data.

Many DM techniques that apply wrapper/filter approaches to combine feature selection, feature extraction, or feature construction processes (as means of dimensionality reduction and/or as means of search for better representation of the problem) and a classifier or other type of learning algorithm may be considered as constructive induction approaches.

The Database Paradigm

A database perspective on DM and knowledge discovery was introduced in [21]. The main postulate of their approach is: "there is no such thing as discovery, it is all in the power of the query language". That is, one can benefit from viewing common DM tasks not as the dynamic operations constructing the new pieces of information, but as operations finding unknown (i.e. not found so far) but existing parts of knowledge.

In [3] an inductive databases framework for the DM and knowledge discovery in databases (KDD) modeling was introduced. The basic idea here is that the data-mining task can be formulated as locating interesting sentences from a given logic that are true in the database. Then knowledge discovery from data can be viewed as querying the set of interesting sentences. Therefore the term "an inductive database" refers to such a type of databases that contains not only data but a theory about the data as well [3].

This approach has some logical connection to the idea of deductive databases, which contain normal database content and additionally a set of rules for deriving new facts from the facts already present in the database. This is a common inner data representation. For a database user, all the facts derivable from the rules are presented, as they would have been actually stored there. In a similar way, there is no need to have all the rules that are true about the data stored in an inductive database. However, a user may imagine that all these rules are there, although in reality, the rules are constructed on demand. The description of an inductive database consists of a normal relational database structure with an additional structure for performing generalizations. It is possible to design a query language that works on inductive databases. Usually, the result of a query on an inductive database is an inductive database as well. Certainly, there might be a need to find a solution about what should

be presented to a user and when to stop the recursive rule generation while querying. We refer an interested reader to [3] for details.

Granular-Computing Approach

Generally, granular computing is a broad term covering theories, methodologies, and techniques that operate with subsets, classes, and clusters (called granules) of a universe. Granular computing concept is widely used in computer science and mathematics. Recently, Zadeh [42] reviewed the concepts of fuzzy information granulation and considered it in the context of human reasoning and fuzzy logic. Lin [28] proposed to use the term "granular computing" to label the computational theory of information granulation. In the same paper Lin introduces a view on DM as a "reverse" engineering of database processing. While database processing organizes and stores data according to the given structure, DM is aimed at discovering the structure of stored data. Lin defines automated DM as "a process of deriving interesting (to human) properties from the underlying mathematical structure of the stored bits and bytes" [28]. Assuming that the underlying mathematical structure of a database relation is a set of binary relations or a granular structure, Lin considers DM as a processing of the granules or structure-granular computing. And then if there is no additional semantics, then the binary relations are equivalence relations and granular computing reduces to the rough set theory [28]. However, since in the DM process the goal is to derive also the properties of stored data, additional structures are imposed. To process these additional semantics, Lin introduces the notion of granular computing in DM context [27].

Yao and Yao [41] applied the granular computing approach to machine learning tasks focusing on covering and partitioning in the process of data mining and showed how the commonly used ID3 and PRISM algorithms can be extended with the granular computing approach.

The Philosophy of Science Paradigm

The categorization of subjectivist and objectivist approaches [4] can be considered in the context of DM. The possibility to compare nominalistic and realistic ontological believes gives us an opportunity to consider data that is under analysis as descriptive facts or constitutive meanings. The analysis of voluntaristic as opposed to deterministic assumptions about the nature of every instance constituting the observed data directs our attitude and understanding of that data. One possibility is to view every instance and its state as determined by the context and/or a law. Another position consists in consideration of each instance as autonomous and independent. An epistemological assumption about how a criterion to validate knowledge discovered (or a model that explains reality and allows making predictions) can be constructed may impact the selection of appropriate DM technique. From the

positivistic point of view such a model-building process can be performed by searching for regularities and causal relationships between the constitutive constructs of a model. And anti-positivism suggests analyzing every individual observation trying to understand it and making an interpretation. Probably some of case-based reasoning approaches can be related to anti-positivism's vision of the reality.

An interesting difference in the views on reality can be found considering ideographic as opposed to nomothetic methodological disputes. The nomothetic school does not see the real world as a set of random happenings. And if so, there must be rules that describe some regularities. Thus, nomothetic sciences seek for establishing abstract (general) laws that describe indefinitely repeatable events and processes. On the contrary, the ideographic sciences are aimed to understand unique and non-recurrent events. They have connection to the ancient doctrine that "all is flux". If everything were always changing, then any generalization intending to be applied for two or more presumably comparable phenomena would never be true. And "averages" of some measures (from the nomothetic way of thinking) usually is not able to represent the behaviour of a single event or entity.

Conclusion on the Theory-Oriented Frameworks

The reductionist approach of viewing DM in terms of one of the theory-oriented frameworks has advantages in strong theoretical background, and easy-formulated problems. The statistics, data compression and constructive induction paradigms have relatively strong analytical background, as well as connections to the philosophy of science. In addition to the above frameworks there exists an interesting microeconomic view on DM [26], where a utility function is constructed and it is tried to be maximized. The DM tasks concerning processes like clustering, regression and classification fit easily into these approaches. Other small-scale yet valuable study related to analysis of interestingness measures of association rules is worth mentioning. Carvalho et al. [5], recognizing the potential gap in estimates of interestingness obtained with objective data-driven measures and true subjective evaluation performed by human, investigated the effectiveness of several data-driven rule interestingness measures by comparing them with the subjective real human interest.

One way or another, we can easily see the exploratory nature of the frameworks for DM. Different frameworks account for different DM tasks and allow preserving and presenting the background knowledge. However, what seems to be lacking in most theory-oriented approaches, are the ways for taking the iterative and interactive nature of the DM process into account [30], and a focus on the utility of DM.

2.2 Process-Oriented Frameworks

Frameworks of this type are known mainly because of works [6,13]. They view DM as a sequence of iterative processes that include data cleaning, feature

transformation, algorithm and parameter selection, and evaluation, interpretation and validation.

Fayyad's View on the Knowledge Discovery Process

Fayyad [13] define KDD as "the nontrivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data". Before focusing on discussion of KDD as a process, we would like to make a note that this definition given by Fayyad is very capacious, it gives an idea what is the goal of KDD and in fact it is cited in many DM related papers in introductory sections. However, in many cases those papers have nothing to do with novelty, interestingness, potential usefulness and validity of patterns which were discovered or could be discovered using proposed in the papers DM techniques.

KDD process comprises many steps, which involve data selection, data preprocessing, data transformation, DM (search for patterns), and interpretation and evaluation of patterns (Fig. 2) [13]. The steps depicted start with the raw data and finish with the extracted knowledge, which was acquired as a result of the KDD process. The set of DM tasks used to extract and verify patterns in data is the core of the process. DM consists of applying data analysis and discovery algorithms for producing a particular enumeration of patterns (or models) over the data. Most of current KDD research is dedicated to the DM step. We would like to clarify that according to this scheme, and some other research literature, DM is commonly referred to as a particular phase of the entire process of turning raw data into valuable knowledge, and covers the application of modeling and discovery algorithms. In industry, however, both knowledge discovery and DM terms are often used as synonyms to the entire process of getting valuable knowledge.

Nevertheless, this core process of search for potentially useful patterns typically takes only a small part (estimated at 15–25%) of the effort of the overall KDD process. The additional steps of the KDD process, such as data preparation, data selection, data cleaning, incorporating appropriate prior

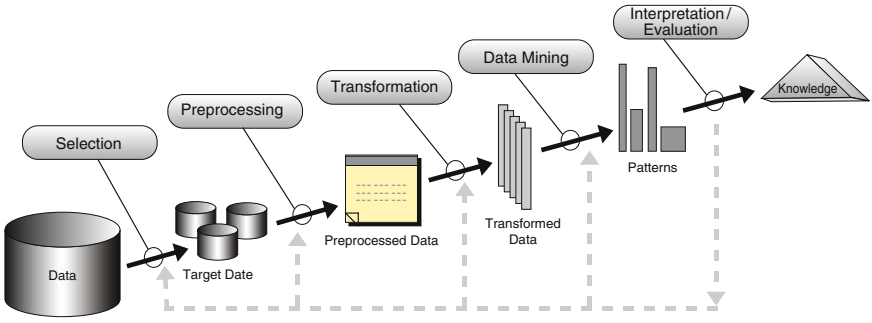


Fig. 2. Basic steps of the KDD process [13]

knowledge, and proper interpretation of the results of mining, are also essential to derive useful knowledge from data.

In our opinion the main problem of the framework presented in Fig. 2 is that all KDD activities are seen from "inside" of DM having nothing to do with the relevance of these activities to practice (business).

CRISP-DM: CRoss Industry Standard Process for Data Mining

The life cycle of a DM project according to the CRISP-DM model (Fig. 3) consists of six phases (though the sequence of the phases is not strict and moving back and forth between different phases normally happens) [6]. The arrows indicate the most important and frequent dependencies between phases. And the outer circle in the figure denotes the cyclic nature of DM – a DM process continues after a solution has been deployed. If some lessons are learnt during the process, some new and likely more focused business questions can be recognized and subsequently new DM processes will be launched.

We will not stop at any phase of CRISP-DM here since it has much overlapping with Fayyad's view and with the framework that is considered in the next section and discussed in more details. However, we would like to notice that the KDD process is put now in a way into some business environment that is represented by the business *understanding* and *deployment* blocks.

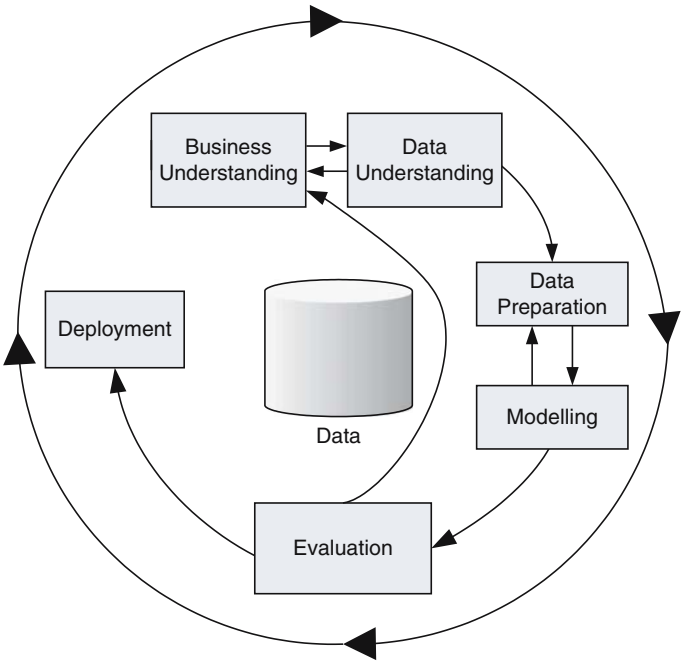


Fig. 3. CRoss industry standard process for data mining [6]

Reunartz's View

Reinartz's framework [37] follows CRISP-DM with some modifications (Fig. 4), introducing a data exploration phase and explicitly showing the accumulation of the experience achieved during the DM/KDD processes. The business-understanding phase is aimed to formulate business questions and translate them into DM goals. The data-understanding phase aims at analyzing and documenting the available data and knowledge sources in the business according to the formulated DM goals and providing initial characterization of data. The data preparation phase starts from target data selection that is often related to the problem of building and maintaining useful data warehouses. After selection, the target data is preprocessed in order to reduce the level of noise, preprocess the missing information, reduce data, and remove obviously redundant features. The data exploration phase aims at providing the first insight into the data, evaluate the initial hypotheses, usually, by means of descriptive statistics and visualization techniques. The DM phase covers selection and application of DM techniques, initialization and further calibration of their parameters to optimal values. The discovered patterns that may include a summary of a subset of the data, statistical or predictive models of the data, and relationships among parts of the data are locally evaluated. The evaluation and interpretation phase aims at analyzing the discovered patterns, determining the patterns that can be considered as the new knowledge, and drawing conclusions about the whole discovery process as well. The deployment phase aims at transferring DM results that meet the success criteria into the business [37].

We think that the main problem with CRISP-DM and Reinartz's frameworks is that they assume that the DM artifact is ready to be applied and easy

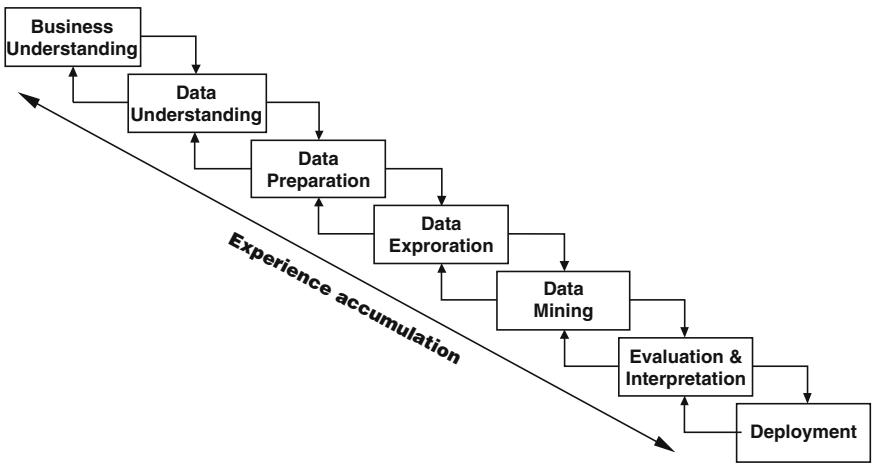


Fig. 4. Knowledge discovery process: from problem understanding to deployment (adapted from [37])

to be deployed and used. Therefore, the development and use processes are almost disregarded in these frameworks though being embedded in an implicit way. Consequently, it is hard to see what the most crucial success factors of a DM project are.

2.3 Conclusions on the Considered Frameworks

With respect to foundations-oriented frameworks, some DM researchers argue for the lack of an accepted fundamental conceptual framework or a paradigm for DM research and consequently for the need of some consensus on the fundamental concepts. Therefore, they try to search for some mathematical bricks for DM. And the approaches based on granular and rough computing present good examples of such attempts. However, others may think that the current diversity in theoretical foundations and research methods is a good thing and also it might be more reasonable to search for an umbrella-framework that would cover the existing variety.

Another direction of research could lie in addressing data to be mined, DM models, and reality views through the prism of the philosophy of science paradigm, that includes consideration of nominalistic vs. realistic ontological beliefs, voluntaristic vs. deterministic assumptions about the nature of every instance constituting the observed data, subjectivist vs. objectivist approaches to model construction, ideographic vs. nomothetic view at reality; and epistemological assumptions about how a criterion to validate knowledge discovered can be constructed.

SPSS whitepaper [6] states that "Unless there's a method, there's madness". It is accepted that just by pushing a button someone should not expect useful results to appear. An industry standard to DM projects CRISP-DM is a good initiative and a starting point directed towards the development of DM meta-artifact (methodology to produce DM artifacts). However, in our opinion it is just one guideline, which is in too general-level, that every DM developer follows with or without success to some extent. Process-oriented frameworks try to address the iterativeness and interactiveness of the DM process. However, the development process of DM artifact and use of that artifact are poorly emphasized.

Lin in Wu et al. [40] notices that a new successful industry (as DM) can follow consecutive phases (1) discovering a new idea, (2) ensuring its applicability, (3) producing small-scale systems to test the market, (4) better understanding of the new technology and (5) producing a fully scaled system. At the present moment there are several dozens of DM systems, none of which can be compared to the scale of a DBMS system. This fact according to Lin indicates that we are still at the third phase with the DM area.

Further Lin in Wu et al. [40] claims that the research and development goals of DM are quite different, since research is knowledge-oriented while development is profit-oriented. Thus, DM research is concentrated on the development of new algorithms or their enhancements but the DM developers in

domain areas are aware of cost considerations: investment in research, product development, marketing, and product support. We agree that this clearly describes the current state of the DM field. However, we believe that the study of the DM development and DM use processes is equally important as the technological aspects and therefore such research activities are likely to emerge *within* the DM field. In fact, the study of development and use processes was recognized to be of importance in the IS field many years ago, and it has resulted in introduction of several interesting IS research frameworks, some of which are discussed in the next section.

3 Information Systems Research Frameworks

Information Systems (IS) are powerful instruments for organizational problem solving through formal information processing [29]. It is very common, especially in the US to use the term Management Information Systems (MIS) as a synonym for IS. From the first definitions of MIS in the first half of 1970s it has been developed as a discipline of its own having unique identity, core journals and conferences, and an official association with thousands members worldwide [1]. During the years different IS research frameworks have been defined and used. We represent in this chapter first the very traditional ones and then more resent ones for the subareas of IS use and development.

3.1 The Traditional Information Systems Perspective

The traditional framework presented by Ives et al. [22] is widely known in the IS community. They used five earlier research models as a base when they developed their own (Fig. 5). In their framework an IS is considered in an organizational environment that is further surrounded by an external environment. According to their framework an IS itself includes three environments: a user environment, an IS development environment, and an IS operations environment. There are accordingly three processes through which an IS has interaction with its environments: the use process, the development process, and the operation process.

The external environment [22] includes legal, social, political, cultural, economic, educational, resource and industry/trade considerations and the organizational environment [22] is marked by the organizational goals, tasks, structure, volatility, and management philosophy/style. In their model the user environment is including and surrounding the primary users of the IS, the development environment consists of wide range of things from the technical (as IS development methods and techniques) to the organizational (as organization and management of IS development and maintenance) and human-oriented ones (as IS design personnel and their characteristics). The IS operations environment [22] incorporates the resources necessary for IS operations. The major components include software, hardware, database, procedures/documentation, organization and management of IS operations, and

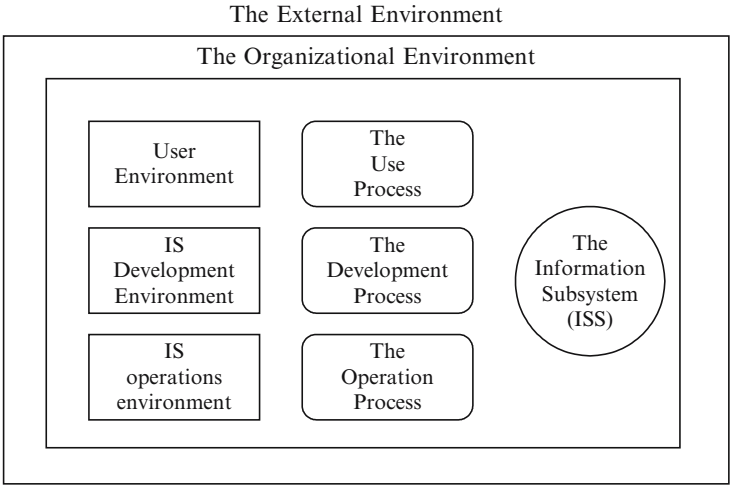


Fig. 5. A framework for IS research [22]

the operations personnel. However, in this chapter, we focus on the user and IS development environments and the corresponding processes.

The research framework is thus very broad resulting in various different research questions and settings. The most extensive ones relate to the effects of IS onto its organizational and external environments. Many research paradigms have been suggested and used in the IS discipline. Currently, Hevner et al. [19] suggest that two paradigms should be recognized within the research in the IS discipline. These are the behavioural-science paradigm and the design-science paradigm. According to the authors, the behavioural science paradigm tries "to develop and verify theories that explain or predict human or organizational behaviour". This paradigm is naturally the most broadly applied in the use process related topics. They continue that "The design-science paradigm seeks to extend the boundaries of human and organizational capabilities by creating new and innovative artifacts" [19]. This second paradigm is the most natural in the IS development related topics where the new user and development environments are planned and experimented with. Some others as e.g. Iivari et al. [20] call the IS development process related research as a constructive type of research because it is based on the philosophical belief that development always involves creation of some new artifacts – conceptual (models, frameworks) or more technical artifacts (software implementations).

3.2 The IS Success Model

As one of IS user environment related models we represent in this section the IS success model developed by DeLone and McLean in 1992 [10]. They report in their ten-year update paper [11] that it has gained wide popularity

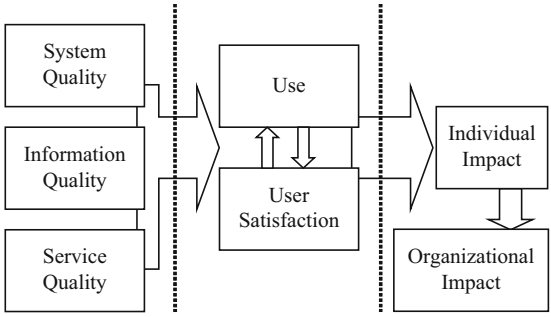


Fig. 6. Adapted from D&M IS Success Model [10] and updated D&M IS Success Model [11]

with nearly 300 articles published in refereed journals referencing their original paper. Because this IS success model is so well known we picked it up as an example of user environment related IS research models. An adapted version of the model is presented in Fig. 6 (it is very similar to the one in http://business.clemson. edu/ISE/).

The original model was developed to "aid in the understanding of the possible causal interrelationships among the dimensions of success and to provide a more parsimonious exposition of the relationships". The investments into information systems are huge every year. Thus it is natural to try to evaluate the effectiveness of those expenditures. The model raises information quality, service quality, and systems quality as key ingredients behind the user satisfaction and the use of IS. These have been found to have essential positive effect to individual impact leading to the organizational impact of information systems.

3.3 The IS Development Environment

The IS development environment is needed to develop and maintain the IS in use. Beside organizing and managing the development and maintenance processes these processes require several kinds of resources: not only the technical ones, as methods and techniques, but also human as motivated people with good enough education for the job. It is natural that in this compound human, organization, and technology complex there is a need to have diversified research methods. One such proposal that has been referred to quite often in the IS literature is the one represented below.

In [25] system development itself is considered as a central part of a multimethodological information systems research cycle (Fig. 7).

Theory building involves discovery of new knowledge in the field of study, however it rarely contributes directly to practice. Nevertheless, the new theory often (if not always) needs to be tested in the real world to show its validity, recognize its limitations and make refinements according to observations

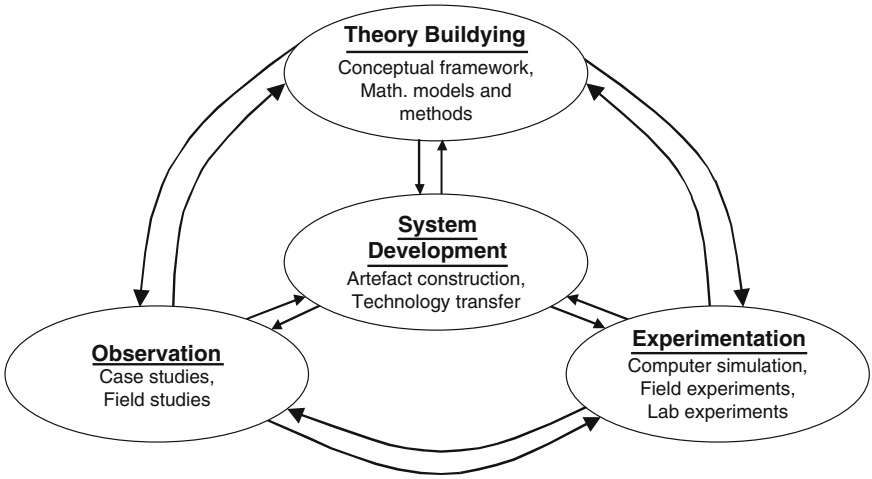


Fig. 7. A multimethodological approach to the construction of an artifact for DM (adapted from [25])

made during its application. According to reasoning research methods can be subdivided into basic and applied research, as naturally both are common for any large system development project. A proposed theory leads to the development of a prototype system in order to illustrate the theoretical framework on the one hand, and to test it through experimentation and observation with subsequent refinement of the theory and the prototype in an iterative manner. Such a view presents the framework of IS as a complete, comprehensive and dynamic research process. It allows multiple perspectives and flexible choices of methods to be applied during different stages of the research process.

In fact, although Dunkel et al. [12] concluded that there is a need and opportunity for computing systems research and development in the context of DMS development, almost 9 years later, to the best of our knowledge there are no significant research papers published in this direction.

4 Our New Research Framework for DM Research

It was mentioned in Sect. 2 that a new successful industry can follow five consecutive phases and that DM is presumably currently at the third phase. The IS discipline on the other hand has during its 30+ year existence been able to develop to the fifth level. One of the key aspects helping the IS area development might have been that it has taken seriously into account human and organizational aspects beside the technological ones. This has raised its relevance and thus attracted more broad interests to support research in the IS area. We see raising the relevance of DM research as an essential aspect towards its more broad applicability, leading to new previously unknown research topics in the DM area.

In this section we suggest a new research framework which includes parts having similarities with the research frameworks applied in the IS discipline. Analogically with the IS research discussion in Sect. 3 we distinguish three environments for a DM system (DMS): the user, development, and operation environment but discuss in this chapter only the first two. We start from these environments in Sects. 4.1 and 4.2 and finish with presenting our new research framework for DM in Sect. 4.3.

4.1 The DMS User Environment

Piatetsky-Shapiro in Wu et al. [40] gives a good example that characterizes the whole area of current DM research: "we see many papers proposing incremental refinements in association rules algorithms, but very few papers describing how the discovered association rules are used". DM is fundamentally application-oriented area motivated by business and scientific needs to make sense of mountains of data [40]. A DMS is generally used to support or do some task(s) by human beings in an organizational environment (see Fig. 8) both having their desires related to DMS. Further, the organization has its own environment that has its own interest related to DMS, for example that privacy of people is not violated.

A similar approach to that with IS is needed with DMS to recognize the key factors of successful use and impact of DMS both at the individual and organizational levels. Questions like (1) how the system is used, and also supported and evolved, and (2) how the system impacts and is impacted by the contexts in which it is embedded are important also in the DMS context. The first efforts in that direction are the ones presented in the DM Review magazine [8,18], referred below. We believe that such efforts should be encouraged in DM research and followed by research-based reports.

Coppock [8] analyzed, in a way, the failure factors of DM-related projects. In his opinion they have nothing to do with the skill of the modeler or the quality of data. But those do include these four (1) persons in charge of the project did not *formulate actionable insights*, (2) the sponsors of the work

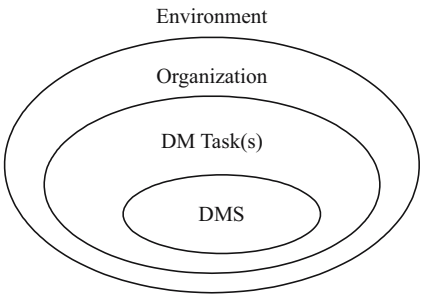


Fig. 8. DMS in the kernel of an organization

did not *communicate the insights* derived to key constituents, (3) the results *don't agree with institutional truths*, and (4) the project never had a *sponsor and champion*. The main conclusion of Coppock's analysis is that, similar to an IS, the leadership, communication skills and understanding of the culture of the organization are not less important than the traditionally emphasized technological job of turning data into insights.

Hermiz [18] communicated his beliefs that there are four critical success factors for DM projects (1) having a clearly articulated business problem that needs to be solved and for which DM is a proper tool; (2) insuring that the problem being pursued is supported by the right type of data of sufficient quality and in sufficient quantity for DM; (3) recognizing that DM is a process with many components and dependencies – the entire project cannot be "managed" in the traditional sense of the business word; (4) planning to learn from the DM process regardless of the outcome, and clearly understanding, that there is no guarantee that any given DM project will be successful. Thus it seems possible that there are also some DMS specific questions that have not maybe been considered from those viewpoints in the IS discipline.

Lin in Wu et al. [40] notices that in fact there have been no major impacts of DM on the business world echoed. However, even reporting of existing success stories is important. Giraud-Carrier [15] reported 136 success stories of DM, covering nine business areas with 30 DM tools or DM vendors referred. Unfortunately, there was no deep analysis provided that would summarize or discover the main success factors and the research should be continued.

4.2 The DMS Artifact Development Environment

If a stated research problem includes a verb like introduce, improve, maintain, cease, extend, correct, adjust, enhance and so on, the study likely belongs to the area of constructive research. These are the kind of actions that researchers in the area of DM perform, when they are developing new theories and their applications as new artifacts to the use of persons and organizations. When a researcher him/herself is acting also as a change agent developing the artifact to an organization he is applying the action research approach.

But how to conceive, construct, and implement an artifact? It is obvious that in order to construct a good artifact background knowledge is needed both about the artifact's components, that are the basic data mining techniques in the DM context and about components' cooperation, that are commonly selection and combination techniques in the DM context. Beside this the developer needs to have enough background knowledge about the human and organizational environment where the artifact is going to be applied. As discussed in Sect. 3 the design science approach is the one concentrating on this kind of research questions. Both these: the action research and design science approach to artifact creation and the evaluation process [24] are presented in Fig. 9.

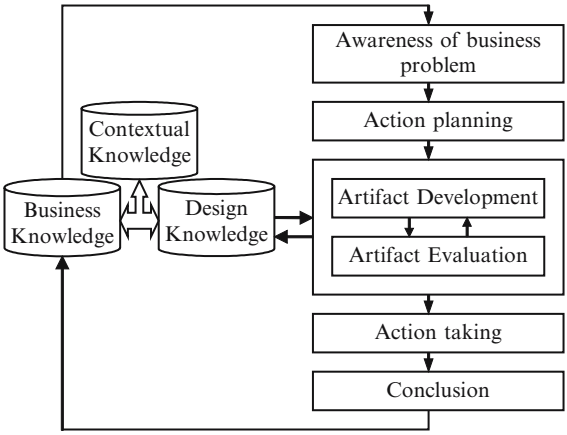


Fig. 9. The action research and design science approach to artifact creation

As discussed in Sect. 3 with Nunamaker's multimethodological approach it is essential that artifacts developed are also experimented with and analyzed using observation type of research. The evaluation process is a key part of any constructive research. This is also true when the artifact developed during research is a DMS (or its prototype). Usually, the experimental approach is used to evaluate a DM artifact. The experimental approach, however, can be beneficial for theory testing and can result in new pieces of knowledge thus contributing to the theory-creating process, too.

A "goodness" criterion of a built theory or an artifact can be multidimensional and it is sometimes difficult to be defined because of mutual dependencies between the compromising variables. However, it is more or less easy to construct a criterion based on such estimates as accuracy of a built model and its performance. On the other hand, it is more difficult or even impossible to include into a criterion such important aspects as interpretability of the artifact's output because estimates of such kind are usually subjective and can be evaluated only by the end-users of a system. This does not eliminate the necessity to research also these topics which are important for users to see the results having relevance.

4.3 New DM Research Framework

Heavner et al. [19] presented a conceptual framework for understanding, conducting and evaluation of the IS research. We adapt their framework to the context of DM research (see Fig. 10). The framework combines together the behavioral-science and design-science paradigms and shows how research rigor and research relevance can be explained, evaluated, and balanced.

We follow Hevner et al. [19] with the description of the figure, emphasizing issues important in DM. The environment defines not only the data

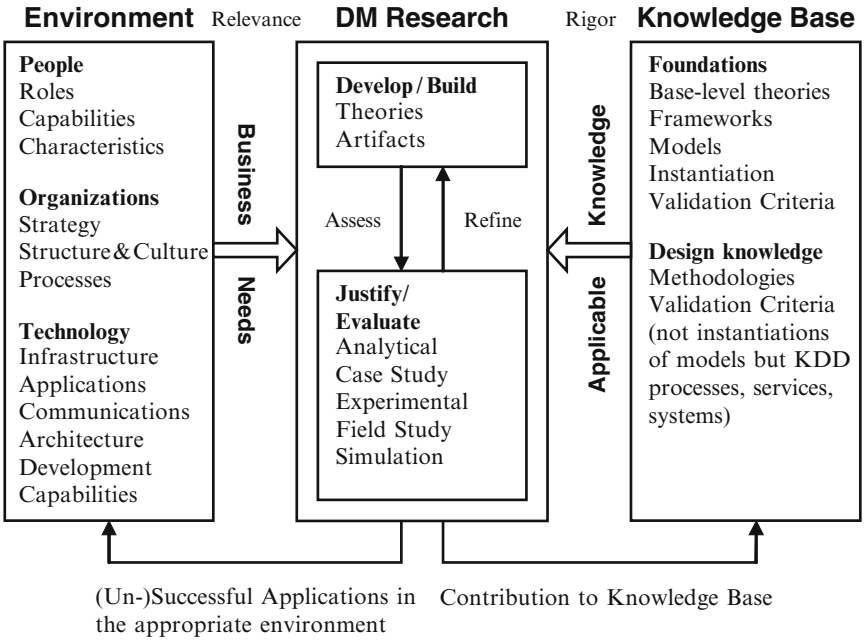


Fig. 10. New research framework for DM research (adapted from [19])

that represents the problem to be mined but people, (business) organizations, and their existing or desired technologies, infrastructures, and development capabilities. Those include the (business) goals, tasks, problems, and opportunities that define (business) needs, which are assessed and evaluated within the context of organizational strategies, structure, culture, and existing business processes. Those research activities that are aimed at addressing business needs contribute to the relevance of research.

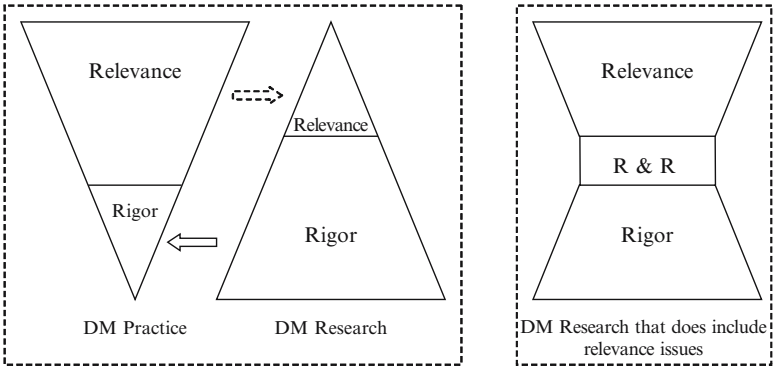
Driven by the business needs, DM research can be conducted in two complementary phases. Behavioral science would guide research through the *development* and *justification* of theories that describe, explain or predict some phenomena associated with the business need being addressed. Design science enables the *building* and *evaluation* of artifacts being developed to address the business need. It is generally accepted that the goal of behavioral science research is truth and the goal of design science research is utility. However, Hevner et al. [19] were likely the first who argued that truth and utility are inseparable – "truth informs design and utility informs theory". They conclude that "an artifact may have utility because of some as yet undiscovered truth. A theory may yet to be developed to the point where its truth can be incorporated into design. In both cases, research assessment via the justify/evaluate activities can result in the identification of weaknesses in the theory or artifact and the need to refine and reassess. The refinement and reassessment process is typically described in future research directions." [19]

The knowledge base provides foundations and methodologies for research (and development) activities. Prior DM research and development and results from reference disciplines (statistics, machine learning, AI, etc.) provide foundational theories, frameworks, models, methods, techniques and their instantiations used in the develop/build phase of research. Methodologies should provide guidelines and techniques for the justify/evaluate phase. Rigor is achieved by appropriately applying existing foundations and methodologies.

5 Discussion and Conclusions

In this chapter we first considered several existing frameworks for DM and their advantages and limitations. Second, we considered a traditional IS framework and two subframeworks: one for the IS user environment and another for the IS development environment. Based on these two we suggested our new research framework for DM. It imports research questions and topics from the IS discipline into the DM area trying to take benefit of the fact that the long developed IS discipline can help the maturing DM research area to raise the relevance of its research and thus its practical importance for people, organizations, and their surroundings.

Figure 11a presents our understanding of the current situation with DM research. The left triangle presents the current DM practice situation where almost merely the relevance aspects, i.e. utility are dominating. The right triangle presents the current DM research situation that is heavily dominated by rigor aspects and almost no attention is paid to DM research relevance. The lower arrow between DM research and practice is solid because some amount of rigor DM research results are flowing to the practice at least through software applications. The upper arrow is dashed because our understanding of the situation is that too seldom DM research takes practice related aspects into



(a) Current disbalance in DM research **(b)** Balanced DM research

Fig. 11. Rigor and relevance aspects of DM research

account and thus the exchange between DM practice and DM research is not as fruitful as it might be.

Even those relevance issues that are recognized within community of DM practitioners or let us say the (current or potential) users of DM systems, DM solutions and DM services, are not studied appropriately from scientific point of view and therefore we can rarely see the transfer of scientific knowledge (and in many cases even valuable feedback) from DM practice to DM research.

Thus, our believe is that within DM research community there should be DM research dealing purely with rigor issues, DM research dealing mostly with relevance issues and, likely the most challenging part of DM research efforts dealing with rigor/relevance aspects (Fig. 11b). With regard to this belief we recognize two important aspects (1) those who practice DM should be well-motivated to share their expertise and scientific insights into relevance issues in DM, and that is not less important, (2) DM research community should be interested in conducting and publishing *academic* research of relevance issues in DM. However, our analysis show that currently DM research often does take relevance into account only from empirical research point of view with regard to possible variety of dataset characteristics, but in most of the cases does not account for many important environment aspects (people, organization etc), i.e. relevance concept originating from design science.

We considered DMSs as a special kind of ISs which have not yet been considered closely enough, in our opinion, from the use and development perspectives. After discussing these two DMS environments, we presented our new DM research framework, which aims at better balancing between the rigor and relevance constituents of research also in the DM area.

In this work we have not provided any examples to demonstrate the applicability of the proposed framework. We have not tried also to describe all the essential issues at the very detailed level, leaving this maturation for further research. However, we believe that our work could be helpful in turning the focus of DM research into a more balanced direction. We see this important from the point of view of raising DM first among those technologies which are able to produce competitive advantage and later to be developed to be one of everyday mainline technologies.

We hope that our work could raise a new wave of interest to the foundations of DM and to the analysis of the DM field from different perspectives, maybe similar to IS and ISD. This can be achieved by the building of knowledge networks across the field boundaries (DM and IS), e.g. by organizing workshops that would include such important topics as DM success, DM costs, DM risks, DM life cycles, methods for analyzing systems, organizing and codifying knowledge about DM systems in organizations, and maximizing the value of DM research. We hope also that meta-level research in DM, directed to the study of current situation and trends and possibilities of further development of the field (as our study does) will be recognized as important and valuable type of research.

Acknowledgements

This research is partly supported by the COMAS Graduate School of the University of Jyvskyl, the Academy of Finland, and by the Science Foundation Ireland under Grant No. S.F.I.-02IN.1I111. We are thankful to the reviewers of this chapter for their valuable comments and suggestions.

References

1. Benbasat, I., Zmud, R. W.: The Identity Crisis Within the IS Discipline: Defining and Communicating the Discipline's Core Properties, *MIS Quarterly*, **27**(2), 2003, 183–194
2. Bensusan, H.: *Is Machine Learning Experimental Philosophy of Science?*, Technical report, Bristol, UK, 2000
3. Boulicaut, J.-F., Klemettinen, M., Mannila, H.: Modeling KDD Processes Within the Inductive Database Framework, *DaWaK'99: Proceedings of the First International Conference on Data Warehousing and Knowledge Discovery*, Springer, Berlin Heidelberg New York, London, 1999, 293–302
4. Burrell, G., Morgan, G.: *Sociological Paradigms and Organizational Analysis*, Heinemann, London, UK, 1979
5. Carvalho, D. R., Freitas, A. A., Ebecken, N. F. F.: Evaluating the Correlation Between Objective Rule Interestingness Measures and Real Human Interest, *PKDD*, 2005, 453–461
6. Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., Wirth, R.: *CRISP-DM 1.0 – Step-by-Step Data Mining Guide*, The CRISPDM Consortium/SPSS Inc., 2000, Availabe on http://www.crisp-dm.org
7. Coppi, R.: A Theoretical Framework for Data Mining: The Informational Paradigm, *Computational Statistics and Data Analysis*, **38**(4), 2002, 501–515
8. Coppock, D. S.: Data Mining and Modeling: So You have a Model, Now What?, *DM Review*, February 2003
9. Davis, G. B.: Information Systems Conceptual Foundations: Looking Backward and Forward, *Organizational and Social Perspectives on Information Technology*, Kluwer, Boston, 2000, 61–82
10. DeLone, W. H., McLean, E. R.: Information Systems Success: The Quest for the Dependent Variable, *Information Systems Research*, **3**(1), 1992, 60–95
11. DeLone, W. H., McLean, E. R.: The DeLone and McLean Model of Information Systems Success: A Ten-Year Update, *Journal of MIS*, **19**(4), 2003, 9–30
12. Dunkel, B., Soparkar, N., Szaro, J., Uthurusamy, R.: Systems for KDD: From concepts to practice, *Future Generation Computer Systems*, **13**(2–3), 1997, 231–242
13. Fayyad, U. M.: Data Mining and Knowledge Discovery: Making Sense Out of Data, *IEEE Expert: Intelligent Systems and Their Applications*, **11**(5), 1996, 20–25
14. Friedman, J. H.: Data Mining and Statistics: What's the Connection?, *Proceedings of the 29th Symposium on the Interface* (Scott, D. Ed.), 1999
15. Giraud-Carrier, C.: *Success Stories in Data/Text Mining*, Technical report, Brigham Young University, 2004, (An updated version of an ELCA Informatique SA White Paper)

16. Hand, D. J.: Data Mining: Statistics and More?, *The American Statistician*, **52**, 1998, 112–118
17. Hand, D. J.: Statistics and Data Mining: Intersecting Disciplines, *SIGKDD Explorations*, **1**, 1999, 16–19
18. Hermiz, K. B.: Critical Success Factors for Data Mining Projects, *DM Review*, February 1999
19. Hevner, A. R., March, S. T., Park, J., Ram, S.: Design Science in Information Systems Research, *MIS Quarterly*, **26**(1), 2004, 75–105
20. Iivari, J., Hirschheim, R., Klein, H. K.: A Paradigmatic Analysis Contrasting Information Systems Development Approaches and Methodologies, *Information Systems Research*, **9**(2), 1998, 164–193
21. Imielinski, T., Mannila, H.: A Database Perspective on Knowledge Discovery, *Communications of ACM*, **39**(11), 1996, 58–64
22. Ives, B., Hamilton, S., Davis, G. B.: A Framework for Research in Computer-based Management Information Systems, *Management Science*, **26**(9), 1980, 910–934
23. Järvinen, P.: *Research methods*, Opinpaja, Tampere, Finland, 2002, (http://www.uta.fi/pj/)
24. Järvinen, P.: *Action Research as an Approach in Design Science*, Technical Report TR D-2005-2, Department of Computer Science, University of Tampere, Finland, 2005
25. Nunamaker, J. F., Chen, M., Purdin, T. D. M.: Systems Development in Information Systems Research, *Journal of Management Information Systems*, **7**(3), 90/91, 89–106
26. Kleinberg, J., Papadimitriou, C., Raghavan, P.: A Microeconomic View of Data Mining, *Data Mining and Knowledge Discovery*, **2**(4), 1998, 311–324
27. Lin, T. Y.: Granular Computing of Binary Relations I: Data Mining and Neighborhood Systems, *Rough Sets and Knowledge Discovery* (Polkowski, L. Skowron A., Eds.), Physica, Heidelberg, 1998, 107–140
28. Lin, T. Y.: Data Mining: Granular Computing Approach, *PAKDD'99: Proceedings of third Pacific-Asia Conference, Methodologies for Knowledge Discovery and Data Mining*, 1999, 24–33
29. Lyytinen, K.: Different Perspectives on Information Systems: Problems and Solutions, *ACM Computing Surveys*, **19**(1), 1987, 5–46
30. Mannila, H.: Theoretical Frameworks for Data Mining, *SIGKDD Explorations*, **1**(2), 2000, 30–32
31. Mason, R.: Experimentation and Knowledge – A Paradigmatic Perspective, *Knowledge: Creation, Diffusion, Utilization*, **10**(1), 1988, 3–24
32. Mehta, M., Rissanen, J., Agrawal, R.: MDL-Based Decision Tree Pruning, *KDD'95*, 1995, 216–221
33. Michalski, R. S.: Seeking Knowledge in the Deluge of Facts, *Fundamenta Informaticae*, **30**(3–4), 1997, 283–297
34. Pechenizkiy, M., Puuronen, S., Tsymbal, A.: The Iterative and Interactive Data Mining Process: The ISD and KM Perspectives, *FDM'04: Proceedings of Foundations of Data Mining Workshop*, 2004, 129–136
35. Pechenizkiy, M., Puuronen, S., Tsymbal, A.: Competitive Advantage from Data Mining: Lessons Learnt in the Information Systems Field, *DEXA'05 Workshop: Philosophies and Methodologies for Knowledge Discovery (PMKD'05)*, IEEE CS Press, New York, 2005, 733–737, (Invited Paper)

36. Pechenizkiy, M., Puuronen, S., Tsymbal, A.: Why Data Mining Does Not Contribute to Business?, *DMBiz'05: Proceedings of Data Mining for Business Workshop*, 2005, 67–71
37. Reinartz, T.: *Focusing Solutions for Data Mining: Analytical Studies and Experimental Results in Real-World Domains*, Springer, Berlin Heidelberg New York, 1999
38. Tan, P.-N., Kumar, V., Srivastava, J.: Selecting the Right Objective Measure for Association Analysis, *Information Systems*, **29**(4), 2004, 293–313
39. Vapnik, V. N.: *The Nature of Statistical Learning Theory*, Springer, Berlin Heidelberg New York, 1995
40. Wu, X., Yu, P. S., Piatetsky-Shapiro, G., Cercone, N., Lin, T. Y., Kotagiri, R., Wah, B. W.: Data Mining: How Research Meets Practical Development?, *Knowledge and Information Systems*, **5**(2), 2003, 248–261
41. Yao, J. T., Yao, Y. Y.: A Granular Computing Approach to Machine Learning, *FSKD'02: Proceedings of the First International Conference on Fuzzy Systems and Knowledge Discovery*, 2002, 732–736
42. Zadeh, L. A.: Toward a Theory of Fuzzy Information Granulation and its Centrality in Human Reasoning and Fuzzy Logic, *Fuzzy Sets Systems*, **90**(2), 1997, 111–127

E-Action Rules

Li-Shiang Tsay[1] and Zbigniew W. Raś[2,3]

[1] North Carolina A&T State University, School of Technology, Greensboro, NC 27411, USA
ltsay@ncat.edu
[2] University of North Carolina, Computer Science Department, Charlotte, NC 28223, USA
ras@uncc.edu
[3] Polish–Japanese Institute of Information Technology, ul. Koszykowa 86, 02-008, Warsaw, Poland
ras@pjwstk.edu.pl

Summary. The ability to discover useful knowledge hidden in large volumes of data and to act on that knowledge is becoming increasingly important in today's competitive world. Action rules were proposed to help people analyze discovered patterns and develop a workable strategy for actions [10]. A formal definition of an action rule was independently proposed in [4]. These rules have been investigated further in [11, 12].

Action rules are constructed from certain pairs of classification rules extracted earlier from the same decision table, each one defining different preferable classes. Attributes in a database are divided into two groups: stable and flexible. Flexible attributes provide a tool for making hints to a user what changes within some values of flexible attributes are needed to re-classify group of objects, supporting action rule, to another decision class.

Classical action rules only involve flexible attributes listed in both classification rules from which an action rule is constructed. The values of the common stable attributes listed in both rules are used to create an action rule but they are not listed in the expression describing that rule. Because of that, there are many options in actual real-life implementations of them. In this chapter, we propose a new class of action rules, called E-Action rules, to solve this issue. Our experience shows that an E-Action rule is more meaningful than a classical action rule or an extended action rule [11] because it is easy to interpret, understand, and apply by users.

1 Introduction

The knowledge extracted from data can provide a competitive advantage in support of decision-making. Finding useful rules is an important task of knowledge discovery from data. People usually evaluate a pattern value based on its interestingness. There are two types of interestingness measure: objective

L.-S. Tsay and Z.W. Raś: *E-Action Rules*, Studies in Computational Intelligence (SCI) **118**, 277–288 (2008)
www.springerlink.com

and subjective (see [1, 6, 13, 14]). Subjective measure is when the judgment is made by people and objective measure is basically when the judgment is made by computer derived methods or strategies. Generally, they evaluate the rules based on their quality and similarity between them.

A rule is deemed actionable, if users can take action to gain an advantage based on that rule [6]. This definition, in spite of its importance, is too vague and it leaves open door to a number of different interpretations of actionability. In order to narrow it down, a new class of rules (called action rules) constructed from certain pairs of classification rules, has been proposed in [10]. A formal definition of an action rule was independently proposed in [4]. These rules have been investigated further in [11].

Classical action rules involve only flexible attributes listed in both classification rules from which the action rule is constructed. Extended action rule is a significant improvement of a classical action rule because of the constraints placed on values of attributes listed only in one of these rules. But, still extended action rules do not include the values of common stable attributes listed in rules from which they are constructed. This implies that the domain of an action rule does not reflect correctly to what class of objects it can be successfully applied. To solve this problem, a new class of rules, called E-Action rules, is proposed. E-action rules extract actionability knowledge among pairs of classification rules in a more accurate way.

E-action rules are useful is many fields, including medical diagnosis and business. In medical diagnosis, classification rules can explain the relationships between symptoms and type of sickness as well as predict the diagnosis of a new patient. Extended action rules are useful in providing suggestions for modifying some symptoms in order to recover from an illness. In business, classification rules can distinguish the good customers from the bad ones. E-action rules can provide specific actions that can be taken by decision-makers to re-classify customers.

The strategy for generating action rules proposed in [11] is significantly improved in the system *DEAR-2* presented in this chapter. It consists of three steps. The first step is to partition the rules into equivalence classes with respect to the values of the decision attribute. In the second step, we use a recursive algorithm to dynamically build a tree structure partitioning each class of rules based on values of their stable attributes. When any two rules have the same stable attribute values, they are placed in the same class. In the final step, instead of comparing all pairs of rules, only pairs of rules belonging to some of these equivalent classes have to be compared in order to construct extended action rules. This strategy significantly reduces the number of steps needed to generate action rules in comparison to the strategy (called *DEAR*) proposed in [11].

In this chapter, we present a new definition of E-Action rules to enhance the action rule and the extended action rules.

2 Motivation for E-Action Rules

To give an example justifying the need of action rules, let us assume that a number of customers decided to close their accounts at one of the banks. To find the cause of their action, possibly the smallest and the simplest set of rules describing all these customers is constructed. Let us assume that $\{Nationality, AccountType\}$ is a list of stable attributes, $\{InterestRate\}$ is a flexible attribute, and $\{Status\}$ is a decision attribute. Additionally, we assume that the extracted classification rule r_1 is represented as

$$[Nationality, Asian] \wedge [InterestRate, 1\%] \longrightarrow [Status, closedAccount].$$

Next,we search for a new set of rules, describing groups of customers who did not leave the bank. A rule r_2 is represented as

$$[Nationality, Asian] \wedge [AccountType, savings] \wedge \\ [InterestRate, 1.5\%] \longrightarrow [Status, openAccount].$$

The classification parts of both rules are quite similar. Their common stable attribute *Nationality* has the same value *Asian*. Now, by comparing these two rules, we want to find out not only the cause why these accounts stay closed for certain customers but also formulate an action that, if undertaken by the bank, may influence these customers to re-open their accounts. A classical extended action rule is constructed from these two rules and is represented by the expression:

$$[AccountType, savings] \wedge [InterestRate, 1\% \longrightarrow 0.5\%] \\ \Longrightarrow [Status, closedAccount \longrightarrow openAccount].$$

It says: if a customer has savings account and its current interest rate of 1% is increased to 1.5%, then he may move from a group of closed account customers to a group that keeps their accounts open. However, if the *Nationality* of a closed account customer is not an *Asian* then this customer will not be supported by that rule. Therefore, even if the bank extends an offer of 1.5% interest rate to a closed account customer, then it may not be able to persuade this customer to re-open his account and not move to another bank. To solve this problem, we should list the values of common stable attributes in that rule. The corresponding (r_1, r_2) E- action rule will be:

$$[(Nationality, Asian) \wedge (AccountType, savings) \wedge \\ (InterestRate, 1\% \longrightarrow 0.5\%) \Longrightarrow ([Status, closedAccount \longrightarrow openAccount)].$$

It should be read: If an *Asian* has a savings account and its current interest rate of 1% is increased to 1.5%, then he may move from a group of closed account customers to a group that keeps their accounts open. This representation makes sense as action strategies are collected from a data set and represent the changeable behaviors under different conditions. Such an action is stimulated by an extended action rule and it is seen as a precise suggestion for actionability of rules. This E-action rule may say that if the bank offers a

1.5% interest rate to an *Asian* group of closed account customers, the offer may entice these customers to re-open their accounts. Sending that offer by regular mail or giving a call to all these customers are examples of an action associated with that E-action rule.

3 Information System and Action Rules

An information system is used for representing knowledge. Its definition, presented here, is due to Pawlak [7].

By an information system we mean a pair $S = (U, A)$, where:

- U is a nonempty, finite set of objects
- A is a nonempty, finite set of attributes i.e. $a : U \longrightarrow V_a$ is a function for any $a \in A$, where V_a is called the domain of a

Elements of U are called objects. In this chapter, for the purpose of clarity, objects are interpreted as customers. Attributes are interpreted as features such as, offers made by a bank, characteristic conditions etc.

We consider a special case of information systems called decision tables [7–9]. In any decision table together with the set of attributes a partition of that set into conditions and decisions is given. Additionally, we assume that the set of conditions is partitioned into stable conditions and flexible conditions. For simplicity reason, we assume that there is only one decision attribute. *Date of birth* is an example of a stable attribute. The *interest rate* on any customer account is an example of a flexible attribute as the bank can adjust rates. We adopt the following definition of a decision table:

By a decision table we mean any information system $S = (U, A_{St} \cup A_{Fl} \cup \{d\})$, where $d \notin A_{St} \cup A_{Fl}$ is a distinguished attribute called the decision. The elements of A_{St} are called stable conditions, whereas the elements of A_{Fl} are called flexible conditions.

As an example of a decision table we take S=$(\{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8\}$, $\{a, c\} \cup \{b\} \cup \{d\})$ represented by Table 1. The set $\{a, c\}$ lists stable attributes,

Table 1. Decision system

	a	b	c	d
x_1	0	S	0	L
x_2	0	R	1	L
x_3	0	S	0	L
x_4	0	R	1	L
x_5	2	P	2	L
x_6	2	P	2	L
x_7	2	S	2	H
x_8	2	S	2	H

b is a flexible attribute and d is a decision attribute. Also, we assume that H denotes a *high* profit and L denotes a *low* one.

In order to induce rules in which the THEN part consists of the decision attribute d and the IF part consists of attributes belonging to $A_{St} \cup A_{Fl}$, for instance *LERS* [5] can be used for rules extraction.

In order to efficiently extract rules when the number of attributes is large, we can use sub-tables $(U, B \cup \{d\})$ of S where B is a d-reduct (see [7]) in S. The set B is called d-reduct in S if there is no proper subset C of B such that d depends on C. The concept of d-reduct in S was introduced with a purpose to induce rules from S describing values of the attribute d depending on minimal subsets of $A_{St} \bigcup A_{Fl}$.

By $L(r)$ we mean all attributes listed in the IF part of a rule r. For example, if $r_1 = [(a_1, 2) \wedge (a_2, 1) \wedge (a_3, 4) \longrightarrow (d, 8)]$ is a rule then $L(r_1) = \{a_1, a_2, a_3\}$.

By $d(r_1)$ we denote the decision value of that rule. In our example $d(r_1) = 8$. If r_1, r_2 are rules and $B \subseteq A_{St} \cup A_{Fl}$ is a set of attributes, then $r_1/B = r_2/B$ means that the conditional parts of rules r_1, r_2 restricted to attributes B are the same. For example if $r_2 = [(a_2, 1) * (a_3, 4) \longrightarrow (d, 1)]$, then $r_1/\{a_2, a_3\} = r_2/\{a_2, a_3\}$.

In our example, we get the following optimal rules:

1. $(a, 0) \longrightarrow (d, L)$, $(c, 0) \longrightarrow (d, L)$
2. $(b, R) \longrightarrow (d, L)$, $(c, 1) \longrightarrow (d, L)$
3. $(b, P) \longrightarrow (d, L)$, $(a, 2) * (b, S) \longrightarrow (d, H)$
4. $(b, S) * (c, 2) \longrightarrow (d, H)$

Now, let us assume that $(a, v \longrightarrow w)$ denotes the fact that the value of attribute a has been changed from v to w. Similarly, the term $(a, v \longrightarrow w)(x)$ means that $a(x) = v$ has been changed to $a(x) = w$. Saying another words, the property (a, v) of object x has been changed to property (a, w).

Let $S = (U, A_{St} \cup A_{Fl} \cup \{d\})$ is a decision table and rules r_1, r_2 have been extracted from S. The notion of action rule was introduced in [10]. Its definition is given below. We assume here that:

- B_{St} is a maximal subset of A_{St} such that $r_1/B_{St} = r_2/B_{St}$
- $d(r_1) = k_1$, $d(r_2) = k_2$ and $k_1 \leq k_2$
- $(\forall a \in [A_{St} \cap L(r_1) \cap L(r_2)])[a(r_1) = a(r_2)]$
- $(\forall i \in p)(\forall b_i \in [A_{Fl} \cap L(r_1) \cap L(r_2)])[[b_i(r_1) = v_i] \& [b_i(r_2) = w_i]]$

By (r_1, r_2)-action rule on $x \in U$ we mean the expression r:

$[(b_1, v_1 \longrightarrow w_1) \wedge (b_2, v_2 \longrightarrow w_2) \wedge ... \wedge (b_p, v_p \longrightarrow w_p)](x)$
$\Longrightarrow [(d, k_1 \longrightarrow k_2)](x)$.

where $(b_j, v_j \rightarrow w_j)$ means that the value of the $j^t h$ flexible attribute b has been changed from v_j to w_j.

The notion of an extended action rule was given in [11]. The following two conditions have been added to the original definition of the action rule:

- $(\forall i \leq q)(\forall e_i \in [A_{St} \cap [L(r_2) - L(r_1)]])[e_i(r_2) = u_i]$
- $(\forall i \leq r)(\forall c_i \in [A_{Fl} \cap [L(r_2) - L(r_1)]])[c_i(r_2) = t_i]$

By an extended (r_1, r_2)-action rule on $x \in U$ we mean the expression:

$[(e_1 = u_1) \wedge (e_2 = u_2) \wedge \ldots \wedge (e_q = u_q) \wedge (b_1, v_1 \longrightarrow w_1) \wedge (b_2, v_2 \longrightarrow w_2) \wedge \ldots \wedge$
$(b_p, v_p \longrightarrow w_p) \wedge (c_1, \longrightarrow t_1) \wedge (c_2, \longrightarrow t_2) \wedge \ldots \wedge (c_r, \longrightarrow t_r)](x)$
$\Longrightarrow [(d, k_1 \longrightarrow k_2)](x)$.

where $(e_i = u_i)$ denotes the value of the i^th stable attribute which is equal to u_i. Additionally, $(c_l, \rightarrow t_l)$ indicates that the value of the l^th flexible attribute has to be changed from an arbitrary value to t_l.

The values of stable attributes which are listed in the second rule are also required to be listed in the extended action rule. This property narrows down the number of objects which have the highest chance to support the action rule. In addition, for any flexible attribute listed in the second rule and not listed in the first rule, its value should be changed to the value listed in the second rule. So, if attributes are correlated, the change of one attribute value will influence the change of another value. If the classification attributes are not correlated then automatically this requirement is not needed at all.

Let $A_{St} \cap L(r_1) \cap L(r_2) = B$. By (r_1, r_2) -E-action rule on $x \in U$ we mean the expression:

$[\prod\{a = a(r_1) : a \in B\}(e_1 = u_1) \wedge (e_2 = u_2) \wedge \ldots \wedge (e_q = u_q) \wedge (b_1, v_1 \longrightarrow w_1) \wedge (b_2, v_2 \longrightarrow w_2) \wedge \ldots \wedge (b_p, v_p \longrightarrow w_p) \wedge (c_1, \longrightarrow t_1) \wedge (c_2, \longrightarrow t_2) \wedge \ldots \wedge (c_r, \longrightarrow t_r)](x) \Longrightarrow [(d, k_1 \longrightarrow k_2)](x)$

where $a = a(r_1)$ denotes that the values of the common stable attributes for both rules are the same.

Common stable attributes used in rules from which an extended action rule is created, are not listed. So, extended action rules do not contain enough information about the condition requirements for domain objects in order to select the specific target objects and to yield highly accurate results. In other words, there are many possibilities, $\prod\{2^{V_a} : a \in B\}$, in applying a single action rule.

The idea behind the definition of E-action rule was to eliminate possibly all these objects in S which have a high chance to fail in supporting action rules. Therefore, the values of the common stable attributes listed in both rules from which E-action rule is created are also required to be listed in it. This requirement narrows down the number of objects supporting the rule to the objects which have the highest chance to succeed. So, E-action rules are more understandable and meaningful to the users than extended action rules.

Object $x \in U$ supports (r_1, r_2)-action rule r in $S = (U, A_{St} \cup A_{Fl} \cup \{d\})$, if the following conditions are satisfied:

- $(\forall i \leq p)[b_i \in L(r)][b_i(x) = v_i] \wedge d(x) = k_1$
- $(\forall i \leq p)[b_i \in L(r)][b_i(y) = w_i] \wedge d(y) = k_2$

- $(\forall j \leq p)[a_j \in (A_{St} \cap L(r_2))][a_j(x) = u_j]$
- $(\forall j \leq p)[a_j \in (A_{St} \cap L(r_2))][a_j(y) = u_j]$
- Object x supports rule r_1
- Object y supports rule r_2

By the support of an extended action rule r in S, denoted by $Sup_S(r)$, we mean the set of all objects in S supporting R. In other words, the set of all objects in S supporting r has the property

$(a_1 = u_1) \wedge (a_2 = u_2) \wedge \ldots \wedge (a_q = u_q) \wedge (b_1 = v_1) \wedge (b_2 = v_2) \wedge \ldots \wedge (b_p = v_p) \wedge (d = k_1)$.

By the confidence of R in S, denoted by $Conf_S(r)$, we mean

$[Sup_S(r)/Sup_S(L(r))][Conf(r_2)]$.

In order to find the confidence of (r_1, r_2)-E-action rule in S, we divide the number of objects supporting (r_1, r_2)-action rule in S by the number of objects supporting left hand side of (r_1, r_2)-E-action rule times the confidence of the second classification rule r_2 in S.

4 Discovering E-Action Rules

In this section we present a new algorithm for discovering E-action rules. Initially, we partition the set of rules discovered from an information system $S = (U, A_{St} \cup A_{Fl} \cup \{d\})$, where A_{St} is the set of stable attributes, A_{Fl} is the set of flexible attributes and, $V_d = \{d_1, d_2, \ldots, d_k\}$ is the set of decision values, into subsets of rules defining the same decision value. Saying another words, the set of rules R discovered from S is partitioned into $\{R_i\}_{i:1 \leq i \leq k}$, where $R_i = \{r \in R : d(r) = d_i\}$ for any $i = 1, 2, \ldots, k$. Clearly, the objects supporting any rule from R_i form subsets of $d^{-1}(\{d_i\})$.

Let us take Table 1 as an example of a decision system S. We assume that a, c are stable attributes and b, d are flexible. The set R of certain rules extracted from S is given below:

1. $(a, 0) \longrightarrow (d, L)$, $(c, 0) \longrightarrow (d, L)$
2. $(b, R) \longrightarrow (d, L)$, $(c, 1) \longrightarrow (d, L)$
3. $(b, P) \longrightarrow (d, L)$, $(a, 2) * (b, S) \longrightarrow (d, H)$
4. $(b, S) * (c, 2) \longrightarrow (d, H)$

We partition this set into two subsets $R_1 = \{[(a, 0) \longrightarrow (d, L)], [(c, 0) \longrightarrow (d, L)], [(b, R) \longrightarrow (d, L)], [(c, 1) \longrightarrow (d, L)], [(b, P) \longrightarrow (d, L)]\}$ and $R_2 = \{[(a, 2) * (b, S) \longrightarrow (d, H)], [(b, S) * (c, 2) \longrightarrow (d, H)]\}$.

Assume now that our goal is to re-classify some objects from the class $d^{-1}(\{d_i\})$ into the class $d^{-1}(\{d_j\})$. In our example, we assume that $d_i = (d, L)$ and $d_j = (d, H)$.

First, we represent the set R as a table (see Table 2). The first column of this table shows objects in S supporting the rules from R (each row represents

Table 2. Set of rules R with supporting objects

	a	b	c	d
$\{x_1, x_2, x_3, x_4\}$	0			L
$\{x_2, x_4\}$		R		L
$\{x_1, x_3\}$			0	L
$\{x_2, x_4\}$			1	L
$\{x_5, x_6\}$		P		L
$\{x_7, x_8\}$	2	S		H
$\{x_7, x_8\}$		S	2	H

a rule). The first five rows represent the set R_1 and the last two rows represent the set R_2. In the general case, assumed earlier, the number of different decision classes is equal to k.

The next step of the algorithm is to build d_i-tree and d_j-tree. First, from the initial table similar to Table 2, we select all rules (rows) defining the decision value d_i. Similarly, from the same table, we also select all rules (rows) which define decision value d_j.

By d_i-tree we mean a tree $T(d_i) = (N_i, E_i)$, such that:

- Each interior node is labelled by a stable attribute from A_1
- Each edge is labelled either by a question mark or by an attribute value of the attribute that labels the initial node of the edge
- Along a path, all nodes (except a leaf) are labelled with different stable attributes
- All edges leaving a node are labelled with different attribute values (including the question mark) of the stable attribute that labels that node
- Each leaf represents a set of rules which do not contradict on stable attributes and also define decision value d_i. The path from the root to that leaf gives the description of objects supported by these rules

Now, taking (d, L) from our example as the value d_i, we show how to construct (d, L)-tree for the set of rules represented by Table 2. The construction of (d, L)-tree starts with a table corresponding to the root of that tree (Table 3 in Fig. 1). It represents the set of rules R_1 defining L with supporting objects from S. We use stable attribute c to split that table into three sub-tables defined by values $\{0, 1, ?\}$ of attribute c. The question mark means an unknown value.

Following the path labelled by value $c = 1$, we get table $T2$. Following the path labelled by value $c = 0$, we get table $T3$. When we follow the path labelled by value $[c =?][a = 0]$, we get table $T4$. Finally, by following the path having the label $[c =?][a =?]$, we get table $T5$.

Now, let us define (d, H)-tree using Table 4 as its root (see Fig. 2). Following the path labelled by value $[c =?]$, we get the table $T6$. When we follow the path labelled by value $[c = 2]$, we get the table $T7$. Both tables can be easily constructed.

Table 3. Time needed to extract rules and action rules by *DEAR*

DataSet	*Rules*	*Action rules DEAR*
Breast Cancer	20 *s*	27 *min* 51 *s*
Cleveland	1 *min* 09 *s*	*Over* 8 *h*
Hepatitis	54 *s*	*Over* 8 *h*

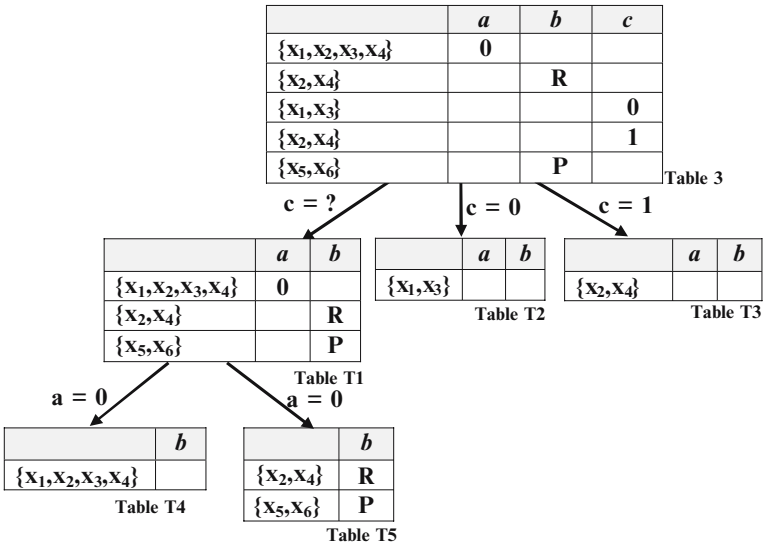


Fig. 1. (d, L)-tree

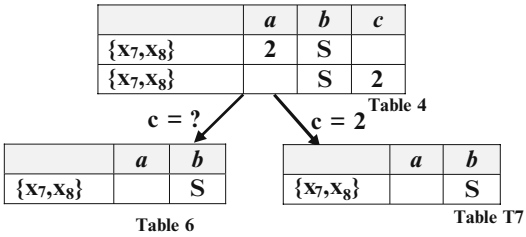


Fig. 2. (d, H)-tree

Now, it can be checked that only pairs of rules belonging to tables $\{[T5, T7],\ [T5, T6], [T2, T6],\ [T3, T6], [T4, T7]\}$ can be used in action rules construction. For each pair of tables, we use the same algorithm as in [11] to construct extended action rules.

This new algorithm (called *DEAR-2*) was implemented and tested on many datasets using PC with 1.8 GHz CPU. The time complexity of this algorithm was significantly lower than the time complexity of the algorithm *DEAR* presented in [11]. Both algorithms extract rules describing values of the decision

Table 4. Time needed to extract action rules by *DEAR-2*

DataSet	*Action Rules DEAR 2*
Breast Cancer	3 *s*
Cleveland	54 *min* 20 *s*
Hepatitis	51 *min* 53 *s*

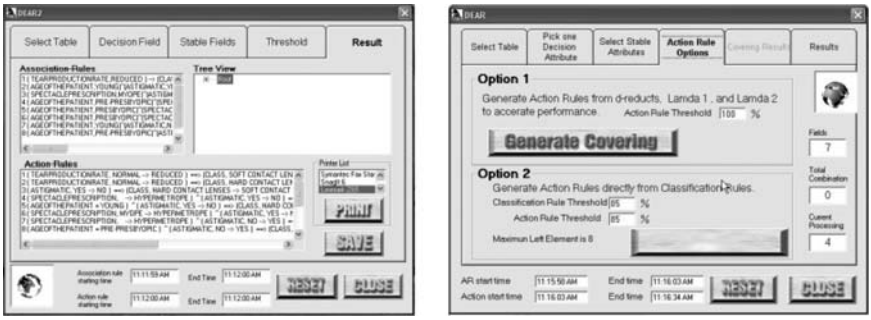


Fig. 3. *DEAR* and *DEAR2* interface

attribute before any action rule is constructed. The next two tables show the time needed by *DEAR* and *DEAR-2* to extract rules and next action rules from three datasets: *Breast Cancer*, *Cleveland*, and *Hepatitis*. These three *UCI* datasets are available at [*http://www.sgi.com/tech/mlc/db/*]. The first one has 191 records described by ten attributes. Only *Age* is the stable attribute. The second one has 303 records described by 15 attributes. Only two attributes *age* and *sex* are stable. The last one has 155 records described by 19 attributes. Again, only two attributes *age* and *sex* are stable.

The interface to both systems, *DEAR* and *DEAR-2*, is written in Visual Basic. The first picture in Fig. 3 shows part of the interface to both systems. The user has an option to generate the coverings (see [7, 8]) for the decision attribute and next use them in the process of action rules extraction or, if he prefers, he can directly proceed to the rules extraction step. It is recommended, by *DEAR-2*, to generate the coverings for the decision attribute if the information system has many classification attributes. By doing this we usually speed up the process of action rules extraction. The second picture in Fig. 3 shows how the results are displayed by *DEAR-2* system.

5 Conclusion

Generally speaking, actionable knowledge discovery based on action rule mining can provide a coarse framework for users in applying the rules to objects. We see that there is a clear need for an effective representation of actionable knowledge by giving the users exactly the information they need. Hence, we

propose E-action rules that enhance the extended action rules by adding to its descriptions the values of common stable attributes listed in both classification rules, used to construct an action rule, in order to provide a more sound and well-defined strategy. Any E-action rule provides a well defined hint to a user of what changes within flexible attributes are needed to re-classify some objects from a lower preferable class to a higher one. Hence, E-action rules mining is a technique that intelligently and automatically forms precise actions that can be adopted by decision-making users in achieving their goals.

System *DEAR-2* initially generates a set of classification rules from S (satisfying two thresholds, the first one for a minimum support and second for a minimum confidence) defining values of a chosen attribute, called decision attribute in S, in terms of the remaining attributes. *DEAR-2* is giving preference to rules which classification part contains maximally small number of stable attributes in S. These rules are partitioned by *DEAR-2* into a number of equivalence classes where each equivalence class contains only rules which classification part has the same values of stable attributes. Each equivalence class is used independently by *DEAR-2* as a base for constructing action rules. The current strategy requires the generation of classification rules from S to form a base, before the process of action rules construction starts. We believe that by following the process similar to *LERS* (see [2, 5]) or *ERID* (see [3]) which is initially centered on all stable attributes in S, we should be able to construct action rules directly from S and without the necessity to generate the base of classification rules.

References

1. Adomavicius G, Tuzhilin A (1997) Discovery of actionable patterns in databases: the action hierarchy approach. In: Proceedings of KDD97 Conference. AAAI, Newport Beach, CA
2. Chmielewski M R, Grzymala-Busse J W, Peterson N W, Than S (1993) The rule induction system LERS – a version for personal computers. In: Foundations of Computing and Decision Sciences. Vol. 18, No. 3–4, Institute of Computing Science, Technical University of Poznan, Poland, 181–212
3. Dardzińska A, Raś Z W (2003) On rule discovery from incomplete information systems. In: Proceedings of ICDM'03 Workshop on Foundations and New Directions of Data Mining, (Eds: Lin T Y, Hu X, Ohsuga S, Liau C). IEEE Computer Society, Melbourne, Florida, 31–35
4. Geffner H, Wainer J (1998) Modeling action, knowledge and control. In: ECAI 98, Proceedings of the 13th European Conference on AI, (Ed: Prade H). Wiley, New York, 532–536
5. Grzymala-Busse J (1997) A new version of the rule induction system LERS. In: Fundamenta Informaticae, Vol. 31, No. 1, 27–39
6. Liu B, Hsu W, Chen S (1997) Using general impressions to analyze discovered classification rules. In: Proceedings of KDD97 Conference. AAAI, Newport Beach, CA

7. Pawlak Z (1991) Rough sets-theoretical aspects of reasoning about data. Kluwer, Dordrecht
8. Pawlak Z (1981) Information systems – theoretical foundations. In: Information Systems Journal, Vol. 6, 205–218
9. Polkowski L, Skowron A (1998) Rough sets in knowledge discovery. In: Studies in Fuzziness and Soft Computing, Physica/Springer, Berlin Heidelberg New York
10. Raś Z, Wieczorkowska A (2000) Action rules: how to increase profit of a company. In: Principles of Data Mining and Knowledge Discovery, Proceedings of PKDD'00 (Eds: Zighed DA, Komorowski J, Zytkow J). Lyon, France, LNCS/LNAI, No. 1910, Springer, Berlin Heidelberg New York, 587–592
11. Raś Z W, Tsay L-S (2003) Discovering extended action-rules (System DEAR). In: Intelligent Information Systems 2003, Proceedings of the IIS'2003 Symposium, Zakopane, Poland, Advances in Soft Computing, Springer, Berlin Heidelberg New York, 293–300
12. Raś Z, Gupta S (2002) Global action rules in distributed knowledge systems. In: Fundamenta Informaticae Journal, IOS Press, Vol. 51, No. 1–2, 175–184
13. Silberschatz A, Tuzhilin A (1995) On subjective measures of interestingness in knowledge discovery. In: Proceedings of KDD́95 Conference, AAAI, Newport Beach, CA
14. Silberschatz A, Tuzhilin A (1996) What makes patterns interesting in knowledge discovery systems. In: IEEE Transactions on Knowledge and Data Engineering, Vol. 8, No. 6, 970–974

Mining E-Action Rules, System DEAR

Zbigniew W. Raś[1,2] and Li-Shiang Tsay[3]

[1] University of North Carolina, Computer Science Department, Charlotte, NC 28223, USA
ltsay@uncc.edu
[2] Polish–Japanese Institute of Information Technology, ul. Koszykowa 86, 02-008, Warsaw, Poland
ras@uncc.edu
[3] North Carolina A&T State University, School of Technology, Greensboro, NC 27411, USA
ltsay@ncat.edu

Summary. The essential problem of Knowledge Discovery in Databases is to find interesting relationships, those that are meaningful in a domain. This task may be viewed as one of searching an immense space of possible actionable concepts and relations. Because the classical knowledge discovery algorithms are not able to determine if a pattern is truly actionable for a user, we focus on a new class of action rules, called e-action rules that can be used not only for automatically analyzing discovered patterns but also for reclassifying some objects in the data from one state into another more desired state. For a quicker and more effective process of e-action rules discovery, action tree algorithm is presented. Support and confidence of the rules are proposed to prune a large number of irrelevant, spurious, and insignificant generated candidates. The algorithm is implemented as $DEAR\_2.2$ system and it is tested on several public domain databases. The results show that actionability can be considered as a partially objective measure rather than a purely subjective one. E-Action rules are useful in many fields such as medical diagnosis and business.

1 Introduction

Finding useful rules is an important task of knowledge discovery in data. Most of the researchers on knowledge discovery focus on techniques for generating patterns, such as classification rules, association rules ... etc, from a data set. They assume that it is user's responsibility to analyze the patterns in order to infer solutions for specific problems within a given domain. The classical knowledge discovery algorithms have the potential to identify enormous number of significant patterns from data. Therefore, people are overwhelmed by a large number of uninteresting patterns, it is very difficult for a human being to analyze them in order to form timely solutions. In other words, not all

Z.W. Raś and L.-S. Tsay: *Mining E-Action Rules, System DEAR*, Studies in Computational Intelligence (SCI) **118**, 289–298 (2008)
www.springerlink.com

discovered rules are interesting enough to be presented and used. Therefore, a significant need exists for a new generation of techniques and tools with the ability to assist people in analyzing a large number of rules for useful knowledge. E-action rules mining is the technique that intelligently and automatically assists humans acquiring useful information. This information can be turned into action and this action can ultimately achieve a competitive advantage and benefit users.

E-action rule [13] structure is a significant improvement of a classical action rule [9] and an extended action rule [10] by providing well-defined definition. E-action rules automatically assists humans acquiring actionable knowledge. They are constructed from certain pairs of classification rules. These rules can be used not only for evaluating discovered patterns but also for reclassifying some objects in the data from one state into another more desired state. Given a set of classification rules found from past and current data, they can interpret regularities about past events and can be used for predicting the class label of data objects. For example, classification rules found from a bank's data are very useful to describe who is a good client (whom to offer some additional services) and who is a bad client (whom to watch carefully to minimize the bank loses). However, if bank managers hope to improve their understanding of customers and seek specific actions to improve services, mere classification rules will not be convincing for them. Therefore, we can use the classification rules to build a strategy of action based on condition features in order to get a desired effect on a decision feature. Going back to the bank example, the strategy of action would consist of modifying some condition features in order to improve their understanding of customers and then improve services. E-action rules can be useful in many other fields, like medical diagnosis. In medical diagnosis, classification rules can explain the relationships between symptoms and sickness and the same help doctors to set up the proper treatment for a new patient.

There are two types of interestingness measure: objective and subjective (see [1,5,11,12]). Subjective interestingness measures include unexpectedness [11] and actionability [1,3]. When the rule contradicts the user's prior belief about the domain, uncovers new knowledge or surprises them, it is classified as unexpected. A rule is deemed actionable, if the user can take an action to gain the advantage based on that rule. Domain experts basically look at a rule and say that this rule can be converted into an appropriate action. In this chapter, our main interest is to tackle actionability issue by analyzing classification rules and effectively develop workable strategies.

Action Tree algorithm is presented for generating e-action rules and implemented as System $DEAR\_2.2$. The algorithm adopts a top-down strategy that searches for a solution in a part of the search space. Seeking at each stage for a stable attribute that has the least amount of values; the set of rules is split using that attribute and then the subsets that result from the split are recursively processed. When all stable attributes are processed, the subsets are split further based on a decision attribute. This strategy generates

an action tree which can be used to construct extended action rules from the leaf nodes of the same parent.

2 Goal of E-Action Rules

The aim of an e-action rule is to look at the actionability in an objective way because e-action rules evaluate discovered classification rules based on statistics and structures of patterns. E-action rule is data driven and domain independent because it is based on the strategy which does not depend on domain knowledge. We claim that actionability does not have to be seen as a purely subjective concept. The definition of e-action rules is objective. However, we can not omit some degree of subjectivity in determining the attribute class and what action to take. In order to do that, we divide all attributes into two parts, stable and flexible. Obviously, this classification has to be done by users to decide which attributes are stable and which are flexible. This is a purely subjective decision. A stable attribute has no influence on reclassification, but a flexible attribute does influence changes. Users have to be careful judging which attributes are stable and which are flexible. If we apply e-action rules on objects then their flexible attributes can be changed but the stable attributes remain the same. Basically, an e-action rule shows that some selected objects can be reclassified from an undesired decision state to a desired one by changing some of the values of their flexible features. How to take an action on those flexible attributes can be determined by following either objective or subjective approach. It depends on the characteristic of the corresponding flexible attributes. If the attribute is an interest rate on the bank account then the bank can take an action as the rule states (i.e., lower the interest rate to 4.75%). So, in this case, it is a purely objective decision. However, if the attribute is a fever then doctors may chose several alternative treatments for decreasing patient′s temperature. The choice of a treatment is a subjective decision. Basically, we cannot eliminate some amount of subjectivity in that process.

3 Information System and E-Action Rules

An information system is used for representing knowledge. Its definition, presented here, is due to Pawlak [7].

By an information system we mean a pair $S = (U, A)$, where:

- U is a nonempty, finite set of objects
- A is a nonempty, finite set of attributes i.e. $a : U \longrightarrow V_a$ is a function for any $a \in A$, where V_a is called the domain of a

Elements of U are called objects. In this chapter, for the purpose of clarity, objects are interpreted as customers. Attributes are interpreted as features such as, offers made by a bank, characteristic conditions, etc.

Table 1. Decision system

	a	b	c	d
x_1	2	1	2	L
x_2	2	1	2	L
x_3	1	1	0	H
x_4	1	1	0	H
x_5	2	3	2	H
x_6	2	3	2	H
x_7	2	1	1	L
x_8	2	1	1	L
x_9	2	2	1	L
x_{10}	2	3	0	L
x_{11}	1	1	2	H
x_{12}	1	1	1	H

We consider a special case of information systems called decision tables [7, 8]. In any decision table together with the set of attributes a partition of that set into conditions and decisions is given. Additionally, we assume that the set of conditions is partitioned into stable conditions and flexible conditions. For simplicity reason, we assume that there is only one decision attribute. *Date of birth* is an example of a stable attribute. The *interest rate* on any customer account is an example of a flexible attribute as the bank can adjust rates. We adopt the following definition of a decision table:

A decision table is an information system of the form $S = (U, A_{St} \cup A_{Fl} \cup \{d\})$, where $d \notin A_{St} \cup A_{Fl}$ is a distinguished attribute called the decision. The elements of A_{St} are called stable conditions, whereas the elements of A_{Fl} are called flexible conditions.

As an example of a decision table we take $S = (\{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}, x_{12}\}, \{a, c\} \cup \{b\} \cup \{d\})$ represented by Table 1. The set $\{a, c\}$ lists stable attributes, b is a flexible attribute and d is a decision attribute. Also, we assume that H denotes a *high* profit and L denotes a *low* one.

In order to induce rules in which the THEN part consists of the decision attribute d and the IF part consists of attributes belonging to $A_{St} \cup A_{Fl}$, *LERS* [2, 4] can be used for rules extraction.

In order to efficiently extract such rules, when the number of attributes is large, we can use sub-tables $(U, B \cup \{d\})$ of S where B is a d-reduct (see [6]) in S. The set B is called d-reduct in S if there is no proper subset C of B such that d depends on C. The concept of d-reduct in S was introduced to induce rules from S describing values of the attribute d depending on minimal subsets of $A_{St} \cup A_{Fl}$.

By $L(r)$ we mean all attributes listed in the IF part of a rule r. For example, if $r_1 = [(a_1, 2) \wedge (a_2, 1) \wedge (a_3, 4) \longrightarrow (d, 8)]$ is a rule then $L(r_1) = \{a_1, a_2, a_3\}$.

By $d(r_1)$ we denote the decision value of that rule. In our example $d(r_1) = 8$. If r_1, r_2 are rules and $B \subseteq A_{St} \cup A_{Fl}$ is a set of attributes, then $r_1/B = r_2/B$ means that the conditional parts of rules r_1, r_2 restricted to

attributes B are the same. For example if $r_2 = [(a_2,1) * (a_3,4) \longrightarrow (d,1)]$, then $r_1/\{a_2,a_3\} = r_2/\{a_2,a_3\}$.

In our example, we get the following optimal rules which support is greater or equal to 2:

$(b,3) * (c,2) \longrightarrow (d,H)$, $(a,1) * (b,1) \longrightarrow (d,L)$,
$(a,1) * (c,1) \longrightarrow (d,L)$, $(b,1) * (c,0) \longrightarrow (d,H)$,
$(a,1) \longrightarrow (d,H)$

Now, let us assume that $(a, v \longrightarrow w)$ denotes the fact that the value of attribute a has been changed from v to w. Similarly, the term $(a, v \longrightarrow w)(x)$ means that $a(x) = v$ has been changed to $a(x) = w$. Saying another words, the property (a, v) of object x has been changed to property (a, w).

Let $S = (U, A_{St} \cup A_{Fl} \cup \{d\})$ is a decision table and rules r_1, r_2 have been extracted from S. The notion of e-action rule was given in [13]. Its definition is given below. We assume here that:

- B_{St} is a maximal subset of A_{St} such that $r_1/B_{St} = r_2/B_{St}$
- $d(r_1) = k_1$, $d(r_2) = k_2$ and $k_1 \leq k_2$
- $(\forall a \in [A_{St} \cap L(r_1) \cap L(r_2)])[a(r_1) = a(r_2)]$
- $(\forall i \leq q)(\forall e_i \in [A_{St} \cap [L(r_2) - L(r_1)]])[e_i(r_2) = u_i]$
- $(\forall i \leq r)(\forall c_i \in [A_{Fl} \cap [L(r_2) - L(r_1)]])[c_i(r_2) = t_i]$
- $(\forall i \leq p)(\forall b_i \in [A_{Fl} \cap L(r_1) \cap L(r_2)])[[b_i(r_1) = v_i] \& [b_i(r_2) = w_i]]$

Let $A_{St} \cap L(r_1) \cap L(r_2) = B$. By (r_1, r_2) -e-action rule on $x \in U$ we mean the expression r:

$[\prod\{a = a(r_1) : a \in B\}(e_1 = u_1) \wedge (e_2 = u_2) \wedge ... \wedge (e_q = u_q) \wedge (b_1, v_1 \longrightarrow w_1) \wedge (b_2, v_2 \longrightarrow w_2) \wedge ... \wedge (b_p, v_p \longrightarrow w_p) \wedge (c_1, \longrightarrow t_1) \wedge (c_2, \longrightarrow t_2) \wedge ... \wedge (c_r, \longrightarrow t_r)](x) \Longrightarrow [(d, k_1 \longrightarrow k_2)](x)$

Object $x \in U$ supports (r_1, r_2)-e-action rule r in $S = (U, A_{St} \cup A_{Fl} \cup \{d\})$, if the following conditions are satisfied:

- $(\forall i \leq p)[\forall b_i \in L(r)][b_i(x) = v_i] \wedge d(x) = k_1$
- $(\forall i \leq p)[\forall b_i \in L(r)][b_i(y) = w_i] \wedge d(y) = k_2$
- $(\forall j \leq p)[\forall a_j \in (A_{St} \cap L(r_2))][a_j(x) = u_j]$
- $(\forall j \leq p)[\forall a_j \in (A_{St} \cap L(r_2))][a_j(y) = u_j]$
- Object x supports rule r_1
- Object y supports rule r_2

By the support of e-action rule r in S, denoted by $Sup_S(r)$, we mean the set of all objects in S supporting R. In other words, the set of all objects in S supporting r has the property

$(a_1 = u_1) \wedge (a_2 = u_2) \wedge ... \wedge (a_q = u_q) \wedge (b_1 = v_1) \wedge (b_2 = v_2) \wedge ... \wedge (b_p = v_p) \wedge (d = k_1)$.

By the confidence of R in S, denoted by $Conf_S(r)$, we mean

$[Sup_S(r)/Sup_S(L(r))][Conf(r_2)]$

To find the confidence of (r_1, r_2)-e-action rule in S, we divide the number of objects supporting (r_1, r_2)-action rule in S by the number of objects supporting the left hand side of (r_1, r_2)-e-action rule times the confidence of the second classification rule r_2 in S.

4 Discovering E-Action Rules

In this section we present a new algorithm, Action-Tree algorithm, for discovering e-action rules. Basically, we partition the set of rules discovered from an information system $S = (U, A_{St} \cup A_{Fl} \cup \{d\})$, where A_{St} is the set of stable attributes, A_{Fl} is the set of flexible attributes and, $V_d = \{d_1, d_2, \ldots, d_k\}$ is the set of decision values, into subsets of rules supporting the same values of stable attributes and the same decision value.

Action-tree algorithm for extracting e-action rules from decision system S is as follows:

i. Build Action-Tree
   a. Divide the rule table, R, taking into consideration all stable attributes
      1. Find the domain $Dom(v_i^{St})$ of each stable attribute v_i^{St} from the initial table.
      2. Assuming that the number of values in $Dom(v_i^{St})$ is the smallest, partition the current table into sub-tables each of which contains only rules supporting values of stable attributes in the corresponding sub-table.
      3. Determine if a new table contains minimum two different decision values and minimum two different values for each flexible attribute. If it does, go to Step 2, otherwise there is no need to split the table further and we place a mark.
   b. Divide each lowest level sub-table into new sub-tables each of which contains rules having the same decision value.
   c. Represent each leaf as a set of rules which do not contradict on stable attributes and also define decision value d_i. The path from the root to that leaf gives the description of objects supported by these rules.
ii. Generate e-action rules
   a. Form e-action rules by comparing all unmarked leaf nodes of the same parent.
   b. Calculate the support and the confidence of a new-formed rule. If its support and confidence meet the requirements, print it.

The algorithm starts with all extracted classification rules at the root node of the tree. A stable attribute is selected to partition theses rules. For each value of the attribute a branch is created, and the corresponding subset of rules that have the attribute value specified by the branch is moved to the newly created child node. Now the process is repeated recursively for each

child node. When we are done with stable attributes, the last split is based on a decision attribute for each branch. If at any time all instances at a node have the same decision value, then we stop developing that part of the tree. The only thing left to build the tree is to decide how to determine which of the stable attributes to split, given a set of rules with different classes. The node selection is based on the stable attributes with the smallest number of possible values among all the remaining stable attributes.

An e-action tree has two types of nodes: a leaf node and a nonleaf node. At a nonleaf node in the tree, the set of rules is partitioned along the branches and each child node gets its corresponding subset of rules. Every path to the decision attribute node, one level above the leaf node, in the action tree represents a subset of the extracted classification rules when the stable attributers have the same value. Each leaf represents a set of rules, which do not contradict on stable attributes and also define decision value d_i. The path from the root to that leaf gives the description of objects supported by these rules.

Let us take Table 1 as an example of a decision system S. We assume that a, c are stable attributes and b, d are flexible. Assume now that our goal is to re-classify some objects from the class $d^{-1}(\{d_i\})$ into the class $d^{-1}(\{d_j\})$. In our example, we assume that $d_i = (d, L)$ and $d_j = (d, H)$.

First, we represent the set R of certain rules extracted from S as a table (see Table 2). The first column of this table shows objects in S supporting the rules from R (each row represents a rule). The construction of an action tree starts with the set R as a table (see Table 2) at the root of the tree (T_1 in Fig. 1). The root node selection is based on a stable attribute with the smallest number of states among all stable attributes. The same strategy is used for the child node selection. After putting all stable attributes on the tree, the tree is split based on the value of the decision attribute. Referring back to the example in Table 1, we use stable attribute a to split that table into two sub-tables defined by values $\{1, 2\}$ of attribute a. The domain of attribute a is $\{1, 2\}$ and the domain of attribute c is $\{0, 1, 2\}$. Clearly, $card[V_a]$ is less than $card[V_c]$ so we divide the table into two: one table with rules containing $a = 0$ and another with rules containing $a = 2$. Each corresponding edge is labelled by the value of attribute a. Next, all objects in the sub-table T_2 have the same decision value. We can not generate any e-action rules from this sub-table so it is not divided any further. Because sub-table T_3 contains different decision

Table 2. Set of rules R with supporting objects

Objects	a	b	c	d
$\{x_3, x_4, x_{11}, x_{12}\}$	1			H
$\{x_1, x_2, x_7, x_8\}$	2		1	L
$\{x_7, x_8, x_9\}$	2		0	L
$\{x_3, x_4\}$		1	0	H
$\{x_5, x_6\}$		3	2	H

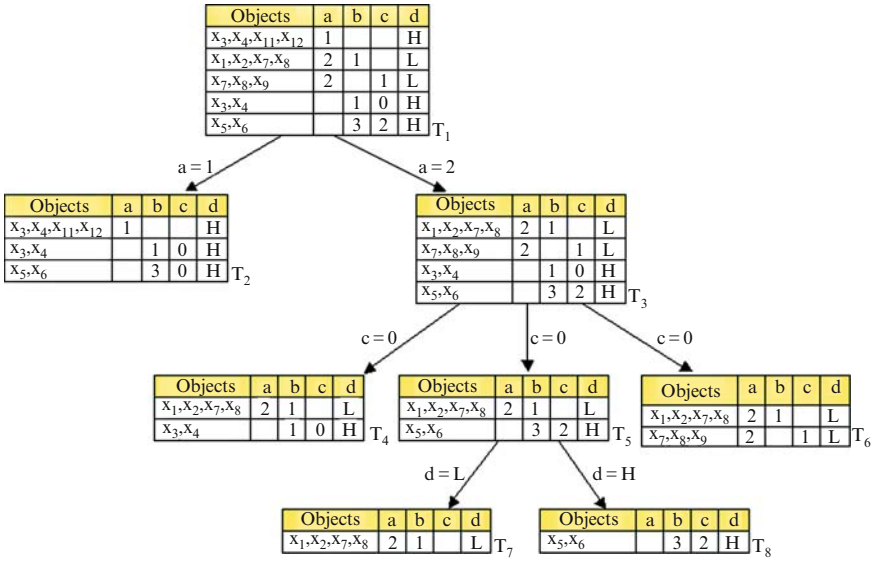


Fig. 1. Action tree

values and stable attribute, it is divided into three, one with rules containing c = 0, one with rules containing c = 1, and one with rules containing c = 2. At this step, each sub-table does not contain any stable attributes. Table T_6 can not be split any further for the same reason as sub-table T_2. All objects in sub-table T_4 have the same value of flexible attribute b, so the table is not partitioned any further. The remaining table T_5 is partitioned into two sub-tables. Each leaf represents a set of rules which do not contradict on stable attributes and also define decision value d_i.

The path from the root to that leaf gives the description of objects supported by these rules. Following the path labelled by value $[a = 2]$, $[c = 2]$, and $[d = L]$, we get table T_7. Following the path labelled by value $[a = 2]$, $[c = 2]$, and $[d = H]$, we get table T_8. Because T_7 and T_8 are sibling nodes, we can directly compare pairs of rules belonging to these two tables and construct one e-action rule such as:

$$[[(a, 2) \wedge (b, 1 \rightarrow 3)] \Rightarrow (d, L \rightarrow H)].$$

After the rule is formed, we evaluate it by checking its support and its confidence. We have discovered an extended action rule given below:

$$[[(a, 2) \wedge (b, 1 \rightarrow 3)] \Rightarrow (d, L \rightarrow H)]; \; sup : 4 conf : 100\%.$$

This new algorithm (called DEAR_2.2) was implemented and tested on many datasets using PC with 1.8 GHz CPU. The time complexity of the action forest algorithm used in system $DEAR\_2$ is $(k * n) + (2k * n * 1) + log(n) = O(n + log(n))$, where n is the number of classification rules, k is the number

of attributes in S. The time complexity of the action tree algorithm is $(k * n) + (2k * n * 1) = O(n)$, where n is the number of classification rules, k is the number of attributes in S. The action tree algorithm is simpler and more efficient then the action forest algorithm.

5 Conclusion

Discovering a set of rules is not the end of knowledge discovery process. A rule is interesting, if it meets some user specified thresholds. As knowledge discovery techniques are increasingly used to solve real life problems, a significant need exists for a new generation of technique, e-action rules mining, with the ability to facilitate human beings in evaluating and interpreting the discovered patterns. Additionally, as the value of data is related to how quickly and effectively the data can be reduced, explored, manipulated and managed, we propose a new strategy, action-tree algorithm for discovering e-action rules. Our results show that actionability can be considered as a partially objective measure rather than a purely subjective one.

References

1. Adomavicius G, Tuzhilin A (1997) Discovery of actionable patterns in databases: the action hierarchy approach. In: Proceedings of KDD97 Conference, AAAI, Newport Beach, CA
2. Chmielewski M R, Grzymala-Busse J W, Peterson N W, Than S (1993) The rule induction system LERS – a version for personal computers. In: Foundations of Computing and Decision Sciences. Vol. 18, No. 3–4, Institute of Computing Science, Technical University of Poznan, Poland: 181–212
3. Geffner H, Wainer J (1998) Modeling action, knowledge and control. In: ECAI 98, Proceedings of the 13th European Conference on AI, (Ed. H. Prade), Wiley, New York, 532–536
4. Grzymala-Busse J (1997) A new version of the rule induction system LERS. In: Fundamenta Informaticae, Vol. 31, No. 1, 27–39
5. Liu B, Hsu W, Chen S (1997) Using general impressions to analyze discovered classification rules. In: Proceedings of KDD97 Conference, AAAI, Newport Beach, CA
6. Pawlak Z (1991) Rough sets-theoretical aspects of reasoning about data. Kluwer, Dordrecht
7. Pawlak Z (1981) Information systems – theoretical foundations. In: Information Systems Journal, Vol. 6, 205–218
8. Polkowski L, Skowron A (1998) Rough sets in knowledge discovery. In: Studies in Fuzziness and Soft Computing, Physica-Verlag/Springer, Berlin Heidelberg New York
9. Raś Z, Wieczorkowska A (2000) Action rules: how to increase profit of a company. In: Principles of Data Mining and Knowledge Discovery, (Eds. D.A. Zighed, J. Komorowski, J. Zytkow), Proceedings of PKDD '00, Lyon, France, LNCS/LNAI, No. 1910, Springer, Berlin Heidelberg New York, 587–592

10. Raś Z W, Tsay L-S (2003) Discovering extended action-rules (System DEAR). In: Intelligent Information Systems 2003, Proceedings of the IIS'2003 Symposium, Zakopane, Poland, Advances in Soft Computing, Springer, Berlin Heidelberg New York, 293–300
11. Silberschatz A, Tuzhilin A (1995) On subjective measures of interestingness in knowledge discovery. In: Proceedings of KDD́95 Conference, AAAI, Newport Beach, CA
12. Silberschatz A, Tuzhilin A (1996) What makes patterns interesting in knowledge discovery systems. In: IEEE Transactions on Knowledge and Data Engineering, Vol. 5, No. 6
13. Tsay L-S, Raś Z W (2006) E-Action Rules. In: Post-Proceeding of FDM'04 Workshop Advances in soft Computing, Springer, Berlin Heidelberg New York

Definability of Association Rules and Tables of Critical Frequencies

Jan Rauch

Department of Information and Knowledge Engineering, University of Economics, Prague, nám. W. Churchilla 4, 130 67 Praha 3, Czech Republic
rauch@vse.cz

Summary. Former results concerning definability of association rules in classical predicate calculi are summarized. A new intuitive criteria of definability are presented. The presented criteria concern important classes of association rules. They are based on tables of critical frequencies of association rules. These tables were introduced as a tool for avoiding complex computation related to verification of rules corresponding to statistical hypotheses tests.

1 Introduction

Goal of this chapter is to contribute to theoretical foundations of data mining. We deal with association rules of the form $\varphi \approx \psi$ where φ and ψ are Boolean attributes derived from columns of the analysed data matrix $\mathcal{M}$. The association rule $\varphi \approx \psi$ says that φ and ψ are associated in a way given by the symbol $\approx$. The symbol $\approx$ is called *4ft-quantifier* . It corresponds to a condition concerning a fourfold contingency table of φ and ψ in $\mathcal{M}$. Association rules of this form were introduced and studied in [2]. They were further studied among others in [4, 8], the results were partly published in [5–7, 9, 10].

This chapter concerns definability of the association rules in classical predicate calculi. The association rules can be understood as formulae of monadic predicate observational calculi defined in [2]. The monadic predicate observational calculus is a modification of classical predicate calculus: only finite models are allowed and generalized quantifiers are added. 4ft-quantifier $\approx$ is an example of the generalized quantifier.

There is a natural question of classical definability of the association rules i.e. the question *which association rules can be expressed by means of classical predicate calculus* (predicates, variables, classical quantifiers $\forall$ and $\exists$, Boolean connectives and the predicate of equality). This question is solved by the Tharp's theorem proved in [2, 13].

J. Rauch: *Definability of Association Rules and Tables of Critical Frequencies*, Studies in Computational Intelligence (SCI) **118**, 299–313 (2008)
www.springerlink.com

The Tharp's theorem is but too general from the point of view of the association rules. A more intuitive criterion of classical definability of association rules was proved in [4] see also [10].

The goal of this chapter is to show that this criterion can be further simplified for several important classes of association rules. The simplified criterion is based on tables of critical frequencies (further we use only *TCF* instead of *table of critical frequencies*). TCF's were introduced as a tool for avoiding complex computation [2, 4] related to the association rules corresponding to statistical hypotheses tests.

The chapter uses notions of association rules and classes of association rules described in [11] in this book. Tables of critical frequencies are introduced in Sect. 2. Results concerning classical definability of the association rules are presented in Sects. 3 and 4. Some concluding remarks are in Sect. 5.

2 Tables of Critical Frequencies

First we prove the theorem about *partial tables of maximal b*. Further we will denote $\mathcal{N}^+ = \{0, 1, 2, \dots\} \cup \{\infty\}$.

Theorem 1. *Let $\approx$ be an equivalency quantifier. Then there is a non-negative function $Tb_{\approx}$ that assigns to each triple $\langle a, c, d\rangle$ of non-negative natural numbers a value $Tb_{\approx}(a, c, d) \in \mathcal{N}^+$ such that*

1. *For each $b \geq 0$ it is $\approx (a, b, c, d) = 1$ if and only if $b < Tb_{\approx}(a, c, d)$.*
2. *If $a' > a$ then $Tb_{\approx}(a', c, d) \geq Tb_{\approx}(a, c, d)$.*

Proof. Let us define

$$Tb_{\approx}(a, c, d) = \min\{b| \approx (a, b, c, d) = 0\}.$$

Let us remember that $\approx$ is equivalency. It means that

$$\approx (a, b, c, d) = 1 \ \wedge \ \ a' \geq a \ \wedge \ b' \ \leq b \ \wedge \ c' \leq c \ \wedge \ d' \geq d$$

implies

$$\approx (a', b', c', d') = 1$$

It means among other

I: If $\approx (a, b, c, d) = 0$ and $v \leq a$ then also $\approx \ (v, b, c, d) = \ 0$.
II: If $\approx (a, b, c, d) = 0$ and $w \geq b$ then also $\approx \ (a, w, c, d) = 0$.

Point II means that it is $\approx (a, b, c, d) = 0$ for all $b \geq \min\{b| \approx (a, b, c, d) = 0\}$. We prove that the function defined in the above given way has the properties 1. and 2.

1. *Let us suppose* $b \geq 0$ *and* $\approx (a,b,c,d) = 1$. *We have to prove that it is* $b < Tb_{\approx}(a,c,d)$. *Let us suppose* $b \geq Tb_{\approx}(a,c,d)$. *It however means according to point II that* $\approx (a,b,c,d) = 0$. *Thus it must be* $b < Tb_{\approx}(a,c,d)$.
   Let us suppose $b \geq 0$ *and* $\approx (a,b,c,d) = 0$. *We have to prove that it is* $b \geq Tb_{\approx}(a,c,d)$. *But it follows from the definition of* $Tb_{\approx}(a,c,d)$.
2. *Let us suppose* $a' > a$ *and also* $Tb_{\approx}(a',c,d) < Tb_{\approx}(a,c,d)$. *Let us denote* $e = Tb_{\approx}(a,c,d)$, *thus it is* $e > 0$. *It means* $Tb_{\approx}(a',c,d) \leq e-1$ *and thus according to the definition of* $Tb_{\approx}(a',c,d)$ *it is* $\approx (a', e-1, c, d) = 0$. *Due to point I it is also* $\approx (a, e-1, c, d) = 0$. *It is but also* $e-1 < e = Tb_{\approx(a,c,d)}$ *and it means* $\approx (a, e-1, c, d) = 1$ *according to already proved point 1. It is a contradiction thus it cannot be* $a' > a$ *and* $Tb_{\approx}(a',c,d) < Tb_{\approx}(a,c,d)$. *It but means that it follows* $Tb_{\approx}(a',c,d) \geq Tb_{\approx}(a,c,d)$ *from* $a' > a$.

   This finishes the proof. □

Remark 2.1 *It is easy to prove for the implicational quantifier* $\Rightarrow^*$ *that the value* $\Rightarrow^* (a,b,c,d)$ *depends neither on* c *nor on* d. *It means that we can write only* $\Rightarrow^* (a,b)$ *instead of* $\Rightarrow^* (a,b,c,d)$ *for the implicational quantifier* $\Rightarrow^*$, *see also Remark 3.1 in [11].*

The just proved theorem has a direct consequence for the implicational quantifiers.

Theorem 2. *Let* $\Rightarrow^*$ *be an implicational quantifier. Then there is a non-negative non-decreasing function* $Tb_{\Rightarrow^*}$ *that assigns a value* $Tb_{\Rightarrow^*}(a) \in \mathcal{N}^+$ *to each non-negative integer* a *such that for each* $b \geq 0$ *it is* $\Rightarrow^* (a,b) = 1$ *if and only if* $b < Tb_{\Rightarrow^*}(a)$.

Proof. Due to Remark 2.1 we can only put $Tb_{\Rightarrow^*}(a) = Tb_{\Rightarrow^*}(a,0,0)$ *where* $Tb_{\Rightarrow^*}(a,c,d)$ *is a function defined in the same way as in Theorem 1.* □

We define the notions of tables of maximal b on the basis of just proved theorems.

Definition 1.

1. *Let* $\approx$ *be an equivalency quantifier and let* $c \geq 0$ *and* $d \geq 0$ *be the natural numbers. Then the* partial table of maximal b *for the quantifier* $\approx$ *and for the couple* $\langle c,d \rangle$ *is the function* $Tbp_{\approx,c,d}$ *defined such that*

$$Tbp_{\approx,c,d}(a) = Tb_{\approx}(a,c,d)$$

   where $Tb_{\approx}(a,c,d)$ *is the function from the Theorem 1.*
2. *Let* $\Rightarrow^*$ *be an implicational quantifier. Then the function* $Tb_{\Rightarrow^*}(a)$ *from the Theorem 2 is a* table of maximal b *for the implicational quantifier* $Tb_{\Rightarrow^*}$.
3. *Let* T *be a partial table of maximal* b *or a table of maximal* b. *Then a* step in the table T *is each such* $a \geq 0$ *for which it is* $T(a) < T(a+1)$.

It is important that the function $Tb_{\Rightarrow^*}$ makes it possible to use a simple test of inequality instead of a rather complex computation. For example we can use inequality $b < Tb_{\Rightarrow^!_{p,\alpha,s}}(a)$ instead of the condition

$$\sum_{i=a}^{a+b} \frac{(a+b)!}{i!(a+b-i)!} p^i (1-p)^{a+b-i} \leq \alpha \wedge a \geq s$$

for quantifier $\Rightarrow^!_{p,\alpha,s}$ of lower critical implication, see [12]. An other form of the table of critical frequencies for implicational quantifier is defined in [2].

Note that if we apply a corresponding data mining procedure then the computation of the function $Tb_{\Rightarrow^!_{p,\alpha,s}}$ usually requires much less effort than the evaluation of all conditions $\sum_{i=a}^{a+b} \frac{(a+b)!}{i!(a+b-i)!} p^i (1-p)^{a+b-i} \leq \alpha \wedge a \geq s$. It depends on the cardinality of the set of relevant questions to be automatically generated and verified, for more details see namely [12].

Let us remark that it can be $Tb_{\Rightarrow^*}(a) = \infty$. A trivial example gives the quantifier $\Rightarrow^T$ defined such that $\Rightarrow^T (a,b) = 1$ for each couple $\langle a, b\rangle$. Then it is $Tb_{\Rightarrow^T}(a) = \infty$ for each a.

The partial table of maximal b and table of maximal b are called *tables of critical frequencies*. Further tables of critical frequencies for Σ-double implicational quantifiers and for Σ-equivalence quantifiers are defined in [8].

3 Classical Definability and TCF

3.1 Association Rules and Observational Calculi

Monadic observational predicate calculi (MOPC for short) are defined and studied in [2] as a special case of observational calculi. MOPC can be understood as a modification of classical predicate calculus such that only finite models (i.e. data structures in which the formulas are interpreted) are admitted and more quantifiers than $\forall$ and $\exists$ are used. These new quantifiers are called *generalized quantifiers*. The 4ft quantifiers are special case of generalized quantifiers.

Classical monadic predicate calculus (CMOPC for short) is a MOPC with only classical quantifiers. In other words it is a classical predicate calculus with finite models. The formulas $(\forall x)P_1(x)$ and $(\exists x)(\exists y)((x \neq y) \wedge P_1(x) \wedge \neg P_2(y))$ are examples of formulas of CMOPC.

If we add the 4ft-quantifiers to CMOPC we get MOPC the formulas of which correspond to association rules. Formulas $(\Rightarrow_{p,Base}\ x)(P_1(x), P_2(x)$ and $(\Leftrightarrow_{p,Base}\ x)(P_1(x) \vee P_3(x), P_2(x) \wedge P_4(x))$ are examples of such association rules. Values of these formulas can be defined in Tarski style see [2]. We suppose that the formulas are evaluated in {0,1} – data matrices (i.e. finite data structures), see example in Fig. 1 where predicates $P_1, \ldots, P_n$ are interpreted by columns – functions $f_1, \ldots, f_n$ respectively.

row of $\mathcal{M}$	P_1	P_2	...	P_n	$P_1 \vee P_3$	$P_2 \wedge P_4$
	f_1	f_2	...	f_n	$\max(f_1, f_3)$	$\min(f_2, f_4)$
o_1	1	0	...	1	0	1
o_2	0	1	...	1	1	0
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$
o_n	1	0	...	0	0	1

Fig. 1. Example of {0,1} – data matrix

The rule $(\approx x)(P_1(x) \vee P_3(x), P_2(x) \wedge P_4(x))$ can be written in various forms, e.g. $(\approx)(P_1 \vee P_3, P_2 \wedge P_4)$ or $P_1 \vee P_3 \approx P_2 \wedge P_4$. Its evaluation is in any case based on the value $\approx (a, b, c, d)$ where $\langle a, b, c, d\rangle$ is the 4ft-table of $P_1(x) \vee P_3(x)$ and $P_2(x) \wedge P_4(x)$ in the data matrix in question. The same is true for each association rule of the form $(\approx x)(\varphi(x), \psi(x))$.

Let us remark that the association rule of the form like $A(1, 2, 3) \approx B(4, 5)$ can be understood (informally speaking) like the rule $A_1 \vee A_2 \vee A_3 \approx B_4 \vee B_5$ where A_1 is a predicate corresponding to the basic Boolean attribute $A(1)$ etc.

3.2 Definability and Associated Function

The natural question is what association rules are classically definable. We say that the association rule $(\approx\ x)\ (\varphi(x),\ \psi(x))$ – formula of MOPC is classically definable if there is a formula Φ of CMOPC with equality such that Φ is logically equivalent to $(\approx x)(\varphi(x), \psi(x))$. It means e.g. that the association rule $(\approx x)(P_1(x) \vee P_3(x), P_2(x) \wedge P_4(x))$ is equivalent to the formula created from the predicates $P_1(x)$, $P_2(x)$, $P_3(x)$, $P_4(x)$, propositional connectives $\neg, \vee, \wedge$, classical quantifiers $\exists, \forall$, and from the binary predicate of equality $=$. The precise formal definition is given in [2], see also [10]. If the association rule $(\approx x)(\varphi, \psi)$ is classically definable then we also say that the 4ft-quantifier $\approx x$ is classically definable and vice-versa.

The question of classical definability of (not only) association rules is solved by the Tharp's theorem proved in [2]. The Tharp's theorem is but too complex and general from the point of view of association rules. A more intuitive criterion of classical definability of association rules is proved in [4] see also [10]. This criterion is based on the associated function $\approx (a, b, c, d)$ of the 4ft-quantifier $\approx$.

The criterion uses the notion of interval in $\mathcal{N}^4$ where $\mathcal{N}$ is the set of all natural numbers. The *interval in* $\mathcal{N}^4$ is defined as the set

$$I = I_1 \times I_2 \times I_3 \times I_4$$

such that it is $I_j = \langle k, l)$ or $I_j = \langle k, \infty)$ for $j = 1, 2, 3, 4$ and $0 \leq k < l$ are natural numbers. The empty set $\emptyset$ is also the interval in $\mathcal{N}^4$.

The criterion of classical definability of association rules is given by the following theorem proved in [4], see also [10].

Theorem 3. *The 4ft-quantifier $\approx$ is classically definable if and only if there are K intervals $I_1, \ldots, I_K$ in $\mathcal{N}^4$, $K \geq 0$ such that it is for each 4ft table $\langle a, b, c, d \rangle$*

$$\approx (a, b, c, d) = 1 \quad \textit{iff} \quad \langle a, b, c, d \rangle \in \bigcup_{j=1}^{K} I_j.$$

3.3 Definability of Equivalency Quantifiers

We use the criterion of classical definability based on associated functions of 4ft-quantifiers to give a very intuitive necessary condition of classical definability of equivalency rules. This condition says that if the equivalency quantifier is definable then each partial table of maximal b of this quantifier has only finite number of steps. It is proved in the next theorem.

Theorem 4. *Let $\approx$ be an classically definable equivalency quantifier. Then each partial table of maximal b for the quantifier $\approx$ has only finite number of steps.*

Proof. We suppose that $\approx$ is classically definable quantifier. Thus according to the Theorem 3 there are K intervals $I_1, \ldots, I_K$ in $\mathcal{N}^4$, $K \geq 0$ such that it is for each 4ft table $\langle a, b, c, d \rangle$

$$\approx (a, b, c, d) = 1 \quad \textit{iff} \quad \langle a, b, c, d \rangle \in \bigcup_{j=1}^{K} I_j.$$

If $K = 0$ then it is $\approx (a, b, c, d) = 0$ for each 4ft table $\langle a, b, c, d \rangle$ and it is $Tbp_{\approx,c,d}(a) = 0$ for each a and for each partial table $Tbp_{\approx,c,d}$ of maximal b of $\approx$. It but means that each such partial table of maximal b has no step.

Let us suppose that $K > 0$ and that it is

$$I_j = \langle a_j, A_j \rangle \times \langle b_j, B_j \rangle \times \langle c_j, C_j \rangle \times \langle d_j, D_j \rangle$$

for $j = 1, \ldots, K$. Suppose that there are u and v such that the partial table $Tbp_{\approx,u,v}$ of maximal b has infinitely many steps. It means that for each natural $n > 0$ there are $a > n$ and $b > n$ such that $\approx (a, b, u, v) = 1$. Thus there must be $m \in 1, \ldots, K$ such that

$$I_m = \langle a_m, \infty \rangle \times \langle b_m, \infty \rangle \times \langle c_m, C_m \rangle \times \langle d_m, D_m \rangle$$

and $u \in \langle c_m, C_m \rangle$ and $v \in \langle d_m, D_m \rangle$. The partial table $Tbp_{\approx,u,v}$ of maximal b has infinitely many steps. It means that there is $a > a_m$ such that $Tbp_{\approx,u,v}(a) < Tbp_{\approx,u,v}(a + 1)$. Thus it is

$$\approx (a, Tbp_{\approx,u,v}(a + 1), u, v) = 0.$$

Let us denote $b' = \max(b_m, Tbp_{\approx,u,v}(a + 1))$, thus it is $\approx (a, b', u, v) = 0$ because of $\approx$ is equivalency (see also point II in the proof of the Theorem 1).

It is however $\langle a, b', u, v \rangle \in I_m$ and it means that $\approx (a, b', u, v) = 1$. It is a contradiction that finishes the proof. □

3.4 Definability of Implicational Quantifiers

The next theorem shows that the necessary condition of definability of equivalency rules proved in Theorem 4 is also the sufficient condition of definability of implicational quantifiers.

Theorem 5. *Let $\Rightarrow^*$ be an implicational quantifier. Then $\Rightarrow^*$ is classically definable if and only if its table of maximal b has only finite number of steps.*

Proof. Let $Tb_{\Rightarrow^*}$ be a table of maximal b of $\Rightarrow^*$.

If $\Rightarrow^*$ is classically definable then we prove that $Tb_{\Rightarrow^*}$ has only finite number of steps in a similar way like we proved in the Theorem 4 that the partial table $Tbp_{\approx}(a, c_0, d_0)$ of maximal b has finite number of steps.

Let us suppose that $Tb_{\Rightarrow^*}$ has K steps where $K \geq 0$ is a natural number. We prove that $Tb_{\Rightarrow^*}$ is classically definable.

First let us suppose that $K = 0$. We distinguish two cases: $Tb_{\Rightarrow^*}(1) = 0$ and $Tb_{\Rightarrow^*}(1) > 0$.

If it is $Tb_{\Rightarrow^*}(1) = 0$ then it is also $Tb_{\Rightarrow^*}(0) = 0$ (there is no step). It but means that $\Rightarrow^* (a, b, c, d) = 0$ for each 4ft table $\langle a, b, c, d\rangle$ because of it cannot be $b < 0$. Thus it is $\Rightarrow^* (a, b, c, d) = 1$ if and only if $\langle a, b, c, d\rangle \in \emptyset$. The empty set $\emptyset$ is but also the interval in $\mathcal{N}^4$ and the quantifier $\Rightarrow^*$ is according to the Theorem 3 classically definable.

If it is $K = 0$ and $Tb_{\Rightarrow^*}(1) > 0$ then it is $\Rightarrow^* (a, b, c, d) = 1$ if and only if

$$\langle a, b, c, d\rangle \in \langle 0, \infty) \times \langle 0, Tb_{\Rightarrow^*}(1)) \times \langle 0, \infty) \times \langle 0, \infty)$$

and thus the quantifier $\Rightarrow^* (a, b, c, d)$ is definable according to the Theorem 3.

Let us suppose that $S > 0$ is a natural number and that

$$0 \leq a_1 < a_2 < \cdots < a_S$$

are all the steps in $Tb_{\Rightarrow^*}$. We will define intervals $I_1, I_2, \ldots, I_{S+1}$ in the following way.

If $Tb_{\Rightarrow^*}(a_1) = 0$ then $I_1 = \emptyset$ otherwise

$$I_1 = \langle 0, a_1 + 1) \times \langle 0, Tb_{\Rightarrow^*}(a_1)) \times \langle 0, \infty) \times \langle 0, \infty).$$

For $j = 2, \ldots, S$ we define

$$I_j = \langle a_{j-1}, a_j + 1) \times \langle 0, Tb_{\Rightarrow^*}(a_j)) \times \langle 0, \infty) \times \langle 0, \infty).$$

The interval I_{S+1} is defined such that

$$I_{S+1} = \langle a_S, \infty) \times \langle 0, Tb_{\Rightarrow^*}(a_S)) \times \langle 0, \infty) \times \langle 0, \infty).$$

Remember that $\Rightarrow^*$ is implicational. Thus it is clear that the intervals $I_1, I_2, \ldots, I_{S+1}$ are defined such that

$$\Rightarrow^* (a, b, c, d) = 1 \quad \textit{iff} \quad \langle a, b, c, d\rangle \in \bigcup_{j=1}^{S+1} I_j$$

and according to the Theorem 3 the quantifier $\Rightarrow^*$ is definable. This finishes the proof. □

4 Undefinability of Particular Quantifiers

There are the following examples of important 4ft-quantifiers:

- 4ft-quantifier $\Rightarrow_{p,Base}$ of founded implication defined in [2] for $0 < p \leq 1$ and $Base > 0$ by the condition

$$\frac{a}{a+b} \geq p \wedge a \geq Base$$

- 4ft-quantifier $\Rightarrow^{!}_{p,\alpha,Base}$ of lower critical implication defined in [2] for $0 < p \leq 1$, $0 < \alpha < 0.5$ and $Base > 0$ by the condition

$$\sum_{i=a}^{a+b} \binom{a+b}{i} p^i (1-p)^{a+b-i} \leq \alpha \wedge a \geq Base$$

- 4ft-quantifier $\Leftrightarrow_{p,Base}$ of founded double implication defined in [3] for $0 < p \leq 1$ and $Base > 0$ by the condition

$$\frac{a}{a+b+c} \geq p \wedge a \geq Base$$

- 4ft-quantifier $\equiv_{p,Base}$ of founded equivalence defined in [3] for $0 < p \leq 1$ and $Base > 0$ by the condition

$$\frac{a+d}{n} \geq p \wedge a \geq Base$$

- Fisher's quantifier $\sim_{\alpha,Base}$ defined in [2] for $0 < \alpha < 0.5$ and $Base > 0$ by the condition

$$\sum_{i=a}^{\min(r,k)} \frac{\binom{k}{i}\binom{n-k}{r-i}}{\binom{r}{n}} \leq \alpha \ \wedge \ ad > bc \ \wedge \ a \geq Base$$

- 4ft-quantifier $\sim^{+}_{p,Base}$ of above average dependence defined in [11] for $0 < p$ and $Base > 0$ by the condition

$$\frac{a}{a+b} \geq (1+p)\frac{a+c}{a+b+c+d} \wedge a \geq Base$$

- The 4ft-quantifier $\rightarrow_{conf,sup}$ corresponding to the "classical" association rule with the confidence $conf$ and the support sup [1] defined by the condition

$$\frac{a}{a+b} \geq conf \wedge \frac{a}{n} \geq sup$$

All these quantifiers are classically undefinable. We will prove it for 4ft-quantifiers $\Rightarrow_{p,Base}$ of founded implication, $\Rightarrow^{!}_{p,\alpha,Base}$ of lower critical impli-

cation, and for Fisher's quantifier $\sim_{\alpha,Base}$. The proof for the other listed 4ft-quantifiers is similar. The proof of classical undefinability of additional 4ft-quantifiers is given in [4].

First we prove that the 4ft-quantifiers $\Rightarrow_{p,Base}$ of founded implication and the 4ft-quantifier $\Rightarrow^{!}_{p,\alpha,Base}$ of lower critical implication are not classically definable. We will use the following lemmas.

Lemma 1. *Let $\Rightarrow^*$ be an implicational quantifier that satisfies the conditions*

(a) There is $A \geq 0$ such that for each $a \geq A$ there is b satisfying $\Rightarrow^* (a,b) = 0$.
(b) For each $a \geq 0$ and $b \geq 0$ such that $\Rightarrow^* (a,b) = 0$ there is $a' \geq a$ for which it is $\Rightarrow^* (a',b) = 1$.

Then the table $Tb_{\Rightarrow^*}$ of maximal b of $\Rightarrow^*$ has infinitely many steps.

Proof. If the quantifier $\Rightarrow^*$ satisfies the condition (a) then it is $Tb_{\Rightarrow^*}(a) < \infty$ for each $a \geq 0$. Remember that it is $\Rightarrow^* (a, Tb_{\Rightarrow^*}(a)) = 0$. If the quantifier $\Rightarrow^*$ satisfies the condition (b) then there is for each $a > A$ an $a' > a$ such that it is $\Rightarrow^* (a', Tb_{\Rightarrow^*}(a)) = 1$. Thus it is $\Rightarrow^* (a, Tb_{\Rightarrow^*}(a)) = 0$ and also it is $\Rightarrow^* (a', Tb_{\Rightarrow^*}(a)) = 1$. It means that between a and a' there must be a step s of the table $Tb_{\Rightarrow^*}$.

We have proved that for each $a > A$ there is a step $s \geq a$ of the table $Tb_{\Rightarrow^*}$. It but means that the table $Tb_{\Rightarrow^*}$ has infinitely many steps. This finishes the proof. □

Lemma 2. *Let us suppose that $\approx$ is an equivalency quantifier and c_0 and d_0 are natural numbers such that the following conditions are satisfied.*

(a) There is $A \geq 0$ such that for each $a \geq A$ there is b for which it is satisfied $\approx (a,b,c_0,d_0) = 0$.
(b) For each $a \geq 0$ and $b \geq 0$ such that $\approx (a,b,c_0,d_0) = 0$ there is $a' \geq a$ for which it is $\approx (a',b,c,d) = 1$.

Then the partial table $Tbp_{\approx}(a,c_0,d_0)$ of maximal b of $\approx$ has infinitely many steps.

Proof. The proof is similar to the proof of the Lemma 1. □

Lemma 3. *Let us suppose that $0 < p < 1$ and that $i \geq 0$ is a natural number. Then it is*

$$\lim_{K\to\infty} \binom{K}{i} p^i (1-p)^{K-i} = 0.$$

Proof. It is:

$$\begin{aligned}\binom{K}{i} p^i (1-p)^{K-i} &\leq K^i p^i (1-p)^K (1-p)^{-i} \\ &= K^i (1-p)^K \left(\frac{p}{1-p}\right)^i.\end{aligned}$$

Thus it is enough to prove that for $r \in (0,1)$ *and* $i \geq 0$ *it is*

$$\lim_{K\to\infty} K^i r^K = 0.$$

To prove this it is enough to prove that for $r \in (0,1)$*, real* x *and a natural* $i \geq 0$ *it is*

$$\lim_{x\to\infty} x^i r^x = 0.$$

It is $\lim_{x\to\infty} x^i = \infty$, $\lim_{x\to\infty} r^x = 0$ *and thus according to the l'Hospital's rule it is*

$$\begin{aligned}\lim_{x\to\infty} x^i r^x &= \lim_{x\to\infty} \frac{x^i}{r^{-x}} = \lim_{x\to\infty} \frac{(x^i)^{(i)}}{(r^{-x})^{(i)}} \\ &= \lim_{x\to\infty} \frac{i!}{(-\ln r)^i r^{-x}} = \lim_{x\to\infty} r^x = 0,\end{aligned}$$

where $(x^i)^{(i)}$ *is an* i*-th derivation of* x^i *and analogously for* $(r^{-x})^{(i)}$*. This finishes the proof.* □

Lemma 4. *Let us suppose* $a \geq 0$ *and* $b \geq 0$ *are natural numbers. Then it is for each* $k \in \langle 0, b\rangle$ *and* $0 < p < 1$

$$\lim_{a\to\infty} \binom{a+b}{a+k} p^{a+k}(1-p)^{b-k} = 0.$$

Proof. It is:

$$\begin{aligned}&\binom{a+b}{a+k} p^{a+k}(1-p)^{b-k} \\ &= \binom{a+b}{a+b-(a+k)} p^{a+k}(1-p)^{b-k} \\ &= \binom{a+b}{b-k} p^{a+k}(1-p)^{b-k} \\ &\leq (a+b)^{b-k} p^{a+k}(1-p)^{b-k}.\end{aligned}$$

Thus it is enough to prove that it is

$$\lim_{a\to\infty} (a+b)^{b-k} p^a = 0.$$

The proof of this assertion is similar to the proof of the assertion

$$\lim_{K\to\infty} K^i r^K = 0.$$

in the Lemma 3. □

Lemma 5.

1. *The 4ft-quantifier* $\Rightarrow_{p,Base}$ *of founded implication satisfies the condition a from the Lemma 1 for each* $0 < p \leq 1$ *and* $Base > 0$.
2. *The 4ft-quantifier* $\Rightarrow^{!}_{p,\alpha,Base}$ *of lower critical implication satisfies the condition a from the Lemma 1 for each* $0 < p < 1$, $0 < \alpha < 0.5$ *and* $Base > 0$.

Proof.

1. *We have to prove that there is* $A \geq 0$ *such that for each* $a \geq A$ *there is* b *such that* $\Rightarrow_{p,Base} (a, b) = 0$ *for each* $0 < p \leq 1$ *and* $Base > 0$. *Let us remember that the 4ft-quantifier* $\Rightarrow_{p,Base}$ *is defined by the condition* $\frac{a}{a+b} \geq p \wedge a \geq Base$·
   Let be $A > Base$ *and* $a \geq A$. *Then we choose* b' *such that* $b' > \frac{a - p*a}{p}$. *Then it is* $\Rightarrow_{p,Base} (a, b') = 0$.
2. *We have to prove that there is* $A \geq 0$ *such that for each* $a \geq A$ *there is* b *such that* $\Rightarrow^{!}_{p,\alpha,Base} (a, b) = 0$ *for each* $0 < p < 1$, $0 < \alpha < 0.5$ *and* $Base > 0$. *Let us remember that the 4ft-quantifier* $\Rightarrow^{!}_{p,\alpha,Base}$ *is defined by the condition*

$$\sum_{i=a}^{a+b} \binom{a+b}{i} p^i (1-p)^{a+b-i} \leq \alpha \wedge a \geq Base.$$

   Let be $A > Base$ *and* $a \geq A$. *We show that there is a natural* b *such that*

$$\sum_{i=a}^{a+b} \binom{a+b}{i} p^i (1-p)^{a+b-i} > \alpha.$$

   It is $\sum_{i=a}^{a+b} \binom{a+b}{i} p^i (1-p)^{a+b-i} > \alpha$ *if and only if*

$$\sum_{i=0}^{a-1} \binom{a+b}{i} p^i (1-p)^{a+b-i} \leq 1 - \alpha$$

   because of $\sum_{i=0}^{a+b} \binom{a+b}{i} p^i (1-p)^{a+b-i} = 1$.
   According to the lemma 3 there is a natural $V > a$ *such that it is*

$$\binom{V}{i} p^i (1-p)^{V-i} \leq \frac{1-\alpha}{a}$$

   for $i = 0, \ldots, a-1$. *Thus it is*

$$\sum_{i=0}^{a-1} \binom{V}{i} p^i (1-p)^{V-i} \leq 1 - \alpha$$

   Let us choose $b = V - a$. *It means*

$$\sum_{i=0}^{a-1} \binom{a+b}{i} p^i (1-p)^{a+b-i} \leq 1-\alpha$$

and it finishes the proof.

□

Lemma 6.

1. *The 4ft-quantifier* $\Rightarrow_{p,Base}$ *of founded implication satisfies the condition b from the lemma 1 for each* $0 < p \leq 1$ *and* $Base > 0$.
2. *The 4ft-quantifier* $\Rightarrow^{!}_{p,\alpha,Base}$ *of lower critical implication satisfies the condition b from the lemma 1 for each* $0 < p < 1$, $0 < \alpha < 0.5$ *and* $Base > 0$.

Proof.

1. *We have to prove that for each* $a \geq 0$, $b \geq 0$ *such that* $\Rightarrow_{p,Base}(a,b) = 0$ *there is* $a' \geq a$ *for that it is* $\Rightarrow_{p,Base}(a',b) = 1$. *The proof is trivial, we use the fact that* $\lim_{a\to\infty} \frac{a}{a+b} = 1$.
2. *We have to prove that for each* $a \geq 0$ *and* $b \geq 0$ *such that it is satisfied* $\Rightarrow^{!}_{p,\alpha,Base}(a,b) = 0$ *there is* $a' \geq a$ *for that it is* $\Rightarrow^{!}_{p,\alpha,Base}(a',b) = 1$.
   Let us suppose that $\Rightarrow^{!}_{p,\alpha,Base}(a,b) = 0$. *It means that* $a < Base$ *or* $\sum_{i=a}^{a+b} \binom{a+b}{i} p^i (1-p)^{a+b-i} > \alpha$.
   According to the Lemma 4 there is natural n *such that for each* e, $e > n$ *and* $k = 0, \ldots, b$ *it is*

$$\binom{e+b}{e+k} p^{e+k} (1-p)^{b-k} < \frac{\alpha}{b+1}.$$

   Let us choose $a' = \max\{a, n, Base\}$. *Then it is* $a' \geq Base$ *and also*

$$\sum_{i=a'}^{a'+b} \binom{a'+b}{i} p^i (1-p)^{a'+b-i} = \sum_{k=0}^{b} \binom{a'+b}{a'+k} p^{a'+k} (1-p)^{b-k} < \alpha.$$

   Thus it is $\Rightarrow^{!}_{p,\alpha,Base}(a',b) = 1$ *and it finishes the proof.*

□

Theorem 6. *The 4ft-quantifier* $\Rightarrow_{p,Base}$ *of founded implication is not classically definable for each* $0 < p \leq 1$ *and* $Base > 0$.

The 4ft-quantifier $\Rightarrow^{!}_{p,\alpha,Base}$ *of lower critical implication is not classically definable for each* $0 < p < 1$, $0 < \alpha < 0.5$ *and* $Base > 0$.

Proof. The table of maximal b of the 4ft-quantifier $\Rightarrow_{p,Base}$ *of founded implication has infinitely many steps according to the Lemmas 2, 5 and 6. Thus it is not classically definable according to the Theorem 5.*

The proof for the quantifier $\Rightarrow^{!}_{p,\alpha,Base}$ *is analogous.* □

Now we prove that the the Fisher's quantifier $\sim_{\alpha,Base}$, is not classically definable. Let us remember that it is defined for $0 < \alpha < 0.5$ and $Base > 0$ by the condition

$$\sum_{i=a}^{\min(r,k)} \frac{\binom{k}{i}\binom{n-k}{r-i}}{\binom{r}{n}} \leq \alpha \ \wedge \ ad > bc \ \wedge \ a \geq Base\cdot$$

We use the following results from [2].

Definition 2. *(see [2]) The equivalency quantifier* $\approx$ *is* saturable *if it satisfies:*

1. *For each 4ft-table* $\langle a, b, c, d\rangle$ *with* $d \neq 0$ *there is* $a' \geq a$ *such that* $\approx (a', b, c, d) = 1$.
2. *For each 4ft-table* $\langle a, b, c, d\rangle$ *with* $a \neq 0$ *there is* $d' \geq d$ *such that* $\approx (a, b, c, d') = 1$.
3. *For each 4ft-table* $\langle a, b, c, d\rangle$ *there is a 4ft-table* $\langle a', b', c', d'\rangle$ *such that it is* $\approx (a', b', c', d') = 0$.

Theorem 7. *The Fisher's quantifier* $\sim_{\alpha,Base}$ *is saturable for* $0 < \alpha < 0.5$ *and* $Base > 0$.

Proof. See [2]. □

Lemma 7. *There are natural numbers* c_0 *and* d_0 *such that the Fisher's quantifier* $\sim_{\alpha,Base}$ *satisfies the conditions a and b from the Lemma 2.*

Proof. We prove that the conditions a and b are satisfied for $c_0 = 1$ *and* $d_0 = 1$. *We have to prove*

(a) There is $A \geq 0$ *such that for each* $a \geq A$ *there is b for which it is satisfied* $\sim_{\alpha,Base} (a, b, 1, 1) = 0$.

(b) For each $a \geq 0$ *and* $b \geq 0$ *such that* $\approx (a, b, 1, 1) = 0$ *there is* $a' \geq a$ *for which it is* $\sim_{\alpha,Base} (a', b, 1, 1) = 1$.

Let us choose $b = a + 1$ *for each* $a \geq Base$, *then it is* $ad < bc$ *and thus it is* $\sim_{\alpha,Base} (a, b, 1, 1) = 0$. *It means that the condition a is satisfied.*

The condition b follows from the fact that the Fisher's quantifier $\sim_{\alpha,Base}$ *is saturable, see Theorem 7.* □

Theorem 8. *The Fisher's quantifier* $\sim_{\alpha,Base}$ *is not classically definable for each* $0 < \alpha < 0.5$ *and* $Base > 0$.

Proof. The partial table $Tbp_{\sim_{\alpha,Base}}(a, 1, 1)$ *of maximal b of* $\sim_{\alpha,Base}$ *has infinitely many steps according to the Lemmas 7 and 2. Thus the Fisher's quantifier is not classically definable according to the Theorem 4.* □

Let us remark that according to the Theorem 5 the implicational quantifier $\Rightarrow^*$ is classically definable if and only if its table $Tb_{\Rightarrow^*}$ of maximal b has only finite number of steps. Let us suppose that we analyze a data matrix with N rows. We can define a new quantifier $\Rightarrow^{*,N}$ such that

$$Tb_{\Rightarrow^{*,N}}(a) = \begin{cases} Tb_{\Rightarrow^*}(a) & \text{if } a \leq N \\ \\ Tb_{\Rightarrow^*}(N+1) & \text{if } a > N \end{cases}$$

It is clear that the table $Tb_{\Rightarrow^{*,N}}$ of maximal b of 4ft-quantifier $\Rightarrow^{*,N}$ has finite number of steps and thus the 4ft-quantifier $\Rightarrow^{*,N}$ is classically definable. It is also clear that it is

$$\Rightarrow^{*,N}(a,b) \;=\; \Rightarrow^*(a,b)$$

for each $a \leq N$ and for each $b \geq 0$.

In this way we can replace general implicational 4ft-quantifier by a classically definable 4ft-quantifier that is equivalent to the given quantifier what concerns behavior on the given data matrix. This approach can be used also for the additional 4ft-quantifiers. The construction of the corresponding formula of the classical predicate calculus is described in [10].

5 Conclusions

We have presented a simple criterion of classical definability of important types of association rules. This criterion is based on the table of critical frequencies that is itself important tool for verification of association rules. The presented criterion depends on the class of association rules (i.e. the class of 4ft-quantifiers) we deal with.

Let us remark that there are further interesting and practically useful relations of tables of critical frequencies, classes of association rules, logical properties of association rules and properties of association rules in the data with missing information. They are partly published in [2, 5–7] and in more details investigated in [4, 8]. An overview of related results is in [11].

Acknowledgement

The work described here has been supported by the grant 201/05/0325 of the Czech Science Foundation and by the project IGA 25/05 of University of Economics, Prague.

References

1. Aggraval R, et al. (1996) Fast Discovery of Association Rules. In: Fayyad UM et al. (eds.) Advances in Knowledge Discovery and Data Mining. AAAI, Menlo Park, CA, 307–328
2. Hájek P, Havránek T (1978) Mechanising Hypothesis Formation – Mathematical Foundations for a General Theory. Springer, Berlin Heidelberg New York
3. Hájek P, Havránek T, Chytil M (1983) GUHA Method. Academia, Prague (in Czech)
4. Rauch J (1986) Logical Foundations of Hypothesis Formation from Databases. PhD Thesis, Mathematical Institute of the Czechoslovak Academy of Sciences, Prague (in Czech)
5. Rauch J (1997) Logical Calculi for Knowledge Discovery in Databases. In: Zytkow J, Komorowski J (eds.) Principles of Data Mining and Knowledge Discovery. Springer, Berlin Heidelberg New York, 47–57
6. Rauch J (1998) Classes of Four-Fold Table Quantifiers. In: Zytkow J, Quafafou M (eds.) Principles of Data Mining and Knowledge Discovery. Springer, Berlin Heidelberg New York, 203–211
7. Rauch J (1998) Four-Fold Table Calculi and Missing Information. In: Wang P (ed.). JCIS '98, Association for Intelligent Machinery, Vol. II. Duke University, Durham
8. Rauch J (1998) Contribution to Logical Foundations of KDD. Assoc. Prof. Thesis, Faculty of Informatics and Statistics, University of Economics, Prague (in Czech)
9. Rauch J (2005) Logic of Association Rules. Applied Intelligence 22, 9–28
10. Rauch J (2005) Definability of Association Rules in Predicate Calculus. In: Lin T Y, Ohsuga S, Liau C J, Hu X (eds): Foundatuons and Novel Approaches in Data Mining. Springer, Berlin Heidelberg New York, 23–40
11. Rauch J (2007) Classes of Association Rules – an Overview. In: Data Mining: Foundations and Practice. Springer, Berlin Heidelberg New York
12. Rauch J, Šimůnek M (2005) An Alternative Approach to Mining Association Rules. In: Lin T Y, Ohsuga S, Liau C J, and Tsumoto S (eds.) Foundations of Data Mining and Knowledge Discovery. Springer, Berlin Heidelberg New York, pp. 219–238
13. Tharp L H (1973) The Characterisation of Monadic Logic. Journal of Symbolic Logic 38, 481–488

Classes of Association Rules: An Overview

Jan Rauch

University of Economics, Prague, nám. W. Churchilla 4 130 67 Prague,
Czech Republic
rauch@vse.cz

Summary. Chapter concerns theoretical foundations of association rules. We deal with more general rules than the classical association rules related to market baskets are. Various theoretical aspects of association rules are introduced and several classes of association rules are defined. Implicational and double implicational rules are examples of such classes. It is shown that there are practically important theoretical results related to particular classes. Results concern deduction rules in logical calculi of association rules, fast evaluation of rules corresponding to statistical hypotheses tests, missing information and definability of association rules in classical predicate calculi.

1 Introduction

The goal of this chapter is to contribute to theoretical foundations of data mining. We deal with association rules. We are however interested with more general rules than the classical association rule [1] inspired by market basket are. We understand the association rule as an expression $\varphi \approx \psi$ where φ and ψ are Boolean attributes derived from columns of an analysed data matrix. The intuitive meaning of the association rule $\varphi \approx \psi$ is that Boolean attributes φ and ψ are associated in a way corresponding to the symbol $\approx$. The symbol $\approx$ is called 4ft-quantifier. It is associated to a condition related to the (fourfold) contingency table of φ and ψ in the analysed data matrix.

Let us emphasize that there is an approach to mining association rules of the form $\varphi \approx \psi$ that does not use the widely known a priori algorithm [1]. The association rules $\varphi \approx \psi$ can be very effectively mined by an algorithm based on representation of analysed data by suitable strings of bits [15]. This approach makes possible to compute very fast necessary contingency tables. This way it is possible to efficiently mine also for additional patterns verified on the basis of various contingency tables [16, 17].

The association rules of the form $\varphi \approx \psi$ were defined and studied in [2] where several classes of association rules were introduced. Additional classes

J. Rauch: *Classes of Association Rules: An Overview*, Studies in Computational Intelligence (SCI) **118**, 315–337 (2008)
www.springerlink.com

of association rules were defined and studied in [7] and [11]. The examples are the classes of implicational [2] and of double implicational [11] rules. The class of *implicational rules* is defined such that it comprises the association rules $\varphi \approx \psi$ expressing in a reasonable way the fact that φ implies ψ. The class of *double implicational rules* is defined such that it comprises the association rules $\varphi \approx \psi$ that in a reasonable way express the fact that both φ implies ψ and ψ implies φ i.e. that φ and ψ are somehow equivalent.

Various aspects of association rules of the form $\varphi \approx \psi$ were studied:

- Logical calculi formulae of which correspond to association rules were defined. Some of these calculi are straightforward modification of classical predicate calculi [2], the other are more simple [12].
- Logical aspects of calculi of association rules e.g. decidability, deduction rules definability in classical predicate calculus were investigated, see namely [2, 12, 13].
- Association rules that correspond to statistical hypotheses tests were defined and studied [2].
- Several approaches to evaluation of association rules in data with missing information were investigated [2, 7].
- Software tools for mining all kinds of such association rules were implemented and applied [3, 5, 15].

This chapter concerns theoretical aspects of association rules. It was shown that most of theoretically interesting and practically important results concerning association rules are related to classes of association rules. Goal of this chapter is to give an overview of important classes of association rules and their properties. Both already published results are mentioned and new results are introduced.

The association rules of the form $\varphi \approx \psi$ including rules corresponding to statistical hypotheses tests are described in Sect. 2. An overview of classes of association rules is in Sect. 3. Evaluation of association rules corresponding to some hypotheses tests (e.g. to Fisher's test) requires complex computation. The complex computation can be avoided by tables of critical frequencies that are again closely related to classes of rules, see Sect. 4. Logical calculi formulae of which correspond to association rules are introduced in Sect. 5. Important deduction rules concerning association rules and their relation to classes of rules are introduced in Sect. 6. There are also interesting results concerning possibilities of definition of association rules by means of the classical predicate calculus see Sect. 7. Problem of evaluation of association rules in data with missing information were investigated [2, 10]. Results are also closely related to the classes of association rules see Sect. 8.

Let us emphasize that the goal of the chapter is to give an overview of important results related to the classes of association rules. Thus the whole chapter is written informally.

2 Association Rules

The association rule is an expression $\varphi \approx \psi$ where φ and ψ are Boolean attributes. The symbol $\approx$ is *4ft-quantifier*. It denotes a condition concerning a fourfold contingency table of φ and ψ. The rule $\varphi \approx \psi$ means that the Boolean attributes φ and ψ are associated in the way given by 4ft-quantifier $\approx$. The Boolean attributes φ and ψ are created from basic Boolean attributes and from propositional connectives $\vee$, $\wedge$ and $\neg$. The basic Boolean attribute is an expression of the form $A(\alpha)$ where $\alpha \subset \{a_1, \dots a_k\}$ and $\{a_1, \dots a_k\}$ is the set of all possible values (i.e. *categories*) of the attribute A.

The association rule concerns an analysed data matrix. Rows of the data matrix correspond to observed objects, columns correspond to attributes of objects. An example of the data matrix with columns – attributes $A, B, C, D, \dots, Z$ is data matrix $\mathcal{M}$ in Fig. 1.

There are also examples of basic Boolean attributes $A(a_1)$ and $B(b_3, b_4)$ in Fig. 1. The basic Boolean attribute $A(\alpha)$ is true in the row of the analysed data matrix if it is $a \in \alpha$ where a is the value of the attribute A in this row. The expression $A(a_1) \wedge B(b_3, b_4)$ is an example of the Boolean attribute derived from the attributes A and B. An example of the association rule is the expression

$$A(a_1) \wedge B(b_3, b_4) \approx C(c_1, c_{11}, c_{21}) \vee D(d_9, d_{10}, d_{11}).$$

The association rule $\varphi \approx \psi$ is true in the analysed data matrix $\mathcal{M}$ if and only if the condition associated to the 4ft-quantifier $\approx$ is satisfied for the *fourfold table* $4ft(\varphi,\psi, \mathcal{M})$ of φ and ψ in $\mathcal{M}$, see Table 1.

Here a is the number of the objects (i.e. the rows of $\mathcal{M}$) satisfying both φ and ψ, b is the number of the objects satisfying φ and not satisfying ψ, c is the number of objects not satisfying φ and satisfying ψ and d is the number of objects satisfying neither φ nor ψ . We write

$$4\text{ft}(\varphi,\ \psi, \mathcal{M}) = \langle a, b, c, d \rangle.$$

object	A	B	C	D	$\dots$	Z	$A(a_1)$	$B(b_3, b_4)$
o_1	a_1	b_8	c_{16}	d_9	$\dots$	z_{14}	1	0
o_2	a_5	b_4	c_7	d_2	$\dots$	z_9	0	1
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$
o_n	a_1	b_3	c_4	d_1	$\dots$	z_6	1	1

Fig. 1. Data matrix $\mathcal{M}$ and basic Boolean attributes $A(a_1)$, $B(b_3, b_4)$

Table 1. 4ft table 4ft(φ, ψ, $\mathcal{M}$)

$\mathcal{M}$	ψ	$\neg\psi$	
φ	a	b	r
$\neg\varphi$	c	d	
	k		n

In addition we sometimes use $r = a + b$, $k = a + c$ and $n = a + b + c + d$, see also Table 1. We use the abbreviation "4ft" instead of the expression "fourfold table". The notion *4ft table* is used for all possible tables $4ft(\varphi,\psi, \mathcal{M})$.

We write

$$\approx (a, b, c, d) = 1$$

if the condition corresponding to the 4ft-quantifier $\approx$ is satisfied for the quadruple $\langle a, b, c, d\rangle$ of integer non-negative numbers, otherwise we write

$$\approx (a, b, c, d) = 0.$$

The $\{0, 1\}$ – valued function $\approx$ (a, b, c, d) defined for all 4ft tables can be informally called *associated function of 4-ft quantifier* $\approx$. The precise definition of associated function of 4ft-quantifier is given in [2].

We write $Val(\varphi \approx \psi, \mathcal{M}) = 1$ if the association rule $\varphi \approx \psi$ is true in data matrix $\mathcal{M}$, otherwise we write $Val(\varphi \approx \psi, \mathcal{M}) = 0$.

Various kinds of dependencies of the Boolean attributes φ and ψ can be expressed by suitable 4ft-quantifiers. Some examples follow (see also [9]).

The 4ft-quantifier $\Rightarrow_{p,Base}$ of *founded implication* for $0 < p \leq 1$ and $Base > 0$ [2] is defined by the condition

$$\frac{a}{a+b} \geq p \wedge a \geq Base.$$

This means that at least $100p$ per cent of objects satisfying φ satisfy also ψ and that there are at least $Base$ objects of $\mathcal{M}$ satisfying both φ and ψ.

The 4ft-quantifier $\Rightarrow^{!}_{p,\alpha,Base}$ of *lower critical implication* for $0 < p \leq 1$, $0 < \alpha < 0.5$ and $Base > 0$ [2] is defined by the condition

$$\sum_{i=a}^{a+b} \binom{a+b}{i} p^i (1-p)^{a+b-i} \leq \alpha \wedge a \geq Base.$$

This corresponds to the statistical test (on the level α) of the null hypothesis $H_0 : P(\psi|\varphi) \leq p$ against the alternative one $H_1 : P(\psi|\varphi) > p$. Here $P(\psi|\varphi)$ is the conditional probability of the validity of ψ under the condition φ.

The 4ft-quantifier $\Leftrightarrow_{p,Base}$ of *founded double implication* for $0 < p \leq 1$ and $Base > 0$ [4] is defined by the condition

$$\frac{a}{a+b+c} \geq p \wedge a \geq Base.$$

This means that at least $100p$ per cent of objects satisfying φ or ψ satisfy both φ and ψ and that there are at least $Base$ objects of $\mathcal{M}$ satisfying both φ and ψ.

The 4ft-quantifier $\Leftrightarrow^{!}_{p,\alpha,Base}$ of *lower critical double implication* for $0 < p \leq 1$, $0 < \alpha < 0.5$ and $Base > 0$ [4] is defined by the condition

$$\sum_{i=a}^{a+b+c} \binom{a+b+c}{i} p^i (1-p)^{a+b+c-i} \leq \alpha \ \wedge \ a \geq Base.$$

The 4ft-quantifier $\equiv_{p,Base}$ of *founded equivalence* for $0 < p \leq 1$ and $Base > 0$ [4] is defined by the condition

$$\frac{a+d}{n} \geq p \wedge a \geq Base.$$

This means that φ and ψ have the same value (either *true* or *false*) for at least $100p$ per cent of all objects of $\mathcal{M}$ and that there are at least $Base$ objects satisfying both φ and ψ.

The 4ft-quantifier $\equiv^{!}_{p,\alpha,Base}$ of *lower critical equivalence* for $0 < p < 1$, $0 < \alpha < 0.5$ and $Base > 0$ [4] is defined by the condition

$$\sum_{i=a+d}^{n} \binom{n}{i} p^i (1-p)^n \leq \alpha \wedge \ a \geq Base.$$

The Fisher's quantifier $\sim_{\alpha,Base}$ (see Paragraph 4.4.20 in [2]) is defined for $0 < \alpha < 0.5$ and $Base > 0$ by the condition

$$\sum_{i=a}^{\min(r,k)} \frac{\binom{k}{i}\binom{n-k}{r-i}}{\binom{r}{n}} \leq \alpha \ \wedge \ ad > bc \ \wedge \ a \geq Base.$$

This corresponds to the statistical test (on the level α) of the null hypothesis of independence of φ and ψ against the alternative one of the positive dependence.

The 4ft-quantifier $\sim^{+}_{p,Base}$ of *above average dependence* for $0 < p$ and $Base > 0$ is defined by the condition

$$\frac{a}{a+b} \geq (1+p)\frac{a+c}{a+b+c+d} \wedge a \geq Base.$$

This means that the relative frequency of objects satisfying ψ among objects satisfying φ is at least $100p$ per cent higher than the relative frequency of objects satisfying ψ among all objects.

The 4ft-quantifier $\sim^{\mathrm{E}}_{\delta,Base}$ of E – equivalence [19] is defined by the condition

$$max\left(\frac{b}{a+b}, \frac{c}{d+c}\right) < \delta.$$

The "classical" association rule with the confidence $conf$ and the support sup can be expressed by the 4ft-quantifier $\rightarrow_{conf,sup}$ defined by the condition

$$\frac{a}{a+b} \geq conf \wedge \frac{a}{n} \geq sup.$$

Let us emphasize that all these 4ft-quantifiers are implemented in the procedure 4ft-Miner see [15].

3 Classes of Association Rules

3.1 Truth Preservation Condition and Implicational Quantifiers

Classes of association rules are defined by classes of 4ft quantifiers. The association rule $\varphi \approx \psi$ belongs to the *class of implicational association rules* if the 4ft quantifier $\approx$ belongs to the *class of implicational quantifiers*. We also say that the association rule $\varphi \approx \psi$ is an *implicational rule* and that the 4ft quantifier $\approx$ is an *implicational quantifier*. This is the same for other classes of association rules.

There are various important classes of 4ft quantifiers defined by *truth preservation conditions* [12]. We say that class $\mathcal{C}$ of 4ft-quantifiers is defined by truth preservation condition $TPC_{\mathcal{C}}$ if there is a Boolean condition $TPC_{\mathcal{C}}(a,b,c,d,a',b',c',d')$ concerning two fourfold contingency tables $\langle a,b,c,d\rangle$ and $\langle a',b',c',d'\rangle$ such that the following is true:
4ft quantifier $\approx$ *belongs to the class* $\mathcal{C}$ *if and only if*

$$\approx(a,b,c,d) = 1 \ \wedge \ \ TPC_{\mathcal{C}}(a,b,c,d,a',b',c',d')$$

implies

$$\approx(a',b',c',d') = 1$$

for all 4ft tables $\langle a,b,c,d\rangle$ *and* $\langle a',b',c',d'\rangle$.

The class of implicational quantifiers was defined in [2] by the *truth preservation condition* $TPC_{\Rightarrow}$ *for implicational quantifiers*. It is

$$TPC_{\Rightarrow} = a' \geq a \ \wedge \ b' \ \leq b.$$

It means that the 4ft quantifier $\approx$ is *implicational* if

$$\approx(a,b,c,d) = 1 \ \wedge \ \ a' \geq a \ \wedge \ b' \ \leq b$$

implies

$$\approx(a',b',c',d') = 1$$

for all 4ft tables $\langle a,b,c,d\rangle$ and $\langle a',b',c',d'\rangle$.

The truth preservation condition $TPC_{\Rightarrow}$ for implicational quantifiers (i.e. $a' \geq a \wedge b' \leq b$) means that the fourfold table $\langle a',b',c',d'\rangle$ is *"better from the point of view of implication"* than the fourfold table $\langle a,b,c,d\rangle$ (i-better according to [2]). If $\langle a,b,c,d\rangle$ is the fourfold table of φ and ψ in data matrix $\mathcal{M}$ and if $\langle a',b',c',d'\rangle$ is the fourfold table of φ and ψ in data matrix $\mathcal{M}'$, then the sentence "*better from the point of view of implication*" means: in data matrix $\mathcal{M}'$ there are more rows satisfying both φ and ψ than in data matrix $\mathcal{M}$ and in $\mathcal{M}'$ there are fewer rows satisfying φ and not satisfying ψ than in $\mathcal{M}$.

Thus if it is $a' \geq a \wedge b' \leq b$ then it is reasonable to expect that if the implicational association rule $\varphi \approx \psi$ (i.e. the rule expressing the implication by $\approx$) is true in the data matrix $\mathcal{M}$ then this rule is also true in data matrix

Table 2. Classes of association rules defined by truth preservation conditions

Class		Truth preservation condition	Examples
Implicational	$TPC_{\Rightarrow}$	$a' \geq a \ \wedge \ b' \leq b$	$\Rightarrow_{p,Base}$ $\Rightarrow^{!}_{p,\alpha,Base}$
Double implicational	$TPC_{\Leftrightarrow}$	$a' \geq a \ \wedge \ b' \leq b \ \wedge \ c' \leq c$	$\Leftrightarrow^{*}_{0.9,0.1}$ $\Leftrightarrow_{p,Base}$
Σ-double implicational	$TPC_{\Sigma,\Leftrightarrow}$	$a' \geq a \ \wedge \ b' + c' \leq b + c$	$\Leftrightarrow_{p,Base}$ $\Leftrightarrow^{!}_{p,\alpha,Base}$
Equivalency	$TPC_{\equiv}$	$a' \geq a \ \wedge \ b' \leq b \ \wedge \ c' \leq c \ \wedge \ d' \geq d$	$\sim_{\alpha,Base}$ $\equiv_{p,Base}$
Σ-equivalency	$TPC_{\Sigma,\equiv}$	$a' + d' \geq a + d \ \wedge \ b' + c' \leq b + c$	$\equiv_{p,Base}$ $\equiv^{!}_{p,\alpha,Base}$

$\mathcal{M}'$ that is better from the point of view of implication. This expectation is ensured for implicational quantifiers by the above given definition.

It is easy to prove that the 4ft-quantifier $\Rightarrow_{p,Base}$ of founded implication is implicational. It is proved in [2] that the 4ft-quantifier $\Rightarrow^{!}_{p,\alpha,Base}$ of lower critical implication is also implicational.

There are several additional important classes of association rules defined by truth preservation conditions, see [2, 4, 9, 11]. Overview of these classes and some examples are given in Table 2. The quantifiers $\Rightarrow_{p,Base}$, $\Rightarrow^{!}_{p,\alpha,Base}$, $\Leftrightarrow_{p,Base}$, $\equiv_{p,Base}$, and $\sim_{\alpha,Base}$ used in Table 2 as examples are defined in Sect. 2, the quantifier $\Leftrightarrow^{*}_{0.9,0.1}$ is explained below.

3.2 Double Implicational Quantifiers

The class of double implicational quantifiers is defined in [11] by the *truth preservation condition $TPC_{\Leftrightarrow}$ for double implicational quantifiers*:

$$TPC_{\Leftrightarrow} = a' \geq a \ \wedge \ b' \ \leq b \ \wedge \ c' \ \leq c.$$

We can see a reason for the definition of double implicational quantifier in an analogy to propositional calculus. If u and v are propositions and both $u \rightarrow v$ and $v \rightarrow u$ are true, then u is equivalent to v (the symbol "$\rightarrow$" is here a propositional connective of implication). Thus we can try to express the relation of equivalence of attributes φ and ψ using a "double implicational" 4ft-quantifier $\Leftrightarrow^{*}$ such that $\varphi \Leftrightarrow^{*} \psi$ if and only if both $\varphi \Rightarrow^{*} \psi$ and $\psi \Rightarrow^{*} \varphi$, where $\Rightarrow^{*}$ is a suitable implicational quantifier.

If we apply the truth preservation condition for implicational quantifier $TPC_{\Rightarrow}$ to $\varphi \Rightarrow^{*} \psi$, we obtain $a' \geq a \ \wedge \ b' \leq b$. If we apply it to $\psi \Rightarrow^{*} \varphi$, we obtain $a' \geq a \ \wedge \ c' \leq c$, ($c$ is here instead of b, see Table 1). This leads to the truth preservation condition for double implicational quantifiers $TPC_{\Leftrightarrow}$ see Table 2.

The class of Σ-double implicational quantifiers is defined to contain useful quantifiers $\Leftrightarrow_{p,Base}$ and $\Leftrightarrow^{!}_{p,\alpha,Base}$. They deal with the summa $b + c$ in the

same way as the quantifiers $\Rightarrow_{p,Base}$ and $\Rightarrow^{!}_{p,\alpha}$ deal with the frequency c from contingency table. Thus it is reasonable to use the $TPC_{\Sigma,\Leftrightarrow}$:

$$TPC_{\Sigma,\Leftrightarrow} = a' \geq a \ \wedge \ b' + c' \leq b + c.$$

The proof that $\Leftrightarrow_{p,Base}$ is Σ-double implicational is trivial. The proof that $\Leftrightarrow^{!}_{p,\alpha,Base}$ is Σ-double implicational is given in [12].

The condition $TPC_{\Sigma,\Leftrightarrow}$ is weaker than the condition $TPC_{\Leftrightarrow}$. Actually, if the pair of the contingency tables $\langle a, b, c, d\rangle$ and $\langle a', b', c', d'\rangle$ satisfies $TPC_{\Leftrightarrow}$ it also satisfies $TPC_{\Sigma,\Leftrightarrow}$. It means that the class of Σ-double implicational quantifiers is a subclass of the class of double implicational quantifiers:

Let us suppose that $\approx$ is a Σ-double implicational quantifier, we prove that $\approx$ is also double implicational. We have to prove that if $\approx (a, b, c, d) = 1$ and contingency tables $\langle a, b, c, d\rangle$ and $\langle a', b', c', d'\rangle$ satisfy $TPC_{\Leftrightarrow}$ then it is also $\approx (a', b', c', d') = 1$. Tables $\langle a, b, c, d\rangle$ and $\langle a', b', c', d'\rangle$ satisfy $TPC_{\Leftrightarrow}$ thus they satisfy also $TPC_{\Sigma,\Leftrightarrow}$. We suppose that $\approx$ is a Σ-double implicational quantifier, we have just shown that $\langle a, b, c, d\rangle$ and $\langle a', b', c', d'\rangle$ satisfy $TPC_{\Sigma,\Leftrightarrow}$ and thus it is $\approx (a, b, c, d) = 1$. This means that each Σ-double implicational quantifier is also double implicational.

The question is if there is a double implicational quantifier that is not Σ-double implicational quantifier. It is proved in [11] that there are such quantifiers. Let us define the quantifier $\Leftrightarrow^{*}_{0.9,\omega}$ for $0 < \omega$ such that

$$\Leftrightarrow^{*}_{0.9,\omega} (a, b, c, d) = \begin{cases} 1 \text{ iff } \ \frac{a}{a+b+\omega c} \geq 0.9 \ \wedge a + b + c > 0 \\ \\ 0 \text{ otherwise.} \end{cases}$$

It is easy to prove that $\Leftrightarrow^{*}_{0.9,\omega}$ is double implicational. Let us suppose $a = 90$, $b = 9$, $c = 2$, $d = 0$, and $0 < \omega < 0.5$. It means $b + \omega c < 10$ and we have

$$\frac{a}{a + b + \omega c} \ = \ \frac{90}{90 + b + \omega c} \ > \ \frac{90}{90 + 10} = 0.9 \text{ thus } \Leftrightarrow^{*}_{0.9,\omega} (90, 9, 2, 0) = 1.$$

Let us suppose that $\Leftrightarrow^{*}_{0.9,\omega}$ is Σ-double implicational. Then it must be also $\Leftrightarrow^{*}_{0.9,\omega} (90, 9 + 2, 0, 0) = 1$. It is but

$$\frac{90}{90 + 9 + 2 + \omega * 0} \ = \ \frac{90}{90 + 11} < 0.9, \text{ thus } \Leftrightarrow^{*}_{0.9,\omega} (90, 9, 2, 0) = 0.$$

This is a contradiction, we can conclude that $\Leftrightarrow^{*}_{0.9,\omega}$ is not Σ-double implicational for $0 < \omega < 0.5$.

Remark 3.1 *It is easy to prove for the implicational quantifier* $\Rightarrow^{*}$ *that the value* $\Rightarrow^{*} (a, b, c, d)$ *depends neither on* c *nor on* d. *It means that we can write only* $\Rightarrow^{*} (a, b)$ *instead of* $\Rightarrow^{*} (a, b, c, d)$ *for the implicational quantifier* $\Rightarrow^{*}$.

Remark 3.2 *It is also easy to prove for the double implicational quantifier* $\Leftrightarrow^{*}$ *that the value* $\Leftrightarrow^{*} (a, b, c, d)$ *does not depend on* d. *Thus we write only* $\Leftrightarrow^{*} (a, b, c)$ *instead of* $\Leftrightarrow^{*} (a, b, c, d)$ *for the double implicational quantifier* $\Leftrightarrow^{*}$.

The natural question is if there are some "pure double implicational" quantifiers. We say that the 4ft-quantifier $\Leftrightarrow^*$ is *pure double implicational* if there is an implicational quantifier $\Rightarrow^*$ such that

$$\Leftrightarrow^* (a, b, c, d) = 1 \text{ if and only if } \Rightarrow^* (a, b) = 1 \text{ and } \Rightarrow^* (a, c) = 1.$$

This and other related questions are solved in [11]. Summary of some related results is given in Sect. 3.4.

3.3 Equivalency Quantifiers

The class of equivalency quantifiers was defined under the name of associational quantifiers in [2] by the *truth preservation condition* $TPC_{\equiv}$:

$$TPC_{\equiv} = a' \geq a \ \wedge \ b' \leq b \ \wedge \ c' \leq c \ \wedge \ d' \geq d.$$

However the widely known "classical" association rules with confidence and support concerning market baskets defined in [1] are not associational in the sense of the $TPC_{\equiv}$ truth preservation condition. The easy proof is given e.g. in [11]. To avoid possible confusion we use the term *equivalency quantifiers*.

Again, we can find a reason for the term equivalency quantifiers in the analogy to the propositional logic. If u and v are propositions and both $u \rightarrow v$ and $\neg u \rightarrow \neg v$ are true, then u is equivalent to v. Thus we can try to express the relation of equivalence of the attributes φ and ψ using an "equivalence" 4ft-quantifier $\equiv^*$ such that $\varphi \equiv^* \psi$ if and only if $\varphi \Rightarrow^* \psi$ and $\neg\varphi \Rightarrow^* \neg\psi$, where $\Rightarrow^*$ is a suitable implicational quantifier. If we apply the truth preservation condition for implicational quantifiers $TPC_{\Rightarrow}$ to $\varphi \Rightarrow^* \psi$ we obtain $a' \geq a \ \wedge \ b' \leq b$, if we apply it to $\neg\varphi \Rightarrow^* \neg\psi$, we obtain $d' \geq d \ \wedge \ c' \leq c$, ($c$ is here instead of b and d is instead of a, see Table 1). This leads to the truth preservation condition $TPC_{\equiv}$ for equivalency quantifiers.

The class of Σ-equivalency quantifiers is defined to contain useful quantifiers $\equiv_{p,Base}$ and $\equiv^!_{p,\alpha,Base}$. Thus it is reasonable to use the preservation condition $TPC_{\Sigma,\equiv}$ for Σ-equivalency quantifiers:

$$TPC_{\Sigma,\equiv} = a' + d' \geq a + d \ \wedge \ b' + c' \leq b + c.$$

The proof that $\equiv_{p,Base}$ is Σ-equivalency is trivial. The proof that $\equiv^!_{p,\alpha,Base}$ is Σ-equivalency is given in [11] and it is similar to the proof that $\Leftrightarrow^!_{p,\alpha,Base}$ is Σ-double implicational given in [12].

It is easy to see that the condition $TPC_{\Sigma,\equiv}$ is weaker than the condition $TPC_{\equiv}$. It means that the class of Σ-equivalency quantifiers is a subclass of the class of equivalency quantifiers. We can define the class of pure equivalency quantifiers analogously to the class of pure double implicational quantifiers. There are interesting properties of these classes see [12], their detailed description is however out of the scope of this chapter. Some related results are in Sect. 3.4. It is proved in [2] that the Fisher's quantifier $\sim_{\alpha,Base}$ is equivalency (i.e. associational in the terminology of [2]) It is also proved in [12] that the Fisher's quantifier $\sim_{\alpha,Base}$ is not Σ-equivalency.

3.4 Additional Classes of Quantifiers

There are some additional classes of 4ft-quantifiers with interesting theoretical and practical properties that are not defined by truth preservation condition. An example is the class of symmetrical quantifiers, see [2]. The 4ft-quantifier $\approx$ is symmetrical if and only if

$$\approx (a,b,c,d) \quad = \quad \approx (a,c,b,d).$$

It means that the association rule $\varphi \approx \psi$ with symmetrical 4ft-quantifier $\approx$ is true if and only if the association rule $\psi \approx \varphi$ is true.

It is easy to prove that 4ft-quantifier $\Leftrightarrow_{p,Base}$ of founded double implication, 4ft-quantifier $\equiv_{p,Base}$ of founded equivalence, the Fisher's quantifier $\sim_{\alpha,Base}$ and 4ft-quantifier $\sim^{+}_{p,Base}$ of above average dependence are symmetrical.

An other interesting class of 4ft quantifiers is the class of the 4ft quantifiers with F-property. This class is defined such that the quantifiers with F-property have the same important theoretical properties as the Fisher's quantifier $\sim_{\alpha,Base}$ [10]: The 4ft quantifier has the *F-property* if it satisfies:

1. If $\approx (a,b,c,d) = 1$ and $b \geq c-1 \geq 0$ then $\approx (a,b+1,c-1,d) = 1$.
2. If $\approx (a,b,c,d) = 1$ and $c \geq b-1 \geq 0$ then $\approx (a,b-1,c+1,d) = 1$.

The proof that the Fisher's quantifier has the F-property is based on consideration published in [6] and it is given in [7].

There are also interesting results concerning pure double implicational, strong double implicational, and typical Σ-double implicational quantifiers defined in [11]:

- The 4ft quantifier $\Leftrightarrow^{*}$ is *pure double implicational* if there is an implicational 4ft quantifier $\Rightarrow^{*}$ such that

$$\Leftrightarrow^{*} (a,b,c) = 1 \text{ if and only if } \Rightarrow^{*} (a,b) \wedge \Rightarrow^{*} (a,c)$$

- The 4ft quantifier $\Leftrightarrow^{*}$ is *strong double implicational* if there are two implicational 4ft quantifier $\Rightarrow^{*}_{1}$ and $\Rightarrow^{*}_{2}$ such that

$$\Leftrightarrow^{*} (a,b,c) = 1 \text{ if and only if } \Rightarrow^{*}_{1} (a,b) \wedge \Rightarrow^{*}_{2} (a,c)$$

- The Σ-double implicational 4ft quantifier $\Leftrightarrow^{*}$ is *typical Σ-double implicational* if there is an integer A such that $1 < Tb_{\Leftrightarrow^{*}}(A) < \infty$

The following facts are among other proved in [11] (the proofs are not too much complicated but they are out of the scope of this chapter):

- The 4ft quantifiers $\Rightarrow_{p,Base}$ of founded implication and $\Rightarrow^{!}_{p,\alpha,Base}$ of lower critical implication are typical Σ-double implicational for $0 < p < 1$ and for $0 < \alpha \leq 0.5$.

- The 4ft quantifiers $\Leftrightarrow^*$ is pure double implicational if and only if there is a non-negative non-decreasing unary function $T_{\Leftrightarrow^*}$ assigning to each natural n a value $T_{\Leftrightarrow^*}(a) \in \mathcal{N}^+$ such that

$$\Leftrightarrow^* (a,b,c) = 1 \text{ if and only if } b < T_{\Leftrightarrow^*}(a) \ \wedge \ c < T_{\Leftrightarrow^*}(a).$$

- The Σ-double implicational quantifier $\Leftrightarrow^*$ is pure double implicational if and only if it is not typical Σ-double implicational.

There are analogous definitions of pure equivalency, strong double implicational, and typical Σ-equivalency quantifiers in [11] and there are also analogous results.

4 Tables of Critical Frequencies

Evaluation of association rules with some 4ft-quantifiers corresponding to statistical hypothesis tests is related to complex computation, e.g.:

- The 4ft-quantifier $\Rightarrow^!_{p,\alpha,Base}$ of lower critical implication is related to the condition $\sum_{i=a}^{a+b} \binom{a+b}{i} p^i (1-p)^{a+b-i} \leq \alpha \wedge a \geq Base$
- The 4ft-quantifier $\Leftrightarrow^!_{p,\alpha,Base}$ of lower critical double implication is related to the condition $\sum_{i=a}^{a+b+c} \binom{a+b+c}{i} p^i (1-p)^{a+b+c-i} \leq \alpha \ \wedge \ a \geq Base$
- The 4ft-quantifier $\equiv^!_{p,\alpha,Base}$ of lower critical equivalence is related to the condition $\sum_{i=a+d}^{n} \binom{n}{i} p^i (1-p)^n \leq \alpha \wedge \ a \geq Base$
- The Fisher's quantifier $\sim_{\alpha,Base}$ is related to the condition $\sum_{i=a}^{\min(r,k)} \frac{\binom{k}{i}\binom{n-k}{r-i}}{\binom{r}{n}} \leq \alpha \ \wedge \ ad > bc \ \wedge \ a \geq Base$

Remark that there are additional 4ft quantifiers requiring complex computation [2, 4]. We show that such computation can be avoided by tables of critical frequencies. Moreover we show that the tables of critical frequencies can be used in a reasonable way to study additional properties of association rules.

We show tables of critical frequencies for three similarly defined classes of association rules:

- Implicational defined by $TPC_{\Rightarrow} : a' \geq a \ \wedge \ b' \leq b$
- Σ-double implicational defined by $TPC_{\Sigma,\Leftrightarrow} : a' \geq a \ \wedge \ b' + c' \leq b + c$
- Σ-equivalency defined by $TPC_{\Sigma,\equiv} : a' + d' \geq a + d \ \wedge \ b' + c' \leq b + c$

Further we will denote $\mathcal{N}^+ = \{0, 1, 2, \dots\} \cup \{\infty\}$.

The table of critical frequencies for implicational quantifier $\Rightarrow^*$ is defined as a *table of maximal b for* $\Rightarrow^*$. It is the function $Tb_{\Rightarrow^*}$ that assigns a value $Tb_{\Rightarrow^*}(a) \in \mathcal{N}^+$ to each $a \geq 0$ such that

$$Tb_{\Rightarrow^*}(a) = \min\{e| \Rightarrow^* (a,e) = 0\}.$$

It is easy to prove that the function $Tb_{\Rightarrow^*}$ is a non-negative and non-decreasing function that satisfies

$$\Rightarrow^* (a,b) = 1 \text{ if and only if } b < Tb_{\Rightarrow^*}(a)$$

for all integers $a \geq 0$ and $b \geq 0$. It means e.g. that

$$\sum_{i=a}^{a+b} \binom{a+b}{i} p^i (1-p)^{a+b-i} \leq \alpha \wedge a \geq Base \text{ if and only if } b < Tb_{\Rightarrow^!_{p,\alpha,Base}}(a)$$

where $Tb_{\Rightarrow^!_{p,\alpha,Base}}$ is a table of maximal b for implicational quantifier $\Rightarrow^!_{p,\alpha,Base}$ of lower critical implication.

We define analogously tables of critical frequencies for Σ-double implicational quantifiers and for Σ-equivalency quantifiers. The table of critical frequencies for Σ-double implicational quantifier $\Leftrightarrow^*$ is defined as a *table of maximal b+c for* $\Leftrightarrow^*$. It is the function $Tb_{\Leftrightarrow^*}$ assigning a value $Tb_{\Leftrightarrow^*}(a) \in \mathcal{N}^+$ to each $a \geq 0$ such that

$$Tb_{\Leftrightarrow^*}(a) = \min\{b+c | \Leftrightarrow^* (a,b,c) = 0\}.$$

It is easy to prove that the function $Tb_{\Leftrightarrow^*}$ is a non-negative and non-decreasing function that satisfies

$$\Leftrightarrow^* (a,b,c) = 1 \text{ if and only if } b + c < Tb_{\Leftrightarrow^*}(a)$$

for all integers $a \geq 0$, $b \geq 0$, and $c \geq 0$.

The table of critical frequencies for Σ-equivalency quantifier $\equiv^*$ is defined as a *table of maximal b+c for* $\equiv^*$. It is the function $Tb_{\equiv^*}$ that assigns a value $Tb_{\equiv^*}(E) \in \mathcal{N}^+$ to each $E \geq 0$ such that

$$Tb_{\equiv^*}(E) = \min\{b+c | \equiv^* (a,b,c,d) = 0 \ \wedge a + d = E.$$

It is easy to prove that the function $Tb_{\equiv^*}$ is a non-negative and non-decreasing function that satisfies

$$\equiv^* (a,b,c,d) = 1 \text{ if and only if } b + c < Tb_{\equiv^*}(a+d)$$

for all integers $a \geq 0$, $b \geq 0$, $c \geq 0$, and $d \geq 0$.

The tables of critical frequencies are practically useful, they can be used to avoid complex computation in the above outlined way. However these tables also describe the behavior of 4ft quantifiers in a very close way. It means that we can use the tables of critical frequencies to study the properties of 4ft quantifiers. Simple property of the table of maximal b for implicational quantifier can be used to decide if the corresponding association rule is definable in the classical predicate calculus or not, see Sect. 7.

We have shown tables of critical frequencies for implicational, Σ-double implicational and Σ-equivalency quantifiers. It is shown in [7] that it is possible to define also a reasonable table of critical frequencies for symmetrical quantifiers with property F. This table can be used to avoid complex computation related to Fisher's quantifier. We will not explain it here in details.

5 Logical Calculi of Association Rules

Logical calculi formulae of which correspond to association rules are defined in [12] according to the following principles:

- The association rule is the expression $\varphi \approx \psi$ where the Boolean attributes φ and ψ are built from the basic Boolean attributes by usual Boolean connectives.
- The basic Boolean attribute is the expression of the form $A(\alpha)$. Here α is the subset of the set of all possible values of the attribute A, see Sect. 2.
- The expression

$$A(a_1) \wedge B(b_3, b_4) \approx C(c_1, c_{11}, c_{21}) \vee D(d_9, d_{10}, d_{11})$$

  is an example of the association rule. Here A and B are the names of the attributes and a_1, b_3, b_4 are the names of its possible values, analogously for C and D, see also Sect. 2. Each attribute has a finite number of possible values called categories.
- The set of all association rules related to the given data matrix is given by:
  - The set of attributes and by the sets of possible values for each attribute
  - The set of all 4ft quantifiers
  - Usual Boolean connectives
- The association rules are interpreted and evaluated in corresponding data matrices. The corresponding data matrix has one column for each attribute and only possible values of the attribute can occur in the corresponding column.
- The evaluation function `Val` with values 1 (truth) and 0 (false) is defined. The value

$$\texttt{Val}(\varphi \approx \psi, \mathcal{M})$$

  of the association rule $\varphi \approx \psi$ in the data matrix $\mathcal{M}$ is done by the four-fold table $4ft(\varphi,\psi, \mathcal{M})$ of φ and ψ in $\mathcal{M}$ and by the condition associated to the 4ft-quantifier $\approx$.

The above described calculi of association rules are special case of observational calculi defined and studied in [2] (main principles are summarized also in [12]). Important observational calculi are defined in [2] by modifications of predicate calculi. The modification consists in

1. Allowing only finite models that correspond to analysed data in the form of {0,1} – data matrices.
2. Adding generalized quantifiers that make possible to express general relations of two or more derived predicates.

The resulting calculi are called *observational predicate calculi* (OPC for short). The 4ft-quantifier $\approx$ is a special case of the generalized quantifier. An example of open formula of observational predicate calculus is the formula

$$(\Rightarrow_{p,B} x)(P_1(x) \wedge P_4(y), P_2(x) \wedge P_3(y))$$

and the expression

$$(\Rightarrow_{p,B} x)(P_1(x) \wedge P_4(x), P_2(x) \wedge P_3(x))$$

is an example of a closed formula. The values of formulas of OPC are defined in Tarski style, see [2].

The formula $(\Rightarrow_{p,B} x)(P_1(x) \wedge P_4(x), P_2(x) \wedge P_3(x))$ corresponds to the association rule

$$P_1 \wedge P_4 \Rightarrow_{p,B} P_2 \wedge P_3$$

defined on {0,1} – data matrix with Boolean columns (i.e. monadic predicates) $P_1, P_2, P_3, P_4, \ldots$. It means that the association rules with Boolean attributes can be understood as formulas of observational predicates calculi. The association rule with categorial attributes defined in Sect. 2 can be also seen as an association rule defined on {0,1} – data matrix with Boolean columns corresponding to monadic predicates, see below.

The attribute A with categories $\{a_1, \ldots a_k\}$ can be represented by k Boolean attributes $A(a_1), \ldots A(a_k)$. Remember that the Boolean attribute $A(a_1)$ is true for the object o if and only if the value of A for the object o is a_1. Thus the basic Boolean attribute $A(a_1, a_2)$ corresponds to the disjunction $A(a_1) \vee A(a_2)$ etc. It means that the rule

$$A(a_1, a_2) \wedge B(b_3) \Rightarrow_{p,B} C(c_4, c_5)$$

can be seen as

$$(A(a_1) \vee A(a_2)) \wedge B(b_3) \Rightarrow_{p,B} C(c_4) \vee C(c_5)$$

and also as

$$(\Rightarrow_{p,B} x)((A(a_1)(x) \vee A(a_2)(x)) \wedge B(b_3)(x), C(c_4)(x) \vee C(c_5)(x))$$

It means that the association rules we deal with are formulas we can get from formulas of classical monadic predicate calculus by adding 4ft-quantifiers. Thus a natural question arises what association rules can be expressed by "classical" quantifiers $\forall$, $\exists$. It is shown that the answer is related to the classes of association rules, see Sect. 7.

6 Deduction Rules

Deduction rules concerning association rules are not only theoretically interesting but also practically important. The deduction rules of the form

$$\frac{\varphi \approx \psi}{\varphi' \approx \psi'}$$

where $\varphi \approx \psi$ and $\varphi' \approx \psi'$ are association rules can be used at least in the following ways:

- *To decrease the number of actually tested association rules:* If the association rule $\varphi \approx \psi$ is true in the analysed data matrix and if $\frac{\varphi \approx \psi}{\varphi' \approx \psi'}$ is the correct deduction rule, then it is not necessary to test $\varphi' \approx \psi'$.
- *To reduce output of a data mining procedure:* If the rule $\varphi \approx \psi$ is included in a data mining procedure output (thus it is true in an analysed data matrix) and if $\frac{\varphi \approx \psi}{\varphi' \approx \psi'}$ is the correct deduction rule then it is not necessary to put the association rule $\varphi' \approx \psi'$ into the output. The used deduction rule must be transparent enough from the point of view of the user of the data mining procedure. An example of a transparent deduction rule is a dereduction deduction rule $\frac{\varphi \Rightarrow^* \psi}{\varphi \Rightarrow^* \psi \vee \chi}$, that is correct for each implicational quantifier $\Rightarrow^*$ [2].

Important examples of deduction rules of the form $\frac{\varphi \approx \psi}{\varphi' \approx \psi'}$ and their interesting properties are presented in [2, 8, 12]. It is shown in [12] that there are transparent criteria of correctness of such deduction rules. These criteria depend on the class of 4ft-quantifier $\approx$. The criteria are informally introduced in this section.

The criteria use the notion saying that the Boolean attribute ψ logically follows from the Boolean attribute φ. Symbolically we write

$$\varphi \vdash \psi.$$

It is $\varphi \vdash \psi$ if for each row o of each data matrix it is true: *If φ is true in o then also ψ is true in o.* It is shown in [12] that there is a formula $\Omega(\varphi, \psi)$ of propositional calculus such that $\varphi \vdash \psi$ if and only if $\Omega(\varphi, \psi)$ is a tautology. The formula $\Omega(\varphi, \psi)$ is derived from φ and ψ by a given way.

The criteria of correctness of deduction rules of the form $\frac{\varphi \approx \psi}{\varphi' \approx \psi'}$ for implication quantifiers concern *interesting implication quantifiers.* The implicational quantifier $\Rightarrow^*$ is interesting if it satisfies:

- $\Rightarrow^*$ is a-dependent
- $\Rightarrow^*$ is b-dependent
- $\Rightarrow^* (0, 0) = 0$

The 4ft quantifier $\approx$ is *a-dependent* if there are non-negative integers a, a', b, c, d such that

$$\approx (a, b, c, d) \neq \approx (a', b, c, d)$$

and analogously for b-dependent 4ft-quantifier. The following theorem is proved in [12]:

If $\Rightarrow^*$ is the interesting implicational quantifier then the deduction rule

$$\frac{\varphi \Rightarrow^* \psi}{\varphi' \Rightarrow^* \psi'}$$

is correct if and only if at least one of the conditions 1 or 2 are satisfied:

1. Both 1.a and 1.b are satisfied
   1.a $\varphi \wedge \psi \vdash \varphi' \wedge \psi'$.
   1.b $\varphi' \wedge \neg\psi' \vdash \varphi \wedge \neg\psi$.
2. $\varphi \vdash \neg\psi$.

It is proved in [7] that the important implicational quantifiers (e.g $\Rightarrow_{p,Base}$ of founded implication and $\Rightarrow^{!}_{p,\alpha,Base}$ of lower critical implication) are interesting implicational quantifiers. The similar theorems are proved for Σ-double implicational quantifier and for Σ-equivalence quantifier in [11], they are presented also in [12].

If $\Leftrightarrow^*$ is the interesting Σ-double implicational quantifier then the deduction rule

$$\frac{\varphi \Leftrightarrow^* \psi}{\varphi' \Leftrightarrow^* \psi'}$$

is correct if and only if at least one of the conditions 1 or 2 are satisfied:

1. Both $(\varphi \wedge \psi) \vdash (\varphi' \wedge \psi')$ and $(\varphi' \wedge \neg\psi') \vee (\neg\varphi' \wedge \psi') \vdash (\varphi \wedge \neg\psi) \vee (\neg\varphi \wedge \psi)$
2. $\varphi \vdash \neg\psi$ or $\psi \vdash \neg\varphi$

The Σ- double implicational quantifier $\Leftrightarrow^*$ is interesting if it is a-dependent, $(b + c)$-dependent and if it is also $\Leftrightarrow^* (0, 0, 0) = 0$. The 4ft-quantifier $\approx$ is $(b + c)$-*dependent* if there are non-negative integers a, b, c, d, b', c' such that

$$b + c \neq b' + c' \text{ and } \approx (a, b, c, d) \neq \approx (a, b', c', d).$$

It is proved in [11] that the important Σ- double implicational quantifiers (e.g $\Leftrightarrow_{p,Base}$ of founded double implication and $\Leftrightarrow^{!}_{p,\alpha,Base}$ of lower critical double implication) are interesting Σ- double implicational quantifiers.

If $\equiv^*$ is the interesting Σ-equivalence quantifier, then deduction rule

$$\frac{\varphi \equiv^* \psi}{\varphi' \equiv^* \psi'}$$

is correct if and only if $(\varphi \wedge \psi \vee \neg\varphi \wedge \neg\psi) \vdash (\varphi' \wedge \psi' \vee \neg\varphi' \wedge \neg\psi')$.

The Σ-equivalence quantifier $\equiv^*$ is interesting if it is $(a + d)$-dependent and if $\equiv^* (0, b, c, 0) = 0$ for $b + c > 0$. The definition of the fact that the 4ft-quantifier $\approx$ is $(a + d)$-dependent is analogous to the definition that it is $(b + c)$-dependent. It is proved in [11] that the important Σ – equivalence quantifiers (e.g $\equiv_{p,Base}$ of founded equivalence and $\equiv^{!}_{p,\alpha,Base}$ of lower critical equivalence are interesting Σ – equivalence quantifiers.

7 Definability in Classical Predicate Calculi

We have shown in Sect. 5 that the association rules we deal with are formulas we can get from formulas of classical monadic predicate calculus by adding 4ft-quantifiers. Thus a natural question arises what association rules can be

expressed by "classical" quantifiers $\forall$, $\exists$. The association rule that is possible to express equivalently by means of classical predicate calculus is called classically definable.

The problem of definability in general monadic observational predicate calculi with equality is solved by the Tharp's theorem, see [2] (and [18] cited in [2]). This theorem gives a criterion of definability of association rules in classical monadic predicate calculus with equality. Tharp's theorem is but too general from the point of view of association rules and it is neither intuitive nor transparent.

The goal of this section is to informally point out that there is an intuitive and transparent criterion of definability that is related to classes of association rules. Formal definitions and theorems are given in [13,14]. We are going to present results concerning implicational quantifiers, there are similar results for equivalency quantifiers. We say that the association rule $\varphi \approx \psi$ is classically definable if the 4ft quantifier $\approx$ is classically definable and vice versa.

We need a notion of the step in the table of maximal b of introduced in Sect. 4. Remember that the table of maximal b for implicational quantifier $\Rightarrow^*$ is the non-negative and non-decreasing function $Tb_{\Rightarrow^*}$ that assigns a value $Tb_{\Rightarrow^*}(a) \in \mathcal{N}^+$ to each $a \geq 0$ such that $Tb_{\Rightarrow^*}(a) = \min\{e|\ \Rightarrow^*(a,e) = 0\}$. It is $\Rightarrow^*(a,b) = 1$ if and only if $b < Tb_{\Rightarrow^*}(a)$ for all integers $a \geq 0$ and $b \geq 0$. A *step in the table* $Tb_{\Rightarrow^*}$ *of maximal* b is each such $a \geq 0$ for which it is $Tb_{\Rightarrow^*}(a) < Tb_{\Rightarrow^*}(a+1)$.

The simple criterion of definability of association rules in classical monadic predicate calculus with equality says that the implicational quantifier $\Rightarrow^*$ is classically definable if and only if its table of maximal b has only finite number of steps [14].

To illustrate how it works we use the implicational quantifier $\rightarrow^*$ defined such that its table of maximal b is given by Table 3. It means: $\rightarrow^*(0,0) = 0$, $\rightarrow^*(0,1) = 0, \ldots, \rightarrow^*(1,0) = 0, \ldots, \rightarrow^*(2,0) = 0, \ldots, \rightarrow^*(3,0) = 1, \rightarrow^*(3,1) = 1, \rightarrow^*(3,2) = 0, \ldots, \ldots, \rightarrow^*(6,0) = 1, \rightarrow^*(6,1) = 1, \rightarrow^*(6,2) = 1, \rightarrow^*(6,3) = 1, \rightarrow^*(6,4) = 0, \ldots$, etc. In other words, it is $\rightarrow^*(a,b) = 1$ if and only if

$$\langle a,b\rangle \in \langle 3,5\rangle \times \langle 0,1\rangle \quad or \quad \langle a,b\rangle \in \langle 6,\infty) \times \langle 0,3\rangle\ .$$

Remark that the table of maximal b $Tb_{\rightarrow^*}$ of the quantifier $\rightarrow^*$ has two steps: 2 (because of $Tb_{\rightarrow^*}(2) < Tb_{\rightarrow^*}(3)$) and 5 (because of $Tb_{\rightarrow^*}(5) < Tb_{\rightarrow^*}(6)$).

Table 3. Table of maximal b $Tb_{\rightarrow^*}$ of the quantifier $\rightarrow^*$

frequency a	Value of $Tb_{\rightarrow^*}(a)$
0–2	0
3–5	2
≥ 6	4

Table 4. 4ft table 4ft(P_1,P_2, $\mathcal{M}$)

$\mathcal{M}$	P_2	$\neg P_2$	
P_1	a	b	r
$\neg P_1$	c	d	
	k		n

To study the definability of the quantifier $\rightarrow^*$ we will observe the association rule $P_1 \rightarrow^* P_2$ that we will understood as the formula

$$(\rightarrow^*, x)(P_1(x), P_2(x))$$

of an observational predicate calculus (see Sect. 5).

To show that this formula is definable in classical predicate calculus we have to find a formula Φ consisting of some of symbols: predicates P_1, P_2, logical connectives $\wedge, \vee, \neg$, classical quantifiers $\forall$, $\exists$, equality $=$ (and of course inequality $\neq$) and variables x_1, x_2 ... that is logically equivalent to $(\rightarrow^*, x)(P_1(x), P_2(x)$.

The fact that the formula Φ is logically equivalent to $(\rightarrow^*, x)(P_1(x), P_2(x)$ means that Φ is true in data matrix $\mathcal{M}$ if and only if

$$\langle a, b\rangle \in \langle 3, 5\rangle \times \langle 0, 1\rangle \quad or \quad \langle a, b\rangle \in \langle 6, \infty) \times \langle 0, 3\rangle$$

where frequencies a, b are given by 4ft table 4ft(P_1,P_2, $\mathcal{M}$) see Table 4. We will construct Φ such that $\Phi = \Phi_1 \vee \Phi_2$, Φ_1 is equivalent to $\langle a, b\rangle \in \langle 3, 5\rangle \times \langle 0, 1\rangle$ and Φ_2 is equivalent to $\langle a, b\rangle \in \langle 6, \infty) \times \langle 0, 3\rangle$. We will use the formulas

$$\kappa_a(x) = P_1(x) \wedge P_2(x) \quad \text{and} \quad \kappa_b(x) = P_1(x) \wedge \neg P_2(x)$$

and the quantifiers $\exists^k$ where k is a natural number. The quantifier $\exists^k$ says "there are at least k mutually different objects". It is defined using the classical quantifier $\exists$ and the predicate of equality. An example of its application is the formula $\exists^3 \kappa_a(x)$ saying "there are at least three mutually different objects satisfying" $\kappa_a(x)$ that is defined this way:

$$(\exists^3 x)\kappa_a(x) \;=\; (\exists x_1 \exists x_2 \exists x_3)\kappa_a(x) \;\wedge\; (x_1 \neq x_2 \wedge x_1 \neq x_3 \wedge x_2 \neq x_3).$$

The formula Φ_1 equivalent to $\langle a, b\rangle \in \langle 3, 5\rangle \times \langle 0, 1\rangle$ can be defined as

$$\Phi_1 \;=\; (\exists^3 x)\kappa_a(x) \;\wedge\; \neg((\exists^6 x)\kappa_a(x)) \wedge \neg((\exists^2 \kappa_b(x))$$

The formula Φ_2 equivalent to $\langle a, b\rangle \in \langle 6, \infty) \times \langle 0, 3\rangle$ can be defined as

$$\Phi_2 \;=\; (\exists^6 x)\kappa_a(x) \;\wedge \neg((\exists^4 \kappa_b(x)).$$

The formula $\Phi = \Phi_1 \vee \Phi_2$ defined this way consists of symbols: predicates P_1, P_2, logical connectives $\wedge, \vee, \neg$, classical quantifier $\exists$, inequality $\neq$ and of suitable variables and it is logically equivalent to $(\rightarrow^*, x)(P_1(x), P_2(x)$. It shows that the quantifier $\rightarrow^*$ is classically definable. We have also seen that the formula Φ is constructed on the basis of the table of maximal b $Tb_{\Rightarrow^*}$ of $\Rightarrow^*$. The table $Tb_{\Rightarrow^*}$ has two steps that are used in the construction of Φ.

8 Missing Information

Missing information is a common problem in data mining. One of possibilities how to deal with missing information is secured X-extension introduced in [2]. It deals with data matrices with missing information. We assume that there is a special symbol X that we interpret as the fact "the value of the corresponding attribute is not known for the corresponding object". An example of data matrix $\mathcal{M}^X$ with missing information is in Fig. 2.

The principle of secured X-extension is to extend the set $\{0,1\}$ of values of Boolean attributes and values of association rules to the set $\{0,1,X\}$ such that the below given conditions are satisfied.

We denote the value of Boolean attribute φ in row o of data matrix $\mathcal{M}$ as $\varphi(o,\mathcal{M})$. It can be $\varphi(o,\mathcal{M})=1$ (i.e. φ is true in row o of $\mathcal{M}$) or $\varphi(o,\mathcal{M})=0$ (i.e. φ is false in row o of $\mathcal{M}$). If we have data matrix $\mathcal{M}^X$ with missing information then it can be $\varphi(o,\mathcal{M}^X)=1$, $\varphi(o,\mathcal{M}^X)=0$ or $\varphi(o,\mathcal{M}^X)=X$.

The secured X-extension deals with completions of data matrix $\mathcal{M}^X$ with missing information. The *completion of the data matrix* $\mathcal{M}^X$ *with missing information* is each data matrix $\mathcal{M}$ with the same rows and columns such that each symbol X is replaced by one of possible values of the corresponding attribute (i.e. the column of $\mathcal{M}$).

The principle of *secured X-extension for Boolean attributes* means that values of each Boolean attribute φ in data matrix $\mathcal{M}^X$ with missing information are defined such that

$$\varphi(o,\mathcal{M}^X)=\begin{cases}\xi\in\{0,1\} \text{ if } \varphi(o,\mathcal{M})=\xi \text{ in each completion } \mathcal{M} \text{ of } \mathcal{M}^X\\ \\ X \qquad\qquad \text{otherwise.}\end{cases}$$

The value $A(\alpha)(o,\mathcal{M}^X)$ of basic Boolean attribute $A(\alpha)$ in row o of data matrix $\mathcal{M}^X$ is according to the principle of secured X-extension defined such that

- $A(\alpha)(o,\mathcal{M}^X)=1$ if $A(o,\mathcal{M}^X)\in\alpha$
- $A(\alpha)(o,\mathcal{M}^X)=0$ if $A(o,\mathcal{M}^X)\notin\alpha\wedge A(o,\mathcal{M}^X)\neq X$
- $A(\alpha)(o,\mathcal{M}^X)=X$ otherwise

Here $A(o,\mathcal{M}^X)$ is the value of attribute A in row o of data matrix $\mathcal{M}^X$. In Fig. 2 there are some examples of values of basic Boolean attributes in data matrix with missing information.

object	A	B	C	D	...	Z	$A(a_1)$	$B(b_3,b_4)$
o_1	a_1	b_8	c_{16}	X	...	z_{14}	1	0
o_2	a_5	X	c_7	d_2	...	X	0	X
⋮	⋮	⋮	⋮	⋮	⋱	⋮	⋮	⋮
o_n	X	b_3	c_4	d_1	...	z_6	X	1

Fig. 2. An example of data matrix $\mathcal{M}^X$

	$\neg$
1	0
X	X
0	1

$\wedge$	1	X	0
1	1	X	0
X	X	X	0
0	0	0	0

$\vee$	1	X	0
1	1	1	1
X	1	X	X
0	1	X	0

Fig. 3. Extended truth tables of $\vee$, $\wedge$ and $\neg$

Table 5. Ninefold table $9ft(\varphi, \psi, \mathcal{M}^X)$

$\mathcal{M}^X$	ψ	ψ_X	$\neg\psi$
φ	$f_{1,1}$	$f_{1,X}$	$f_{1,0}$
φ_X	$f_{X,1}$	$f_{X,X}$	$f_{X,0}$
$\neg\varphi$	$f_{0,1}$	$f_{0,X}$	$f_{0,0}$

The values of Boolean attributes derived from basic Boolean attributes using propositional connectives $\vee$, $\wedge$ and $\neg$ are defined by truth tables of these connectives extended by the principle of secured X-extension [2] see Fig. 3.

The principle of *secured X-extension for association rules* means that values of association rule $\varphi \approx \psi$ in data matrix $\mathcal{M}^X$ with missing information are defined such that

$$Val(\varphi \approx \psi, \mathcal{M}^X) = \begin{cases} \xi \in \{0,1\} \text{ if } Val(\varphi \approx \psi, \mathcal{M}) = \xi \\ \qquad\qquad \text{in each completion } \mathcal{M} \text{ of } \mathcal{M}^X \\ X \qquad\quad \text{otherwise.} \end{cases}$$

The evaluation of association rules in data matrices with missing information was studied e.g. in [2, 6, 7, 10]. The results show that we can construct *secured fourfold table* $\langle a_s, b_s, c_s, d_s \rangle$ for a given data matrix $\mathcal{M}^X$ with missing information such that it is

$$Val(\varphi \approx \psi, \mathcal{M}^{\mathcal{X}}) = 1 \text{ iff } \approx (a_s, b_s, c_s, d_s) = 1.$$

The secured fourfold table depends on the class of evaluated association rule. The results concerning implicational, Σ-double implicational and Σ-equivalency classes are rather trivial, however the study of behavior of Fisher's quantifier in data with missing information led to the definition of the class of 4ft quantifiers with F-property, see Sect. 3.4 and [7].

The secured fourfold table is constructed from a ninefold table of φ and ψ in $\mathcal{M}^X$ that is denoted $9ft(\varphi, \psi, \mathcal{M}^X)$, see Table 5.

Here $f_{1,1}$ is the number of rows o of $\mathcal{M}^X$ such that both $\varphi(o, \mathcal{M}^X) = 1$ and $\psi(o, \mathcal{M}^X) = 1$, $f_{1,X}$ is the number of rows o of $\mathcal{M}^X$ such that both $\varphi(o, \mathcal{M}^X) = 1$ and $\psi(o, \mathcal{M}^X) = X$, etc.

The problem is that we have to deal with $2^{f_{1,X}+f_{X,1}+f_{X,0}+f_{0,X}} * 4^{f_{X,X}}$ 4ft tables corresponding to all possible completions of data matrix $\mathcal{M}^X$. If $\mathcal{M}$ is a completion of $\mathcal{M}^X$ and if it is $\langle a, b, c, d \rangle = 4ft(\varphi, \psi, \mathcal{M})$ then it is also

$a = f_{1,1} + f^a_{1,X} + f^a_{X,1} + f^a_{X,X}$
$b = f_{1,0} + f^b_{1,X} + f^b_{X,0} + f^b_{X,X}$
$c = f_{0,1} + f^c_{X,1} + f^c_{0,X} + f^c_{X,X}$
$d = f_{0,0} + f^d_{0,X} + f^d_{X,0} + f^d_{X,X}$.

Here $f^a_{1,X}$ is the number of rows of $\mathcal{M}^X$ such that in $\mathcal{M}^X$ it is $\varphi(o, \mathcal{M}^X) = 1$ and $\psi(o, \mathcal{M}^X) = X$ and after completion it is $\varphi(o, \mathcal{M}) = 1$ and $\psi(o, \mathcal{M}) = 1$, similarly for $f^a_{X,1}$, $f^a_{X,X}$, $f^b_{1,X}$ etc.

The association rule $\varphi \Rightarrow^* \psi$ with an implicational 4ft-quantifier $\Rightarrow^*$ is true in all completions of data matrix $\mathcal{M}^X$ (i.e. $\varphi \Rightarrow^* \psi$ is true in $\mathcal{M}^X$) if and only if it is

$$\Rightarrow^* (f_{1,1},\ f_{1,0} + f_{1,X} + f_{X,0} + f_{X,X}) = 1.$$

It follows from the truth preservation condition $TPC_{\Rightarrow}$ for implicational quantifiers $a' \geq a\ \wedge\ b' \leq b$ and from the fact that for any completion $\langle a, b, c, d\rangle$ of the ninefold table $9ft(\varphi, \psi, \mathcal{M}^X)$ it must be

$$a \geq f_{1,1}\ \wedge\ b \leq f_{0,1} + f_{1,X} + f_{X,0} + f_{X,X}$$

see [2]. It means that secured fourfold table $\langle a_I, b_I, c_I, d_I\rangle$ for data $\mathcal{M}^X$ with missing information and for an implicational 4ft quantifier $\Rightarrow^*$ can be constructed e.g. as

$$\langle a_I, b_I, c_I, d_I\rangle = \langle f_{1,1},\ f_{1,0} + f_{1,X} + f_{X,0} + f_{X,X}, 0, 0\rangle.$$

The analogous results we can get for additional classes of association rules. The association rule $\varphi \Leftrightarrow^* \psi$ with a Σ-double implicational quantifier $\Leftrightarrow^*$ is true in all completions of data matrix $\mathcal{M}^X$ if and only if it is

$$\Leftrightarrow^* (f_{1,1}, f_{1,0} + f_{1,X} + f_{X,0} + f_{X,X},\ f_{0,1} + f_{0,X} + f_{X,1}) = 1.$$

The association rule $\varphi \equiv^* \psi$ with a Σ-equivalency quantifier $\equiv^*$ is true in all completions of data matrix $\mathcal{M}^X$ if and only if it is

$$\equiv^* (f_{1,1}, f_{1,0} + f_{1,X} + f_{X,0} + f_{X,X}, f_{0,1} + f_{0,X} + f_{X,1}, f_{0,0}) = 1.$$

It is shown in [7] (see also [6]) that the secured 4ft table $\langle a_F, b_F, c_F, d_F\rangle$ for the 4ft-quantifiers with the F-property (see Sect. 3) is defined such that

$a_F = f_{1,1}$,
$b_F = f_{1,0} + f_{1,X} + f_{X,0} + F_1$,
$c_F = f_{0,1} + f_{0,X} + f_{X,0} + F_2$,
$d_F = f_{0,0}$

where $F_1 + F_2 = f_{X,X}$ and $|b_F - c_F|$ is minimal. Let us remark that it is proved in [7] that the 4ft quantifier corresponding to Chi-squared test has the F-property.

Acknowledgment

The work described here has been supported by the grant 201/05/0325 of the Czech Science Foundation and by the project IGA 25/05 of University of Economics, Prague.

References

1. R. Aggraval, et al. (1996) Fast Discovery of Association Rules. In: U.M. Fayyad, et al. (eds.) Advances in Knowledge Discovery and Data Mining. AAAI, Menlo Park, CA. 307–328
2. P. Hájek, T. Havránek (1978) Mechanising Hypothesis Formation – Mathematical Foundations for a General Theory. Springer, Berlin Heidelberg New York
3. P. Hájek (guest editor) (1978) International Journal of Man–Machine Studies, special issue on GUHA, vol. 10
4. P. Hájek, T. Havránek, M. Chytil (1983) GUHA Method. Academia, Prague (in Czech)
5. P. Hájek, A. Sochorová, J. Zvárová (1995) GUHA for Personal Computers. Computational Statistics and Data Analysis 19, pp. 149–153
6. J. Rauch (1975) Ein Beitrag zu der GUHA Methode in der dreiwertigen Logik. Kybernetika, vol. 11, pp. 101–113
7. J. Rauch (1986) Logical Foundations of Hypothesis Formation from Databases (in Czech), Mathematical Institute of the Czechoslovak Academy of Sciences, Prague, Czech Republic, Dissertation, 1986
8. J. Rauch (1997) Logical Calculi for Knowledge Discovery in Databases. In: J. Zytkow, J. Komorowski (eds.) Principles of Data Mining and Knowledge Discovery. Springer, Berlin Heidelberg New York. 47–57
9. J. Rauch (1998) Classes of Four-Fold Table Quantifiers. In: J. Zytkow, M. Quafafou (eds.) Principles of Data Mining and Knowledge Discovery. Springer, Berlin Heidelberg New York. 203–211
10. J. Rauch (1998) Four-Fold Table Calculi and Missing Information. In: P. Wang, (ed.) JCIS '98, Association for Intelligent Machinery, Vol. II. Duke University, Durham
11. J. Rauch (1998) Contribution to Logical Foundations of KDD. Assoc. Prof. Thesis, Faculty of Informatics and Statistics, University of Economics, Prague (in Czech)
12. J. Rauch (2005) Logic of Association Rules. Applied Intelligence 22, 9–28
13. J. Rauch (2005) Definability of Association Rules in Predicate Calculus. In: T.Y. Lin, S. Ohsuga, C.J. Liau, X. Hu (eds.) Foundatuons and Novel Approaches in Data Mining. Springer, Berlin Heidelberg New York. 23–40
14. J. Rauch (2005) Definability of Association Rules and Tables of Critical Frequencies. In: Data Mining: Foundations and practice. Springer, Berlin Heidelberg New York
15. J. Rauch, M. Šimůnek (2005) An Alternative Approach to Mining Association Rules. In: T.Y. Lin, S. Ohsuga, C.J. Liau, and S. Tsumoto (eds.) Foundations of Data Mining and Knowledge Discovery. Springer, Berlin Heidelberg New York. 219–238

16. J. Rauch, M. Šimůnek, V. Lín (2005) Mining for Patterns Based on Contingency Tables by KL-Miner – First Experience. In: T.Y. Lin, S. Ohsuga, Liau C J, Hu X (eds.) Foundations and Novel Approaches in Data Mining. Springer, Berlin Heidelberg New York. 155–167
17. J. Rauch, M. Šimůnek (2005) GUHA Method and Granular Computing. In: X. Hu, Q. Liu, A. Skowron, T.Y. Lin, R. Yager, B. Zang (eds.) Proceedings of IEEE conference Granular Computing 2005. IEEE Beijing. pp. 630–635
18. L.H. Tharp (1973) The Characterisation of Monadic Logic. Journal of Symbolic Logic 38, 481–488
19. R. Zembowicz, J. Zytkow (1996) From Contingency Tables to Various Forms of Knowledge in Databases. In: Fayyad UM, et al. (eds.) Advances in Knowledge Discovery and Data Mining. AAAI, Menlo Park, CA. 329–349

Knowledge Extraction from Microarray Datasets Using Combined Multiple Models to Predict Leukemia Types

Gregor Stiglic[1], Nawaz Khan[2], and Peter Kokol[1]

[1] Faculty of Electrical Engineering and Computer Science, University of Maribor, 2000 Maribor, Slovenia
gregor.stiglic@uni-mb.si
[2] School of Computing Science, Middlesex University, The Burrough, Hendon, London NW4 4BT, UK
n.x.khan@mdx.ac.uk

Summary. Recent advances in microarray technology offer the ability to measure expression levels of thousands of genes simultaneously. Analysis of such data helps us identifying different clinical outcomes that are caused by expression of a few predictive genes. This chapter not only aims to select key predictive features for leukemia expression, but also demonstrates the rules that classify differentially expressed leukemia genes. The feature extraction and classification are carried out with combination of the high accuracy of ensemble based algorithms, and comprehensibility of a single decision tree. These allow deriving exact rules by describing gene expression differences among significantly expressed genes in leukemia. It is evident from our results that it is possible to achieve better accuracy in classifying leukemia without sacrificing the level of comprehensibility.

1 Introduction

Clinical diagnosis for disease prediction is one of the most important emerging applications of microarray gene expression study. In the last decade, a new technology, DNA microarrays, has allowed screening of biological samples for a huge number of genes by measuring expression patterns. This technology enables the monitoring of the expression levels of a large portion of a genome on a single slide or "chip", thus allowing the study of interactions among thousands of genes simultaneously [1]. Usually microarray datasets are used for identification of differentially expressed genes, which from data mining point of view represents a feature selection problem. The objectives of this research are to select important features from leukemia predictive genes and to derive a set of rules that classify differentially expressed genes. The study

G. Stiglic et al.: *Knowledge Extraction from Microarray Datasets Using Combined Multiple Models to Predict Leukemia Types*, Studies in Computational Intelligence (SCI) **118**, 339–352 (2008)
www.springerlink.com

follows the comprehensibility of a single decision tree. Although, there are many research that have demonstrated a higher level of accuracy in classifying cancer cells, for example [2, 3], the comprehensibility issue of decision trees to gain best accuracy in the domain of microarray data analysis has been ignored [4–7].

In this study, we attempt to combine the high accuracy of ensembles and the interpretability of the single tree in order to derive exact rules that describe differences between significantly expressed genes that are responsible for leukemia. To achieve this, Combined Multiple Models (CMM) method has been applied, which was proposed originally by Domingos in [8]. In our study the method is adapted for multidimensional and real valued microarray datasets to eliminate the colinearity and multivariate problems. All datasets from our experiment are publicly available from the Kent Ridge Repository described in [20]. These microarray samples are the examples of human tissue extracts that are related to a specific disease and have been used for comprehensible interpretation in this study. The following sections explore the datasets, methods of CMM adaptation and testing. It also presents the results that are obtained by applying the adapted method on four publicly available databases. Finally the chapter presents a validation study by providing an interpretation of the results in the context of rule sets and then by comparing the proposed adaptations with the combined and simple decision trees for leukemia grouping.

2 Combined Multiple Models for Gene Expression Analysis

Data mining is the process of autonomously extracting useful information or knowledge from large datasets. Many different models can be used in data mining process. However, it is required for many applications not only to involve model that produce accurate predictions, but also to incorporate comprehensible model. In many applications it is not enough to have accurate model, but we also want comprehensible model that can be easily interpreted to the people not familiar with data mining. For example, Tibshirani and Knight [9] proposed a method called Bumping that tries to use bagging and produce a single classifier that best describes the decisions of bagged ensemble. It builds models from bootstrapped samples and keeps the one with the lowest error rate on the original data. Typically this is enough to get good results also on test set. We should also mention papers that suggest different techniques of extracting decision trees from neural networks or ensembles of neural networks that can all be seen as a "black-box" method [10–12].

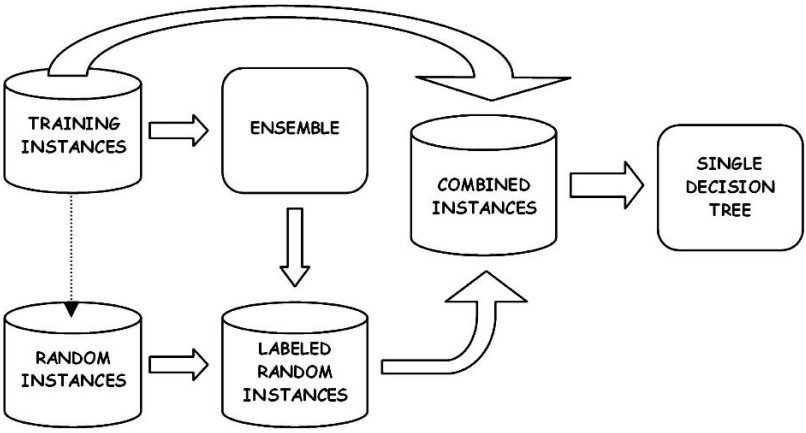


Fig. 1. Building decision tree from ensemble using CMM

2.1 CMM Approach

One of the methods that are able to build comprehensible model from an ensemble of models is called Combined Multiple Models (CMM) and was presented in [8]. CMM was later studied and improved by Estruch et al. in [13]. Basic idea of CMM is to build a single classifier that would retain most of the accuracy gains of the ensemble models. This is done by adding additional "artificial data points" to the learning dataset. Those additional data points are then classified (i.e. labeled) by applying the ensemble of classifiers that was trained on the learning dataset. The next step is joining the original training dataset with the new "artificial" dataset. This final dataset is used to build a single comprehensible classifier. The whole process is shown in Fig. 1. The idea of generating the "artificial data points" when building classifiers, was already used in several papers. One of the first such methods is the active learning method proposed by Cohn et al. in [14]. Another application of artificial examples was presented by Craven and Shavlik in [15] where they describe the learning of decision trees from neural networks. This approach was later used in several papers on neural networks knowledge extraction.

2.2 Proposed Modifications

This chapter presents the CMM based method that is specialized for microarray dataset classification problems. Optimization of the original method was done on artificial data points creation due to specific structure of the microarray datasets. Opposite to the original research [8], where most of the best results were achieved on the datasets containing nominal values, microarray analysis presents pure continuous-valued data-sets. Therefore a new method called Combined Multiple Models for Continuous values (CMMC) is proposed. In this chapter three new artificial data points creation techniques

for decision tree building are examined which will be referred to as CMMC-1, CMMC-2 and CMMC-3. Both methods are based on multiplication of the original training set instances. Data points are generated from original training set by creating copies of original training set instances by slightly changing the values of attributes. First method is based on the variance of the gene expression values and each attribute can be changed by adding the random value from one of the intervals $\left\{-\sigma, -\frac{1}{3}\sigma\right\}$ and $\left\{\frac{1}{3}\sigma, \sigma\right\}$ to the original gene expression value. Because of the large number of attributes we change only 50% randomly selected attributes. Result of such data point multiplication is a wide dispersion of the points around their base data point, but original training set distribution of the samples is still preserved. Second method tries to maintain the original distribution on even tighter area than the first one, especially when data points lie tightly together. This is done by generating the random points in the interval $x \quad d$, where x is the value of the attribute and d is the distance to the nearest neighbour value of this attribute. Again only 50% of attributes are randomly selected for modification. Another modification of the original approach was done in application of different ensemble building method. Based on our own tests and also reports in some papers [16], we decided to use Random Forest ensemble building method that is based on one of the first ensemble building methods called bagging [17]. To compose ensemble from base classifiers using bagging, each classifier is trained on a set of n training examples, drawn randomly with replacement from the original training set of size m. Such subset of examples is also called a bootstrap replicate of the original set. Breiman upgraded the idea of bagging by combining it with the random feature selection for decision trees. This way he created Random Forests, where each member of the ensemble is trained on a bootstrap replicate as in bagging. Decision trees are than grown by selecting the feature to split on at each node from randomly selected number of nodes. We set number of chosen features to $log_2(k+1)$ as in [18], where k is the total number of features. Random Forests are the ensemble method that works well even with noisy content in the training dataset and are considered as one of the most competitive methods that can be compared to boosting [19]. To get the most out of the proposed multiplication of data points another version of CMMC algorithm was derived. This version (CMMC-3) is based upon multiplication of data points in each of the leafs which are later extended by additional subtree. Since our decision trees are pruned they achieve good generalization and are less complex than unpruned decision trees. Therefore we try to "upgrade" each leaf by attaching another subtree under the leaf. These subtrees are built using CMMC technique described above (basically the same as CMMC-2). Thereby existing data points that got to the leaf are multiplied (again by adding 1,000 artificial data points labelled by Random Forest) and the problem of a small number of samples in lower nodes of trees is reduced but not solved as we cannot be certain about the correct labelling of the artificial samples.

3 Experiment and Results

3.1 Datasets

Four widely used publicly available gene expression datasets were used in our experimental evaluation of the proposed method. They were obtained from Kent Ridge Biomedical Data Set Repository which was described in [20].

Leukemia1 Dataset (amlall)

The original data comes from the research on acute leukemia by Golub et al. [21]. Dataset consists of 38 bone marrow samples from which 27 belong to acute lymphoblastic leukemia (ALL) and 11 to acute myeloid leukemia (AML). Each sample consists of probes for 6,817 human genes. Golub used this dataset for training. Another 34 samples of testing data were used consisting of 20 ALL and 14 AML samples. Because we used leave-one-out cross-validation, we were able to make tests on all samples together (72).

Breast Cancer Dataset (Breast)

This dataset was published in [22] and consists of extremely large number of scanned gene expressions. It includes data on 24,481 genes for 78 patients, 34 of which are from patients who had developed distance metastases within 5 years, the rest 44 samples are from patients who remained healthy from the disease after their initial diagnosis for interval of at least 5 years.

Lung Cancer Dataset (Lung)

Lung cancer dataset includes the largest number of samples in our experiment. It includes 12,533 gene expression measurements for each of 181 tissue samples. The initial research was done by Gordon et al. [23] where they try to classify malignant pleural mesothelioma (MPM) and adenocarcinoma (ADCA) of the lung.

Leukemia2 Dataset (mll)

This Leukemia dataset tries to discern between three types of leukemia (ALL, MLL, AML). Dataset contains 72 patient samples, each of them containing 12,582 gene expression measurements. Data was collected by Armstrong et al. and results published in [24].

3.2 Gene Selection

It has been shown that selecting a small subset of informative genes can lead to improved classification accuracy and greatly improves execution time of data mining tools [25]. The most commonly used gene selection approaches are based on gene ranking. In these gene ranking approaches, each gene is evaluated individually and assigned a score representing its correlation with the class. Genes are then ranked by their scores and the top ranked ones are selected from the initial set of features (genes). To make our experiments less dependent of the filtering method, we use three different filtering methods. This way we get 12 different microarray datasets with a pre-defined number of most relevant gene expressions. All used filtering methods are part of WEKA toolkit [26] that we were using in our experiments. The following filtering methods were used:

GainRatio filter. This is the heuristic that was originally used by Quinlan in ID3 [27]. It is implemented in WEKA as a simple and fast feature selection method. The idea of using this feature selection technique for gene ranking was already presented by Ben-Dor et al. [28].

Relief-F filter. The basic idea of Relief-F algorithm [29] is to draw instances at random, compute their nearest neighbors, and adjust a feature weighting vector to give more weight to features that discriminate the instance from neighbors of different classes. A study comparing Relief-F to other similar methods in microarray classification domain was conducted by Wang and Makedon [30] where they conclude that the performance of Relief-F is comparable with other methods.

SVM filter. Ranking is done using Support Vector Machines (SVM) classifier. Similar approach using SVM classifier for gene selection was already used in papers by Guyon et al. [31] and Fujarewicz et al. [32].

3.3 Experiment Setting

The experiments are designed to test the accuracy gain of all three CMMC methods compared to accuracy of a single J48 tree (Java C4.5 tree implementation in WEKA toolkit). The study has followed n-fold cross-validation process for testing. The n-fold cross-validation is typically implemented by running the same learning system n times and each time on a different training set of size $(n\text{-}1)/n$ times the size of the original data set. A specific variation of n-fold cross-validation, called leave-one-out cross-validation method (LOOCV), is used in the experiment. In this approach, one sample in the training set is withheld, the remaining samples of the training set are used to build a classifier to predict the class of withheld sample, and the cumulative error is then calculated. LOOCV was often criticized, because of higher error variance in comparison to five or tenfold cross-validation [33], but a recent study by Braga-Neto and Dougherty [34] shows that LOOCV can be considered very useful for microarray datasets, because they have not been

able to verify the substantial differences in performance among the mentioned methods. Because of random nature in tested classifier building methods, this research attempts to repeat LOOCV ten times for all random based methods (both CMMC and Random Forests) and then computes average accuracy for all runs. As indicated in [8], 1,000 artificial data points are generated for CMMC-1 and CMMC-2 methods, while CMMC-3 uses the same number of artificial data points in every "upgraded" leaf.

3.4 Results

This section highlights the key findings that are obtained by applying the adapted CMM method on four microarray datasets available in public domain. Table 1 shows the accuracy comparison. The tests show that all three proposed methods gained some accuracy comparing to a simple decision tree, but they are still lacking a lot of accuracy compared to ensemble of classifiers.

To keep the complexity level low for built decision trees, we used pruning in all decision trees that are used in the experiment. Average complexity (i.e. number of rules) of decision trees is presented in Table 2. We do not present the complexity of Random Forest Method as it can be simply estimated as approximately 100 times larger than the simple decision tree and therefore completely unacceptable for interpretation. The most significant fact revealed from the Table 2 is low rule complexity of CMMC-2 generated decision trees, especially when compared to CMMC-1 trees. Trees from our second proposed method

Table 1. Comparison of accuracy for decision tree (C4.5), proposed decision tree building methods and Random Forests (RF)

Dataset	C4.5	CMMC1	CMMC2	CMMC3	RF
amlall1	80.56	90.40	89.20	87.58	97.92
amlall2	79.17	91.29	88.70	88.44	98.30
amlall3	79.17	90.85	87.94	88.30	98.80
amlallAvg	79.63	90.85	88.61	88.11	98.34
breast1	66.67	73.89	67.19	72.85	85.84
breast2	61.54	64.74	65.62	65.66	81.00
breast3	71.79	66.49	66.78	65.04	90.21
breastavg	66.67	68.37	66.53	67.97	85.68
lung1	96.13	96.37	97.55	96.64	99.45
lung2	97.79	96.61	97.95	97.06	98.97
lung3	98.90	96.53	98.58	97.28	99.45
lungavg	97.61	96.50	98.03	96.99	99.29
mll1	79.17	88.89	89.48	87.08	97.62
mll2	88.89	88.29	88.89	91.96	94.64
mll3	84.72	88.29	88.89	87.62	96.03
mllAvg	84.26	88.49	89.09	88.87	96.10
Average	82.04	86.05	85.56	85.49	94.85

Table 2. Comparison of tree complexity (number of leafs) for decision tree (C4.5) and proposed decision tree building methods (CMMC-1, CMMC-2 and CMMC-3)

Dataset	C4.5	CMMC1	CMMC2	CMMC3
amlall1	2.93	73.30	5.24	6.19
amlall2	2.93	75.57	5.38	5.45
amlall3	2.93	75.77	5.05	6.29
amlallAvg	2.93	74.88	5.22	5.98
breast1	6.26	46.35	18.78	11.22
breast2	7.12	11.37	15.65	14.53
breast3	6.76	11.92	14.68	23.05
breastAvg	6.71	23.21	16.37	16.27
lung1	3.99	75.32	5.05	4.35
lung2	4.00	64.05	9.21	4.31
lung3	4.00	72.71	5.52	4.21
lungAvg	4.00	70.69	6.59	4.29
mll1	3.00	115.94	6.39	4.88
mll2	3.00	118.31	8.68	5.24
mll3	3.92	121.76	7.60	5.00
mllAvg	3.31	118.67	7.56	5.04
Average	4.24	71.86	8.94	7.90

Table 3. Comparison of accuracy by feature selection method

Feature selection	C4.5	CMMC1	CMMC2	CMMC3	RF
GainRatio	80.63	87.39	85.86	86.04	95.21
ReliefF	81.85	85.23	85.29	85.78	93.23
SVM-FS	83.65	85.54	85.55	84.56	96.12

generate only two times more rules than simple decision trees. Even better results were obtained using CMMC-3 method. Low complexity at CMMC-3 based trees is a consequence of the decision tree building technique in which trees are generated at the beginning using the initial training sets without artificial data points. The artificial data points are added in later stages.

Table 3 presents the results based on average results on each dataset for each gene selection method. The best results, with exception of CMMC-3 method, were achieved when SVM based feature selection method was used. From this table it can also be seen that the majority of accuracy gain of the CMMC-1 method compared to CMMC-2 was due to first method's better accuracy when used with the GainRatio based feature selection method.

3.5 Subgrouping Leukemia Type

To demonstrate the practical advantage of our proposed adaptation to CMM method, two sample trees were constructed from the Leukemia (AML-ALL)

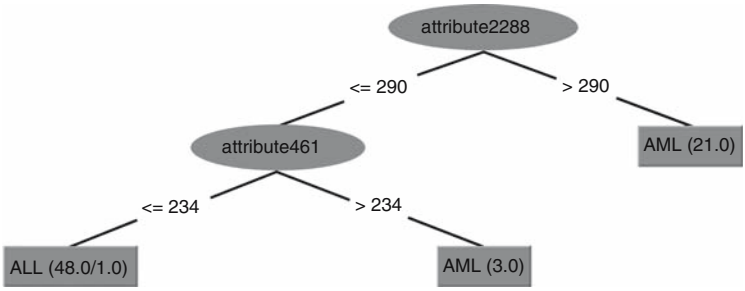


Fig. 2. J48 decision tree generated from amlall3 dataset (98.61% accuracy)

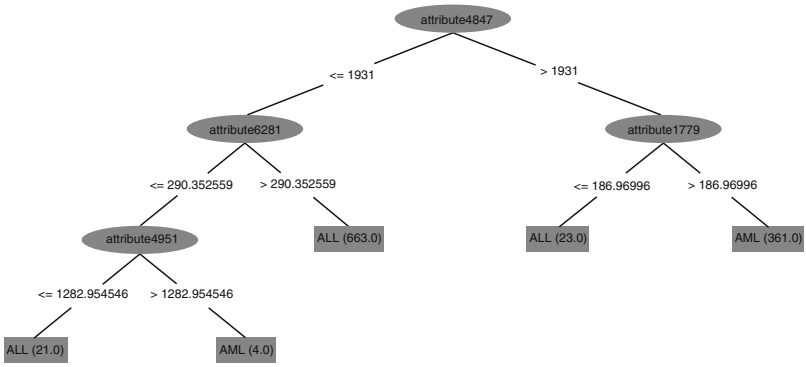


Fig. 3. CMMC-2 decision tree from amlall3 dataset (100% accuracy)

dataset. The first tree (Fig. 2) was constructed using J48 algorithm, while the second tree (Fig. 3) used CMMC-2 algorithm.

The first tree was built from all 72 samples of the amlall3 dataset, where SVM based filtering was used. J48 tree classified all but one sample correctly using only two genes. On the other hand CMMC-2 tree classified all 72 and additional 1,000 artificial samples correctly using only four genes. The decision tree yields five rules.

The goal of the decision tree presented in Fig. 3 is to derive short rules that explicitly clarify the types of leukemia and are convenient for expert interpretation. The rules are summarized in Table 5 that is accompanied by Table 4. The rule extraction process is based on the attribute to gene name mappings that are presented in Table 4.

To demonstrate the difference between the two constructed trees (J48 and CMM), a set of the strongest rules were chosen that were revealed by the CMM tree but failed to be recognised by the J48 tree. The rules, presented in Table 4, are directly extracted according to the corresponding branches of the decision tree, presented in Fig. 3. An interpretation of the rules (Table 5) is provided in Sect. 4.

Table 4. Attributes for gene mapping used in both decision trees

Attribute no.	Gene description
2,288	DF D component of complement (adipsin)
461	Liver mRNA for interferon-gamma inducing factor (IGIF)
4,847	Zyxin X95735_at
6,281	MYL1 Myosin light chain (alkali)
1,779	MPO Myeloperoxidase
4,951	Nucleoside-diphosphate kinase

Table 5. Rules for Leukemia classification derived from CMM decision tree

1. IF (zyxin X95735 NOT EXPRESSED) AND (MYL1 Myosin Light Chain EXPRESSED) ⇒ ALL
2. IF (zyxin X95735 NOT EXPRESSED) AND (MYL1 Myosin Light Chain (Alkali) NOT EXPRESSED) AND (Nucleoside-diphosphate kinase EXPRESSED) ⇒ AML
3. IF (zyxin X95735 EXPRESSED) AND (MPO Myeloperoxidase EXPRESSED) ⇒ AML

4 Discussion and Conclusion

Acute leukemia which is of lymphoid origin is called Acute Lymphocytic Leukemia (ALL) and a malignant disorder where myeloid blast cells accumulate in the marrow and bloodstream is called Acute Myelocytic Leukemia (AML). A study conducted by Golub et al. [21] has revealed 50 predictive genes that differentiate between ALL and AML. The report has shown that over expression of myosin light chain (M31211) leads to the ALL. The report also has indicated that zyxin 2 X95735, an adhesion plaque protein a component of a signal transduction pathway that mediates adhesion stimulated changes in gene expression, plays a significant role in AML. In a recent study conducted by Umpai and Aitken [35] has demonstrated that the gene X95735 zyxin significantly determines AML whereas myosin light chain over expression frequency is higher in ALL patients. Some other studies also have demonstrated the similar result. For example, Aris and Rece [36] has demonstrated the differentiation technique of AML and ALL based on the fact that zyxin is significant in AML, on the other hand, myosin light chain expression is significant in ALL patients. The French–American–British (FAB) group [37] has standardized the nature of ALL and AML on the basis of myeloperoxidase (MPO) expression. According to the criteria, the AML group demonstrates greater than 3% MPO and/or Sudan Black B (SBB) blast. The study has also revealed that, 70–75% of AML cases show myeloid associated antigens positive, for example, CD13, CD33, MPO etc., thus, it is evident that zyxin

X95735 and myeloperoxidase overexpression determines the AML subgroup of acute leukemia. Expression of nm23-H1/Neocleoside diphosphate kinase (NDPK) correlates inversely with the metastasising potential of some human tumours. The nucleoside diphosphate kinase enzymatic activity possessed by several isomers, for example, Nm23 H1 and NM23 H2, is increased significantly in AML cells and a higher level of nm23-H1 expression is correlated with a poor prognosis in AML. A study, conducted by Yokoyama et al. revealed that 110 AML patients have demonstrated the increased nm23-H1 mRNA level which showed a resistance in response to initial chemotherapy. They also have demonstrated that nm23 H1 has an enormous prognostic affect in AML, especially in AML-M5 (acute monocytic leukemia) [38]. From the discussion above it is evident that proposed adaptation of CMM model gives accurate trees that carry additional knowledge compared to classical decision trees. The research shows that the proposed CMMC demonstrated 2% higher accuracy compared to the classical decision trees. In addition to that, it has been demonstrated that the best CMMC method even can manage to keep the complexity level of the tree very low. Therefore, it is evident that CMMC tree was only twice as large as an original tree. The proposed CMMC model used a well known C4.5 algorithm for building the final decision tree. One of the problems with classical decision tree algorithms is that splitting in the lower lying nodes is based on fewer samples than splitting in the nodes near the root of the tree. Therefore, splitting toward the leaves is less reliable. The idea of using less artificial data points in the upper nodes by introducing them at the lower nodes of a decision tree can be applied in future for better accuracy. Although a part of this idea was already proposed in this chapter, there are still a lot of possible variations to this idea left to be researched. This study has also opened another important direction for further research. A small ensembles comprising of only three or five classifiers that can still be interpreted in the form of rules or some novel visualization techniques can be introduced to demonstrate the comprehensibility of microarray data in order to predict diseases.

References

1. L.-H. Loo, Identifying Differentially Expressed Genes in DNA Microarray Data, PhD Thesis, Drexel University, 2004
2. Z. Guo, T. Zhang, X. Li, Q. Wang, J. Xu, H. Yu, J. Zhu, H. Wang, C. Wang, E. J. Topol, Q. Wang and S. Rao, Towards precise classification of cancers based on robust gene functional expression profiles, BMC Bioinformatics, vol. 6, no. 1, p. 58, 2005
3. J. Khan, J. S. Wei, M. Ringner, L. H. Saal, M. Ladanyi, F. Westermann, F. Berthold, M. Schwab, C. R. Antonescu, C. Peterson and P. S. Meltzer, Classification and diagnostic pre-diction of cancers using gene expression profiling and artificial neural networks, Nature Medicine, vol. 7, no. 6, pp. 673–679, 2001

4. B. Brors, A. Kohlmann, S. Schnittger, C. Schoch, T. Haferlach and R. Eils, Classification of Cytogenetically Defined AML Patients by Decision Tree Analysis of Statistically Selected Gene Expression Data, in Proceedings of 43rd Annual Meeting of the American Society of Hematology (ASH01), Orlando, FL (USA), December 7–12, 2001
5. J. Li and K. Ramamohanarao, A Tree-based Approach to the Discovery of Diagnostic Biomarkers for Ovarian Cancer, in Proceedings of the PAKDD 2004, pp. 682–691, Sydney, Australia, February 2004
6. M. Dettling, BagBoosting for tumor classification with gene expression data, Bioinformatics, vol. 20, no. 18, pp. 3583–3593, 2004
7. D. P. Berrar, B. Sturgeon, I. Bradbury, C. S. Downes and W. Dubitzky, Microarray Data Integration and Machine Learning Techniques For Lung Cancer Survival Prediction, in Proceedings of Critical Assessment of Microarray Data Analysis (CAMDA 2003), Durham, North Carolina, USA, pp. 43–54, November 2003
8. P. Domingos, Knowledge discovery via multiple models, Intelligent Data Analysis, vol. 2 no. 1–4, pp. 187–202, 1998
9. R. Tibshirani and K. Knight, Model search and inference by bootstrap bumping, Journal of Computational and Graphical Statistics, vol. 8, pp. 671–686, 1999
10. O. Boz, Converting a Trained Neural Network To a Decision Tree DecText – Decision Tree Etxractor, PhD thesis, Computer Science and Engineering, Lehigh University, 2000
11. M. W. Craven, Extracting Comprehensible Models from Trained Neural Networks, PhD thesis, University of Wisconsin – Madison, 1996
12. Z.-H. Zhou and Y. Jiang, NeC4.5: neural ensemble based C4.5, IEEE Transactions on Knowledge and Data Engineering, vol. 16, no. 6, pp. 770–773, 2004
13. V. Estruch, C. Ferri, J. Hernndez-Orallo and M. J. Ramrez-Quintana, Simple Mimetic Classifiers, in Proceedings of IAPR International Conference on Machine Learning and Data Mining (MLDM2003), pp. 156–171, 2003
14. D. Cohn, L. Atlas and R. Ladner, Improving generalization with active learning, Machine Learning, vol. 15, pp. 201–221, 1994
15. M. W. Craven and J. W. Shavlik, Extracting comprehensible concept representations from trained neural networks, in Working Notes on the IJCAI'95 Workshop on Comprehensibility in Machine Learning, Montreal, Canada, pp. 61–75, 1995
16. H. Zhang, C. Y. Yu and B. Singer, Cell and Tumor Classification Using Gene Expression Data: Construction of Forests, in Proceedings of National Academy of Sciences U S A, vol. 100, no. 7, pp. 4168–4172, 2003
17. L. Breiman, Bagging predictors, Machine Learning, Vol. 24, no. 2, pp. 123–140, 1996
18. L. Breiman, Random forests, Machine Learning, Vol. 45, no. 1, pp. 5–31, 2001
19. T. G. Dietterich, Ensemble Learning, in The Handbook of Brain Theory and Neural Networks, 2nd ed., M. A. Arbib, Ed. MIT, Cambridge, MA, pp. 405–408, 2002
20. J. Li and H. Liu, Ensembles of Cascading Trees, in Proceedings of IEEE International Conference on Data Mining (ICDM 2003), IEEE Computer Society, Melbourne, p. 585

21. T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield and E. S. Lander, Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring, Science, vol. 286, no. 5439, pp. 531–537, 1999
22. L. J. van 't Veer, H. Dai, M. J. van De Vijver, Y. D. He, A. A. Hart, M. Mao, H. L. Peterse, K. Der Kooy, M. J. Marton, A. T. Witteveen, G. J. Schreiber, R. M. Kerkhoven, C. Roberts, P. S. Linsley, R. Bernards and S. H. Friend, Gene expression profiling predicts clinical outcome of breast cancer, Nature, vol. 415, pp. 530–536, 2002
23. G. J. Gordon, R. V. Jensen, L.-L. Hsiao, S. R. Gullans, J. E. Blumenstock, S. Ramaswami, W. G. Richards, D. J. Sugarbaker and R. Bueno, Translation of microarray data into clinically relevant cancer diagnostic tests using gene expression ratios in lung cancer and mesothelioma, Cancer Research, vol. 62, no. 17, pp. 4963–4967, 2002
24. S. A. Armstrong, J. E. Staunton, L. B. Silverman, R. Pieters, M. L. den Boer, M. D. Min-den, S. E. Sallan, E. S. Lander, T. R. Golub and S. J. Korsmeyer, MLL translocations specify a distinct gene expression profile that distinguishes a unique leukemia, Nature Genetics, vol. 30, no. 1, pp. 41–47, 2002
25. Y. Lu and J. Han, Cancer classification using gene expression data, Information Systems, vol. 28, no. 4, pp. 243–268, 2003
26. I. Witten and E. Frank, Data Mining: Practical Machine Learning Tools with Java Implementations, Morgan Kaufmann, San Francisco, 2000
27. J. R. Quinlan, Induction of decision trees, Machine Learning, vol. 1, pp. 81–106, 1986
28. A. Ben-Dor, N. Friedman and Z. Yakhini, Scoring genes for relevance, Agilent Technologies Technical Report AGL-2000-13
29. I. Kononenko, Estimating Attributes: Analysis and Extensions of Relief, in Proceedings of ECML'94, pp. 171–182, Springer, Berlin Heidelberg New York, 1994
30. Y. Wang and F. Makedon, Application of Relief-F Feature Filtering Algorithm to Selecting Informative Genes for Cancer Classification Using Microarray Data, in Proceedings of IEEE Computational Systems Bioinformatics Conference, pp. 497–498, Stanford, California, 2004
31. I. Guyon, J. Weston, S. Barnhill and V. Vapnik, Gene selection for cancer classification using support vector machines, Machine Learning, vol. 46, no. 1–3, pp. 389–422, 2002
32. K. Fujarewicz, M. Kimmel, J. Rzeszowska-Wolny and A. Swierniak, A note on classification of gene expression data using support vector machines, Journal of Biological Systems, vol. 11, no. 1, pp. 43–56, 2003
33. T. Hastie, R. Tibshirani and J. Friedman, The Elements of Statistical Learning, Springer, Berlin Heidelberg New York, 2001
34. M. Braga-Neto and E.R. Dougherty, Is cross-validation valid for small-sample microarray classification?, Bioinformatics, vol. 20, no. 3, pp. 374–380, 2004
35. T. Umpai and S. Aitken, Feature selection and classification for microarray data analysis: Evolutionary methods for identifying predictive genes, BMC Bioinformatics, vol. 6, no. 148, 2005
36. V. Aris and M. Rece, A Method to Improve Detection of Disease Using Selectively Expressed Genes in Microarray Data, Methods of Microarray Data Analysis, Kluwer, Dordecht, 2002

37. A. Venditti, G.D. Peeta, F. Buccisano, A. Tambarini, et. al., Minimally differentiated acute myleoid leukemia (AML-MO): Comparisson of 25 cases with other French–American–British subtypes, Blood, vol. 89, no. 2, pp. 621–629, 1997
38. A. Yokoyama, J. Okabe-Kado, et. al., Evaluation by multivariate analysis of the differentiation inhibitory factor nm23 as a prognostic factor in acute myelogenous leukemia and application to other hematologic malignancies, Blood, vol. 91, no. 6, pp. 1845–1851, 1998

On the Complexity of the Privacy Problem in Databases

Bhavani Thuraisingham

University of Texas, Dallas, TX, USA

Summary. This paper explores the complexity of the privacy problem. In particular the recursion theoretic properties of the privacy problem are examined. We view the privacy problem as a form of inference problem in databases. We develop a theory for the privacy problem based on deductive databases and then prove some properties of the problem. Essentially while our previous papers describe strategies and approaches to handle the privacy problem as well as designs of privacy controllers and privacy preserving data mining tools, this paper explores the foundations of the privacy problem.

1 Introduction

As we have stated in our previous papers (see [THUR03a], [THUR03b], [THUR03c] and [THUR03d]), privacy is about protecting information about individuals. Privacy has been discussed a great deal in the past especially when it relates to protecting medical information about patients. Social scientists as well as technologists have been working on privacy issues. However, privacy has received enormous attention during the recent years. This is mainly because of counter-terrorism and national security. For example in order to extract information about various individuals and perhaps prevent and/or detect potential terrorist attacks data mining tools are being examined. We have heard a lot about national security vs. privacy in newspapers, magazines and television talk shows. This is mainly due to the fact that people are now realizing that to handle terrorism, the government may need to collect information about individuals and mine this information. This is causing a major concern with various civil liberties unions [THUR03a], [THUT03b]. Consequently there is now much research on privacy preserving data mining that attempts to maintain some level of privacy while carrying out data mining [THUR05a], [THUR05b].

Our previous papers (see [THUR05c]) attempt to provide solutions to handle the privacy problem. In particular, we described a system that enforces

B. Thuraisingham: *On the Complexity of the Privacy Problem in Databases*, Studies in Computational Intelligence (SCI) **118**, 353–374 (2008)
www.springerlink.com

privacy constraints during database query, update and design operations. We have also examined the use of semantic data models for reasoning about privacy constraints [THUR06]. In our current paper we focus on developing a theory of the privacy problem based on recursive functions and computability theory. A formulation of the privacy problem is given and its recursion theoretic properties are investigated.

Our ultimate goal is to obtain a complete characterization of the privacy problem and investigate measures of complexity. Such an investigation could usefully begin with aspects of recursive function theory. This is because recursive function theory from which the notion of computability is derived (see [ROGE67]) has provided the basis from which other abstract theories such as computational complexity (see [BLUM67]) and Kolmograv complexity theory (see [KOLM65]) have evolved (we refer to [BRAI74], [MATC78] and [CALU88] for a discussion on this evolution). Moreover the work of Rosza Peters (see [PETE81]) has shown the practical significance of recursive function theory in computer science and provided the basis for possible exploitation of such results. Therefore the study of the foundations of the privacy problem could usefully begin with aspects of recursive function theory. We have investigated the recursion theoretic properties of the privacy problem associated with database design. We give a formulation of the privacy problem as a database design problem and investigate the recursion theoretic properties of this problem. Our research is influenced by our work on privacy constraints and privacy enhanced databases (see [THUR05c]) as well as our prior research on the inference problem (see [THUR91]). For a background on the inference problem we refer to [MORG87], [THUR87], and [HINK88]. For a discussion of multilevel secure databases we refer to [AFSB83], [THUR05b].

The organization of this paper is as follows. In Sect. 2, which is the essence of this paper, we define a set X(L) corresponding to each privacy level L. This set consists of all databases D that are not privacy enhanced with respect to privacy level L. The privacy problem with respect to private level L would then be the membership problem for the set X(L). That is, given a database D, if it can be effectively decided whether D belongs to X(L), then one can decide whether the design of the database D is privacy enhanced with respect to level L. By privacy enhanced at a level L we mean that all information that should be labeled at privacy level L is correctly labeled at level L. We prove properties of the set X(L). In Sect. 3 we discuss directions for future work on the foundations and complexity of the privacy problem.

2 A Theory of the Privacy Problem

2.1 Overview

The theory developed in this section is based on privacy enhanced/non-privacy enhanced database designs. Given a database and a set of privacy constraints, if it can be effectively decided that the database is designed in such a way

that privacy violations via inference cannot occur, the privacy problem can be solved. The privacy controller, which is the device that handles the privacy problem (see [THUR03c]) will implement the decision procedure for the set, which consists of all the non-privacy enhanced database designs.

We use the notion of a deductive database in our formulation of the privacy problem. A deductive database consists of a database and a set of rules, which enable new data to be deuced from the extensional data [GALL78]. We first define a deductive database and then define the privacy problem with respect to a privacy level. This problem is the set of all multilevel databases that are non-privacy enhanced at that privacy level. We state and prove recursion theoretic properties of the set. In this way, a classification of the privacy problem can be obtained based on the classifications of the recursively enumerable sets.

Much of the work described in this section is built on the work of Cleave (see [CLEAV72], [CLEA73], [CLEAV75]) and Thuraisingham (see [THUR82], [THUR83, [THUR86]) on deductive systems. Their work is derived from Post's celebrated work on the reduction of combinatorial decision problems (see [POST43], [POST44]). The research is also influenced by Thuraisingham's more recent work on the inference problem in secure databases (see [THUR91]).

The organization of this section is as follows. In Sect. 2.2 we first define a database. Associated with the notion of databases is a set of rules, which enable new data to be deduced from the extensional data in the database. The rules will be modeled by a function called a privacy function. In Sect. 2.3 some recursion theoretic properties of the privacy problem are stated and proved. While Sect. 2.3 assumes that the privacy functions are classified at the lowest privacy level (which we will call system-low and is usually the level Public), in Sect. 2.4 we discuss multilevel privacy functions where privacy constraints are themselves are assigned different privacy levels.

2.2 The Privacy Problem

The definition of the privacy problem will be progressively given below. We first define the notions of a database (as given in [CHAN82]), a deductive database, a multilevel database, and a multilevel deductive database.

Database: Let U be some countable domain. A relational database (or database) is a tuple $B = (P, R1, R2, \ldots Rk)$ where P is some finite subset of U and for each i $(1 \leq i \leq k)$ Ri is a subset of Dai for some ai $>= 0$. The integer ai is called the rank of Ri and B is said to be of type $a = (a1, a2, \ldots ak)$.

Deductive Database: A deductive database is a pair $\langle B, R \rangle$ where B is a database and R is a finite set of rules. These rules may be used to deduce new data from the data in B.

The following is an example of a deductive database:

B = {m1, m2, ... mn)
R = { (i) m1 → e
(ii) m3, m4 → f
(iii) m5, m2 → g
(iv) m6 → A*m6 (* is the concatenation operation)
(v) m7, e → p
(vi) m6, m8 → b, c
(vii) X → Y where Y is a subset of a set X
(viii) if Y = {y1, y2, ... yt) is a subset of X = {x1, x2, ... xk} and if
y1, y2, ... yt → z1, z2, ... zj (1 ≤ i ≤ j) is a rule in R, then
X → X ∪ {z1,z2, ... zj} is also a rule in R

Note that the symbol → is the "implies" relationship. The symbol ∪ is the union relationship and the symbol ∈ is the membership relationship. While we focus on relational databases, the discussion applies for any database.

A deductive database can be regarded to be a semi-thue system. For a discussion of such systems we refer to [HOPC79]). The set of data that is deduced from B is not necessarily finite. Furthermore, the set of rules R can be regarded as a privacy function whose definition is given below.

Privacy Function: A function f: G → Pw(G) is a privacy function if there exists recursive functions a and b such that for all x, f(x) = Da(x) and f-1(x) = Db(x) where De is the eth finite set in some standard enumeration, G is a countable set of entities (an entity could also be a database), Pw(G) is the set of all finite subsets of G and

f-1(x) = {y: x ∈ f(y)}
Furthermore, for an X ∈ Pw(G)
f(X) = ∪x ∈ X {y: y ∈ f(x)}
Note that f-1 is the inverse of f.
The symbol → takes a function from domain to range.

The set of all privacy functions is denoted by **PF**.
A privacy function is deterministic if f(x) has atmost one member.

Consequences: Next we define Cnf(x), which is the set of all consequences of x by a privacy function f. This set consists of all the information that can be inferred from a set of axioms x by f.

Define y ∈ Cnf(x) if one of the following conditions holds:

(i) y = x
(ii) y ∈ f(x)
(iii) There exists a sequence of numbers y1, y2, ... yn such that y1 = x, yn = y and for all i (1 ≤ I ≤ n-1), yi+1 ∈ Cnf(yi).

The following remarks are in order.

- Corresponding to a deductive database ⟨B, R⟩ there is a pair ⟨n, F⟩ where n is the Gödel number of B in the effective enumeration of all databases and F is the privacy function, which corresponds to R.
- There is no loss of generality in regarding a deductive database ⟨B, R⟩ as ⟨B, F⟩ or ⟨n, F⟩ where F is the privacy function which corresponds to R and n is the Gödel number of B in the effective enumeration of all databases.

Figure 1 illustrates a partial graphical representation of the privacy function, which corresponds to the deductive database considered in this example. A rule R is applied to a data set to obtain a new data set.

Multilevel Database: A multilevel database is a triple ⟨B, T, A⟩ where B is a database, T is a recursive set of privacy constraints and A is an algorithm (i.e. an effective procedure) which assigns privacy levels to the data based on T. (Note that since T is recursive, one can effectively decide whether a privacy constraint belongs to T.)

For example consider the database B = {m1, m2, ... mn}. If a subset {m1, m4, m5} of the database is given as input to A, A terminates with the privacy level of the subset of the output. The privacy level is computed based on the privacy constraints in T. In this example, the privacy level is the maximum privacy level of the sets {m1}, {m4}, {m5}, {m1, m4}, {m1, m5},

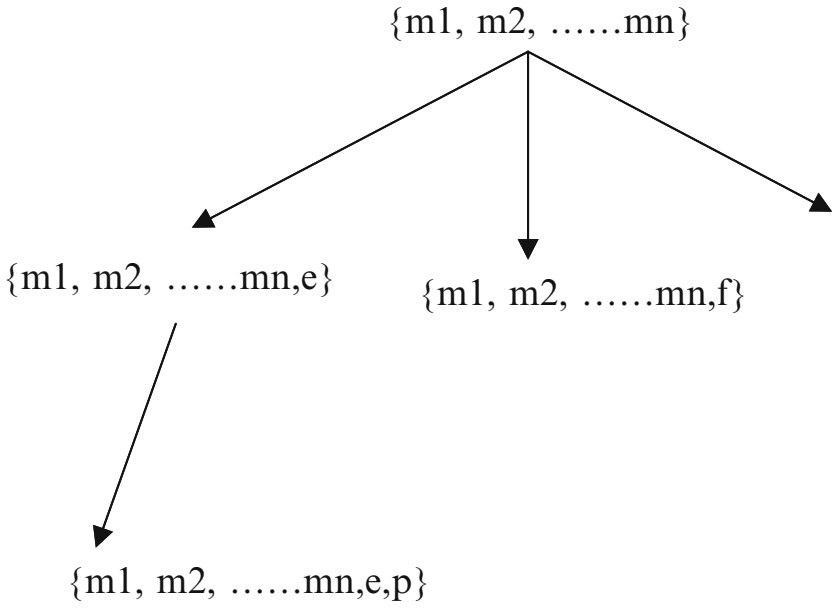


Fig. 1. Graphical representation of a privacy function

{m4, m5} and {m1, m4, m5}. If a privacy constraint does not explicitly classify a piece of data then its privacy level is assumed to be system-low which we assume to be the lowest privacy level (e.g., public) supported by the system. This is usually the level Public. Since T is recursive, one can determine the privacy level of any piece of data.

Multilevel deductive database: A multilevel deductive database is a quadruple ⟨B, F, T, A⟩ where B is a database, F is a privacy function, T is a recursive set of privacy constraints and A is an algorithm that assigns privacy levels to the data in the database as well as to the derived data.

For example, in the graph of Fig. 1, algorithm A assigns privacy levels to each node based on the recursive set of privacy constraints. We assume that there is a unique algorithm, which assigns privacy levels to all the data. Therefore we do not include the algorithm A in the discussion. Some directions toward algorithm A are given in [THUR05c].

Privacy problem: The privacy problem with respect to privacy level L is the set of all triples ⟨B, F, T⟩ such that there is some x belongs to CnF(B) and the privacy level of x dominates L. Note that we assume that the set of privacy levels form a lattice (see also [THUR03c]). Formally stated the privacy problem at level L is the set:

PP[L] = {⟨B, F, T> | Level (B) ≤ L and ∃x (x ∈ CnF(B) and Level(x) ⟩ L)} where ∃ is the "there exists" symbol.

If the set **PP**[L] is decidable then given a database B, a privacy function F and a set of privacy constraints T, the privacy controller can decide whether B is privacy enhanced with respect to the privacy level L under the constraints T and the inference rules F.

2.3 Properties of the Privacy Problem

As stated earlier if the privacy problem is solvable, then there is no problem. That is, given a database B, a set of inference rules R and a set of privacy constraints T, one can effectively decide whether the database is privacy enhanced with respect to a privacy level. Unfortunately, as we will see, the privacy problem in general is not solvable. In this section we will state and prove unsolvability results of the privacy problem. Once it is determined that the problem is unsolvable. The next question that needs to be answered is: to what extent is the privacy problem unsolvable? Is the problem creative? If so, then it is of the highest degree of unsolvability. Our approach to showing that an unsolvable problem is creative is to first show that it is nonsimple. This is because no creative set can be simple. The next step is to show that the nonsimple problem is a cylinder. This is because all creative sets are cylinders. Finally we show that the cylindrical problem is creative.

In the discussion given in this section we assume that the privacy functions as well as the privacy constraints are classified at the level Public (which is

system-low). That is, they are visible to all users. By assuming that the privacy functions are public, we mean that users at all privacy levels utilize the same strategies to draw inferences.

In Theorem 1 we show that the privacy problem is recursively enumerable with respect to all privacy levels. In addition, the privacy problem is either recursive or nonsimple. If the privacy functions are deterministic, then we show that the privacy problem is either recursive or a cylinder. We show that if the privacy level L1 dominates the privacy level L2 (such as the Private level dominates the Public level), then the privacy problem with respect to L1 is a subset of the privacy problem with respect to L2.

Theorem 1.

(i) For each privacy level L, PP[L] is recursively enumerable.
(ii) For each privacy level L, PP[L] is either recursive or nonsimple.
(iii) If all privacy functions which model the rules in deductive databases are deterministic, then for each privacy level L, PP[L] is either recursive or a cylinder.
(iv) If the privacy level L1 dominates the privacy level L2, then PP[L1] $\subseteq$ PP[L2], where $\subseteq$ is the subset function.

Proof of Theorem 1

Proof of Theorem 1(i). Let $\langle$B1, F1, T1$\rangle$, $\langle$B2, F2, T2$\rangle$, ... be an effective enumeration of all multilevel deductive databases where B1, B2, B3 ..... are all visible at level L.

The set **PP**[L] is enumerated as follows:

Step 1: Perform 1 step in the computation of CnF1(B1). If it converges, let X be the result. Check whether the privacy constraints in T1 classify X at a level which dominates L. If so, list $\langle$B1, F1, G1$\rangle$ as a member of **PP**[L].

Step 2: If CnF1(B1) did not converge in step 1, perform two steps in the computation of CnF1(B1). If it converges let X be the result. Check whether the privacy constraints in T1 classify X at a level which dominates L. If so, list $\langle$B1, F1, T1$\rangle$ as a member of **PP**[L].
Perform one step in the computation of CnF1(B2), CnF2(B1), CnF2(B2). For each computation CnF(B) do the following: if it converges let X be the result. Let T be the privacy constraints associated with $\langle$B, F$\rangle$, Check whether the privacy constraints in T classify X at a level which dominates L. If so, list $\langle$B, F, T$\rangle$ as a member of **PP**[L].

Step 3: The following computations are performed:
If CnF1(B1) did not converge in step 2, perform three steps in the computation of CnF1(B1).
If CnF2(B1), CnF2(B2), CnF2(B2) did not converges in step 2, perform two steps in their computation

Perform one step in the computation of CnF1(B3), CnF2(B3), CnF3(B3), CnF3(B1), CnF3(B2).
For each computation CnF(B) performed in step 3, if it converges, then using the corresponding set T of privacy constraints check whether the privacy level of the result dominates L. If so place ⟨B, F, T⟩ in the list.
Continue with steps 4, 5, 6, - - - -. The list that is generated is **PP**[L].

This proves Theorem 1(i).

Proof of Theorem 1(ii). If **PP**[L] is recursive, then it is nonsimple. Assume that **PP**[L] is nonrecursive. We need to show that for every privacy level L, the complement of **PP**[L] (denoted COMP-**PP**[L]) has an infinite recursively enumerable subset. It suffices to prove the following result α.

Result α: There is an element ⟨B, f, T⟩ of the complement of **PP**[L] such that Cnf(B) is infinite.
If result α does not hold, then a decision procedure can be given for **PP**[L] as follows:

Given a triple ⟨B, f, T⟩ list the members of Cnf(B). As a member x is listed, use the privacy constraints in T to compute the level of x. If the level is not dominated by L, then ⟨B, f, T⟩ belongs to **PP**[L]. For those elements, which do not belong to **PP**[L], by our assumption, Cnf(B) is finite. This means that if Cnf(B) is not finite, then it will definitely be the case that there is an x in Cnf(B) such that the privacy level of x is not dominated by L. This gives a decision procedure for **PP**[L] which contradicts our assumption. Therefore Result α holds.

The infinite recursively enumerable subset S of COMP-**PP**[L] can be enumerated as follows:

Let ⟨B, f, T⟩ be the element of **PP**[L] such that Cnf(B) is infinite. The following is an enumeration of S:

$$\langle B, f, T\rangle, \langle f(B), f, T\rangle, \langle f(f(B), f, T\rangle, \text{- - - - - -}$$

This proves Theorem 1(ii).

Proof of Theorem 1(iii). By Young's Lemma [YOUN66], it suffices to prove the following result β.

Result β*:* For every privacy level L, there is a recursive function g such that:

$X \in$ **PP**[L] $\rightarrow$ Wg(X) $\subseteq$ **PP**[L
$X \in$ COMP-**PP**[L] $\rightarrow$ Wg(X) $\subseteq$ **PP**[L]
And Wg(X) is infinite

Note that the privacy functions associated with **PP**[L] are assumed to be deterministic and We is the eth recursively enumerable set in some standard enumeration. COMP-Z is the complement of set Z.

We now prove the result β.

Since **PP**[L] is recursively enumerable and nonsimple, its complement has a recursively enumerable subset. Let this subset be Q.

Given a triple ⟨B, f, T⟩, compute the following procedure:

Step 0: List ⟨B, f, T⟩
Step r (r ≥ 0): Compute fr(B)
Note that f2(B) is f(f(B)) and f3(B is f(f(f(B)))

(i) If fr(B) is empty, then list the members of **PP**[L] and stop the procedure (note that the procedure does not terminate as **PP**[L] is infinite. What we actually mean is do not go to step (r+1).
(ii) If fr(B) is assigned a privacy level by T which is not dominated by L then enumerate the members of set Q and stop. (Note again that this procedure does not halt).
(iii) If neither (i) nor (ii) hold, list ⟨fr(B), f, T⟩) and go to step (r + 1).

It can be shown that for each privacy level L, there is a recursive g such that given an element ⟨B, f, T⟩ the list enumerated is Wg(⟨B, enumerated is Wg(⟨B, f, T⟩).

This proves Theorem 1(iii).

Proof of Theorem 1(iv). Let the privacy level L1 dominate the ⟨B, f, T⟩ belongs to **PP**[L1] then ⟨B, f, T⟩ also belongs to to **PP**[L1] then ⟨B, f, T⟩ also belongs to **PP**[L2].

If ⟨B, f, T⟩ belongs to **PP**[L1] then there is an x which belongs to Cnf(B) such that the privacy level of x is not dominated by L1. Then the privacy level of x is also not dominated by L2. Therefore, x belongs to **PP**[L2]. That is, **PP**[L1] ⊆ **PP**[L2].

(Note that we make the assumption that the same privacy function is used for all privacy levels. Furthermore the constraints themselves are assumed to be public. That is, the constraints are visible at all privacy levels.)

This proves Theorem 1(iv).

In Theorem 1 we have shown that the privacy problem is recursively enumerable with respect to all privacy levels. This does not mean that there is a situation where the privacy problem is nonrecursive with respect to a privacy level. By a situation we mean a particular scenario in the world. The privacy functions and the privacy constraints used will be specific to a situation. As the situation changes the privacy problem also changes. In Theorem 2 we state some counter examples. A counter example depends on the particular situation under consideration.

In Theorem 2 we assume that there are only two privacy levels: Private and Public. This means at the private level there is no privacy problem. This is because no constraint will assign a level that is higher than the private level (e.g., highly private). Therefore the privacy problem at the private level will be

the empty set. At the public level the privacy problem could be nonempty. We show that there are situations in which the privacy problem at the public level is (i) nonrecusive, (ii) not creative if the privacy functions are deterministic, and (iii) neither recursive nor a cylinder.

Theorem 2.

(i) There is a situation where PP[Public] is not recursive.
(ii) Assuming that the privacy functions are deterministic, there is a situation where PP[Public] is not creative.
(iii) There is a situation where PP[Public] is neither recursive nor a cylinder.

Proof of Theorem 2(i). We first show that given a recursively enumerable set W, there is a situation S such that

$W \equiv m$**PP**[Public]. Note that $\equiv$ m is the many-one equivalence relationship. The result is then immediate from the following reasoning.

* It has been shown that there is set K which is creative. K is the set {x: the xth partial recursive function halts on input x}

* The situation S that is constructed from the recursively enumerbale set K will guarantee that **PP**[Public] is creative. This is because if the two sets A and B are many one equivalent and A is creative, then so is B. Therefore, if **PP**[Public] is creative then it cannot be recursive.

Given a recursively enumerable set W, we create a situation S by defining a set of privacy constraints and a privacy function. Let the set of privacy constraints be {(0,0)}. That is, the only element that is assigned the private level is the pair (0,0). We consider pairs of natural numbers. This does not cause any problem due to the existence of the pairing function from N × N onto N where N is the set of all natural numbers. The set of privacy constraints is recursive (note that in this case it is also finite) and does not depend on W.

We define a privacy function, which depends on W as follows. We assume that e is the index of W. The privacy function f for a pair (u, v) is defined as follows:

$$
\begin{aligned}
& \{(u, v+1)\} \text{ if } u \neq 0 \text{ AND NOT } T(e, u, v) \\
& \{(0, u+v) \text{ if } u \neq 0 \text{ AND } T(e, u, v) \\
f(u, v) = & \{(u, v-1) \text{ if } u = 0 \text{ AND } v \neq 0 \\
& \phi \text{ (the empty set) if } u = 0 \text{ AND } v = 0 \\
& \text{Note that T is the Kleene's T Predicate}
\end{aligned}
$$

The graphical representation of f is shown in Fig. 2. Note that defining f on pairs of natural numbers does not cause any problem. This is because one can define a privacy function g on N such that g(x) = {y: x = j(u,v) and y = j(p,q) for some x,y,p,q and (p,q) ∈ f(u,v)}
where j is the pairing function from NxN to N.

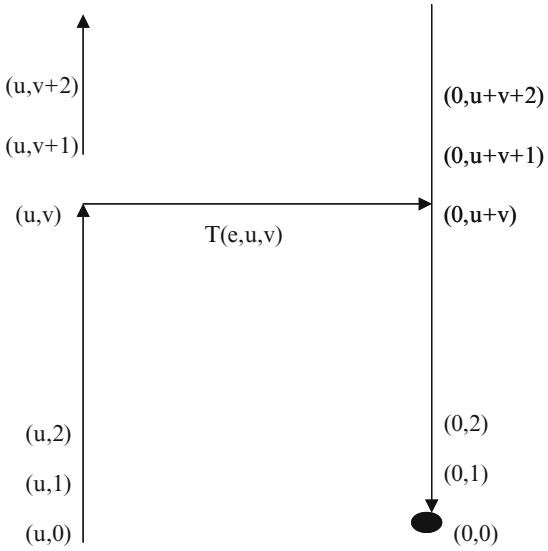


Fig. 2. Graphical representation of the privacy function f

$$\mathbf{PP}[\text{Public}] = \{(u, v) : \text{ there is a path via f from } (u, v) \text{ to } (0, 0)\}$$

It remains to be shown that We $\equiv$ m **PP**[Public] where We is the eth recursively enumerable set. Note that $\equiv$ m is many-one equivalence.

Let a $\in$ We and b $\in$ COMP-We. (COMP-We is the complement of We)

The function h defined below is a many-one reduction function from **PP**[Public] to We.

$$\begin{aligned} h(u, v) &= u \text{ if } u \neq 0 \text{ AND NOT}\exists w\langle v\ T(e, u, w) \\ &= b \text{ if } u \neq 0 \text{ and } \exists w\langle v\ T(e, u, w) \\ &= a \text{ if } u = 0 \end{aligned}$$

The function k defined below is a many-one reduction function from We to **PP**[Public]. k(u) = (u, 0) for all u.

This proves Theorem 2(i).

Proof of Theorem 2(ii). The privacy function f constructed for the proof of Theorem 2(i) was a deterministic function. Therefore we have shown that given a recursively enumerable set W, there is a situation S with deterministic privacy functions such that W is many-one equivalent to **PP**[Public]. It has been shown that there exist recursively enumerable sets, which are not creative. An example is the existence of a simple set [ROGE67]. Any set that is many-one equivalent to a non-creative set is also non-creative. Therefore, there is a situation in which **PP**[Public] is non-creative.

This proves Theorem 2(ii).

Proof of Theorem 2(iii). To prove that there is a situation in which **PP**[Public] is neither recursive nor a cylinder, we show that no partial recursive function $\sum e$ ($e \geq 0$) is a semi-cylinder function for **PP**[Public]. Then by Young's result on semi-cylinders [YOUN63], it follows immediately that **PP**[Public] is neither recursive nor a cylinder.

The following graph theory concepts are needed for the proof of the result.

Definitions

Let D be a digraph whose points (nodes) are in N and whose lines (edges) are ordered pairs of natural numbers. By x ∈D we mean x is a point in D. If x ∈ D and y ∈ D then by x → y (D) we mean that either x = y or (x, y) is a directed line in D, or there exist distinct points y1, y2, .. . yn such that x = y1, y = yn and for each i (1 ≤ i ≤ n-1) yi → yi+1. If x ∈ D, then D(x) is the connected component of D containing x. The in-degree (out-degree) respectively of x is the number of points y such that (y, z) ((x, y) respectively) is a directed line in D. We assume that the in-degree as well as out-degree of each point is atmost 2. A point is a root (leaf respectively) if its in-degree (out-degree respectively) is 0. We denote r(x) (t(x) respectively)) to be the least point y such that y is a root (leaf respectively) and y → x (x → y respectively) in D.

By x//y is meant x and y belong to different connected components. If x/y, then x and y belong to the same connected component, but it is not the case that x → y or y → x. If (x,y) and (x,z) are lines, then z is denoted by y with respect to x (note that y is denoted by z* with respect to x).*

A digraph is labeled if some of its points are distinguished from one another by names drawn from some given infinite list. By the term introduce the labels K1, K2, - - - - Kn (n ≥ 1) to the digraph D is mean the following: Find the least n numbers x1, x2, ... xn which are not points in D. Adjoin these numbers as points in D so that each point forms a new connected component. Label each xi (1 ≤ i ≤ n) by Ki.

By the term extend the digraph D to digraph D is meant the following:*
Let r1, r2, . . . rn be the roots of D and let t1, t2, ... tm be its leaves.
Find the least 4n+4m numbers $x_1 < x_2 < x_3 < \ldots . X_{4n} < y_1 < y_2 < y_3 < \ldots . y_{4m}$ *not in D. Adjoin these numbers as new points and include the following lines:*
(x1,r1), (x2,r1), (x1,x3), (x2,x4), (x5,r2), (x6,r2), (x5,x7), (x6,x8), - - - - - - (x4n-3, rn), (x4n-2, rn), (x4n-3,x4n-1), (x4n-2, x4n), (t1,y1), (t1,y2), (y3,y1), (y4,y2), (t2,y5), (t2,y6), (y7,y5), (y8,y6), ..., (tm,y4m-3), (tm,y4m-2), (y4m-1, y4m-3), (y4m,y4m-2).
The resulting digraph is D.*

A point is private if a privacy constraint classified that point.

Figure 3 illustrates the concepts we have described above.

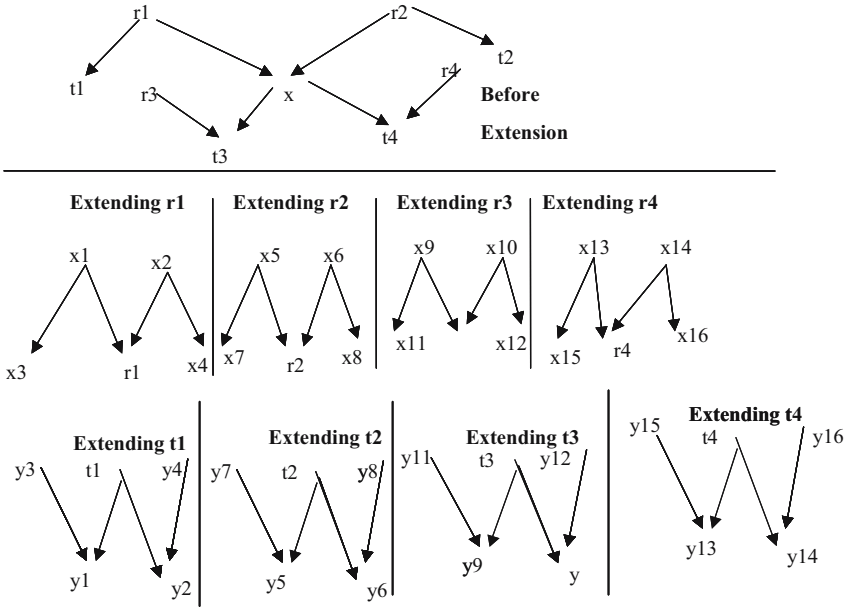


Fig. 3. Extending digraphs

Continuation of Proof We now prove Theorem 2(iii). The proof is divided into two parts. The first part consists of a program and the second part consists of three lemmas.

The program to be constructed in Part 1 (see Fig. 4) of the proof is the construction of labeled digraphs D0, D1, D2, ... with the following properties:

(i) There is a recursive function g such that for each m, g(m) is the Gödel number of Dm.
(ii) For each m, m is a point of Dm.
(iii) For each m, Dm + 1 is an extension of Dm, i.e. points of Dm are points of Dm + 1 and if x, y are points of Dm, then there is a line x to y in D if and only if there is a line from x to y in Dm + 1 where D = ∪{Di: i ∈ N}. Also Dm + 1 has a point which is the least number not a point in Dm.
(iv) If m ¡ p, then for any point x of Dm, all lines incident with x in Dm are lines of Dm + 1.
(v) For each m, a component Dm has atmost one label. These labels are taken from a given set {Pe: e ≥ 0} of markers. Also in the program, the dependence of a number on another number will be defined by induction.
(vi) The privacy constraints classify just the one point 0 at the Private level.

The second part of the proof consists of three lemmas by means of which it will be proved that there is a privacy function f such that for no e is it true that the eth partial recursive function $\sum$e is a semi-cylinder for **PP**[Public].

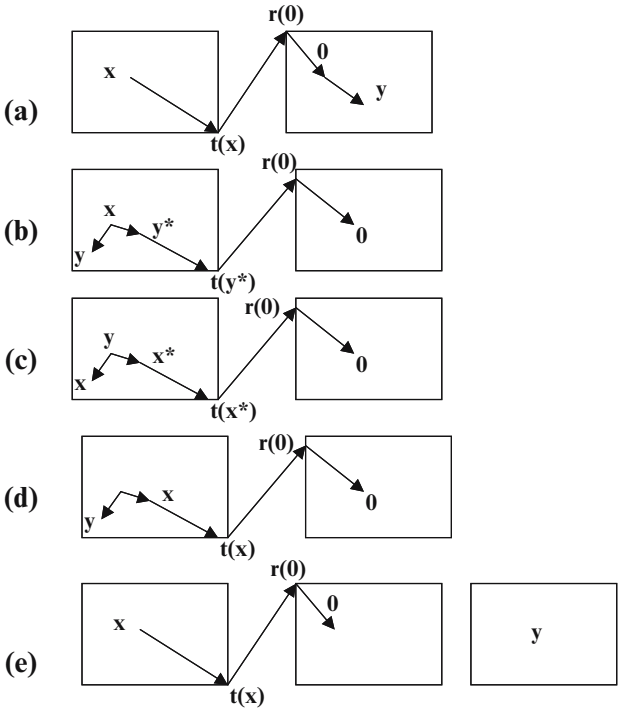


Fig. 4. Construction of a program

Program

Stage 0: D0 consist of the points 0 and 1 with 1 labeled P0.
Stage m (m ≥ n1):

Step 1: Introduce the label Pm to Dm-1 and extend the resulting graph to Dm*.

Step 2: Find the smallest number e ≤ m such that there exist numbers x, y, z all ≤ m satisfying the condition R(x, y, z, e, m) where R is the conjunction of the following conditions R1, R2 and R3.

$$\text{R1} \equiv \text{T(e, x, z) and M(z)} = \text{y}$$
$$\text{R2} \equiv \text{NOT (x} = \text{y)}$$
$$\text{R3} \equiv \text{x is labeled Pe in Dm}*$$

Note that T is the Kleene's T predicate and R1 implies that $\sum$e(x) = y. For a discussion of the functions T and M we refer to [MEND79].

If there is no such an e, then set Dm = Dm* and go to stage m + 1. Otherwise do the following:

$$\begin{aligned} \text{Define } em &= (\mu e)(\exists z, x, yall \leq m)\ R(z, e, x, y, e, m) \\ xm &= (\mu e)(\exists z, yboth \leq m)\ R(z, em, x, y, e, m) \\ ym &= (\mu e)(\exists z \leq m)\ R(z, em, xm, y, e, m) \end{aligned}$$

where μe is the least number e satisfying the condition.

For convenience let em, xm, ym be e, x, y, respectively. In step 3 of the program it will be ensured that:

$$x \in \mathbf{PP}[\text{Public}]\ \text{NOT} \equiv y \in \mathbf{PP}[\text{Public}]$$

where NOT ≡ is the not equivalent symbol.
The application of step 3 on e is called an attack on e.

Step 3:
(a) Delete Pe;
(b) Reintroduce al labels Ps such that s depends on e. Let the resulting graph be Dm+;
(c) Construct Dm from Dm+ a follows:

Case 1: y belongs to the connected component of 0 in Dm+.

1.1 It is not the case that y→0 in Dm+.
Set Dm = Dm ∪ (t(x), r(0)) (see Fig. 4a)
(That is, Dm results from Dm+ by joining the line (t(x), r(0)).
1.2 Case 1.1 does not hold
Set Dm = Dm+.

Case 2: y belongs to the connected component x in Dm+
2.1 x → y (Dm+)
Set Dm = Dm+ ∪ (t(y*), r(0)) (Fig. 4b)
2.2 y → x (Dm+)
Set Dm = Dm+ ∪ (t(x*), r(0)) (Fig. 4c)
2.3 x/y (Dm+)
Set Dm = Dm+ ∪ (t(x), r(0)) (Fig. 4d)

Case 3: Case 1 and Case 2 do not hold.

3.1 The connected component of y does not have a label.
Set Dm = Dm+
3.2 The connected component of y has a label Pj.
(i) If j ⟩ e, delete Pj and reintroduce it
Set Dm = Dm+ ∪ (t(x), r(0)) (Fig. 4e)
(ii) If j ⟨ e, record that e depends on j.
Set Dm = Dm+ ∪ (t(x), r(0)) (Fig. 4e)

This Ends the Program

Set D = ∪ {Dm: m ∈ N} where

$$\begin{aligned} \mathrm{Di} \cup \mathrm{Dj} &= \mathrm{Di} \text{ if } \mathrm{i} > \mathrm{j} \\ &= \mathrm{Dj} \text{ if } \mathrm{i} \leq \mathrm{j} \end{aligned}$$

Clearly for any point x of Dm, all lines incident with X in D are lines of Dm + 1.

$$\begin{aligned} \text{Define } \mathrm{f(x)} &= \{\mathrm{y}: \ (\mathrm{x}, \mathrm{y}) \text{ is a point of D}\} \\ &= \{\mathrm{y}: \ (\mathrm{x}, \mathrm{y}) \text{ is a line of Dx} + 1\} \\ \mathrm{f} - 1(\mathrm{x}) &= \{\mathrm{y}: \ (\mathrm{y}, \mathrm{x}) \text{ is a line of D}\} \\ &= \{\mathrm{y}: \ (\mathrm{y}, \mathrm{x}) \text{ is a point of Dx} + 1\} \end{aligned}$$

Clearly f belongs to **PF**. That is, f is a privacy function and

$$\mathbf{PP}[Public] = \{x: \ x \textit{ is public and } x \rightarrow 0 \ (D)\}$$

This ends Part I of the proof.

We now state and prove three lemmas which constitute the second part of the proof of Theorem 2(iii). We first need the following definition.

Definition

A label P is fixed at stage numbed H if either P remains assigned to the same point at all stages numbered n ≥ H or P remains unassigned at all stages numbered n ≥ H.

Lemma 1. *For each e, there exists a stage H(e) at which all labels Pi where i ≤ e are fixed.*

Proof of Lemma 1. There are four ways in which a label Pe can be moved.

(M1) Introduction at stage e (step 1)
(M2) Deletion at stage e (step 3a)
(M3) Reintroduction via an attack on i where e depends on i (step 3b)
(M4) Reintroduction via an attack on i under step 3c, case 3.2 (i)

The proof of the lemma is by induction on e.

Basis e = 0: P0 is introduced at stage 0 (M1). As 0 does not depend number and there is no i such that i < 0, P0 cannot be moved by M3 or M4. Therefore, if 0 is never attacked then P0 is fixed at stage 0. If 0 is attacked at stage m, then P0 is fixed and unassigned at stage m, i.e., H(0) = m.

Inductive step: As inductive hypothesis, assume that all labels Pi < e are fixed at stage H(e-1). Pe is introduced at stage e (M1). It cannot stage e (M1). It cannot be moved by M3 or M4 at a stage m ≥ H = MAX{H(e-1),e}. For,

if not, then a number I ⟨e will be attacked at a stage m. This is a contraction as Pi is fixed at stage H(e-1). Suppose e is never attacked at a stage m ≥ H, then Pe is fixed and unassigned at stage m, i.e. H(e) = m.

Then the inductive hypothesis implies that all labels Pi where i ≤ e are fixed at some stage H(e). The statement of the lemma now follows by induction on e.

Lemma 2. *For each e, if Pe is fixed and unassigned at state H(e) then* $\sum e$ *is not a semi-cylinder function for PP[Public].*

Proof of Lemma 2. Pe is introduced at state e. If it is fixed and unassigned at stage H(e), there is a stage m (e ≤ m ≤ H(e)) at which they were last deleted. That is, e was attacked at stage m. Suppose at stage m Pe was assigned to the point x. Then there exists a y ≤ m where NOT(x = y) such that $\sum$e(x) = y. Furthermore, in step 3c of stage m it would have been ensured that

$$x \to 0 \text{ NOT} \equiv y \to 0 \text{ (Dm)}$$

It now suffices to prove the following statement (V).

$$\text{(V)}: \ x \to 0 \text{ NOT} \equiv y \to 0 \text{ (D)}$$

For if (V) holds then

$$\{x\} \subseteq \mathbf{PP}[\text{Public}] \text{ NOT} \equiv y \in \mathbf{PP}[\text{Public}]$$

Therefore, as $\sum$e (x) = y, $\sum$e is not a semi-cylinder function for **PP**[Public].

Proof of Statement (V): To prove (V), we need to examine each case of the construction of the program and show that (V) holds. We only show this for case 1-1. In case 1-1, it is ensured that x → 0 (Dm), and it is not the case that y → 0 (Dm). We need to prove by induction on s (s ≥ 0) that:

(i) x → 0 (Dm+s); and
(ii) NOT y → 0 (Dm+s)

Now, (i) and (ii) are true for s = 0. It can be shown by examining the various cases that if (i) and (ii) are true for s = k, then they are true for s = k + 1.

This proves the statement (V) for case 1-1. Similarly we can show that (V) holds for the other cases also, Hence the lemma 2.

Lemma 3. *For each e,* $\sum e$ *is not a semi-cylinder function for* ***PP****[Public].*

Proof of Lemma 3. Assume that $\sum e$ *is total. If Pe is fixed and unassigned at stage H(e) then by lemma 2,* $\sum e$ *cannot be a semi-cylinder function for* ***PP****[Public]. Suppose Pe is fixed and assigned to the point x at stage H(e). Then as* $\sum e$ *is total,* $\sum e(x)$ *is defined. Let* $\sum e(x) = y$*. If x=y, then*

$$x \in \mathbf{PP}[\text{Public}] \text{ NOT} \equiv y \in \mathbf{PP}[\text{Public}] - \{x\}$$

Therefore $\sum$e cannot be a semi-cylinder function for **PP**[Public]. If x $\neq$ y, then x and y satisfy the condition R2 of step 2 of the program. Then for some stage m $\geq$ H(e), e will be attacked and Pe will be deleted via step 3a of the program. This is a contradiction as Pe is fixed and assigned at stage H(e). Hence the condition x = y holds. Therefore $\sum$e cannot be a semi-cylinder function for **PP**[Public]. This proves lemma 3 and hence Theorem 2(iii).

2.4 A Note on Multilevel Privacy Functions

In the previous section we assumed that all privacy functions as well as the privacy constraints were at system-low. That is, the functions and constraints were visible at the Public level. Note that if all inference rules are at system-low then all users could use the same strategies to make inferences and deduce information say at the private level. In reality it may be possible for certain users to use some additional inference strategies in order to make inferences. We assume that those who can view private information may possess some additional strategies. In order to model these additional strategies, we need to introduce the notion of a multilevel privacy function.

A multilevel privacy function is a privacy function, which has different views at different privacy levels. That is, the result of a multilevel privacy function applied to some data will give different values at different levels. Figure 5 illustrates a multilevel privacy function, Inference that can be made at the Public level is shown by unbold lines. At the private level it is possible to make all the inferences that can be made at the public level. Some additional inferences can be made at the private level. These inferences are shown by the bold line.

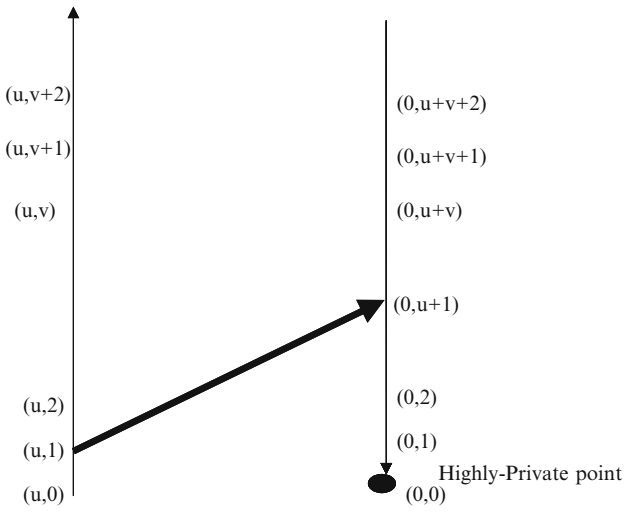


Fig. 5. Multilevel privacy function

If f is the privacy function that is represented by Fig. 5, then the version of f at the public and private levels are denoted by f[Public] and f[Private] respectively. These versions are defined as follows:

$$\begin{aligned} \text{f[Public]}(\text{u}, \text{v}) = &\{(\text{u}, \text{v}+1)\} \text{if } \text{u} \neq 0 \\ &\{(\text{u}, \text{v}-1)\} \text{if } \text{u} = 0 \text{ AND } \text{v} \neq 0 \\ &\phi \text{ (the empty set) if } \text{u} = 0 \text{ AND } \text{v} = 0 \end{aligned}$$

$$\begin{aligned} \text{f[Private]}(\text{u}, \text{v}) = &\{(\text{u}, \text{v}+1)\} \text{ if } \text{u} \neq 0 \text{ AND } \text{v} \neq 1 \\ &\{(\text{u}, \text{v}+1),\ (0, \text{u}+1)\} \text{if } \text{u} \neq 0 \text{ AND } \text{v} = 1 \\ &\{(\text{u}, \text{v}-1)\} \text{if } \text{u} = 0 \text{ AND } \text{v} \neq 0 \\ &\phi \text{ (the empty set) if } \text{u} = 0 \text{ AND } \text{v} = 0 \end{aligned}$$

As a result those at the private level could infer information which the public users cannot infer. For example, consider the privacy function illustrated in Fig. 5. We assume that there is only one privacy constraint, which classifies the point $(0, 0)$ at the Highly private level. It can be seen that:

$$\begin{aligned} \mathbf{PP}[\text{Public}] &= \{(0, \text{v}) : \text{v} \geq 1\} \\ \mathbf{PP}[\text{Private}] &= \{(0, \text{v}) : \text{v} \geq 1\} \cup \{(\text{u}, 0) : \text{u}\rangle 1\} \cup \{(\text{u}, 1) : \text{u}\rangle 1\} \end{aligned}$$

Therefore it is no longer true that $\mathbf{PP}[\text{Private}] \subseteq \mathbf{PP}[\text{Public}]$.

All other results obtained in Sect. 2.3 are valid even if we consider privacy functions to be multilevel. This is because these results are with respect to a single privacy level. (Note that Theorem 1(iv) is with respect to multiple privacy levels).

If we assume that the privacy constraints are themselves assigned different privacy levels, then Theorem 1(iv) cannot be valid. For example, consider the privacy function shown in Fig. 6. In this example there is only one privacy constraint which classifies the point $(0, 0)$ at the highly-private level. Let us assume that the constraint itself is at the private level. This means that the constraint does not apply at the public level. It can be seen that:

$$\begin{aligned} \mathbf{PP}[\text{Public}] &= \text{the Empty Set} \\ \mathbf{PP}[\text{Private}] &= \{(0, \text{v}) : \text{v} \geq 1\} \cup \{(\text{u}, 0) : \text{u}\rangle 1\} \cup \{(\text{u}, 1) : \text{u}\rangle 1\} \end{aligned}$$

Therefore it is no longer true that

$$\mathbf{PP}[\text{Private}] \subseteq \mathbf{PP}[\text{Public}].$$

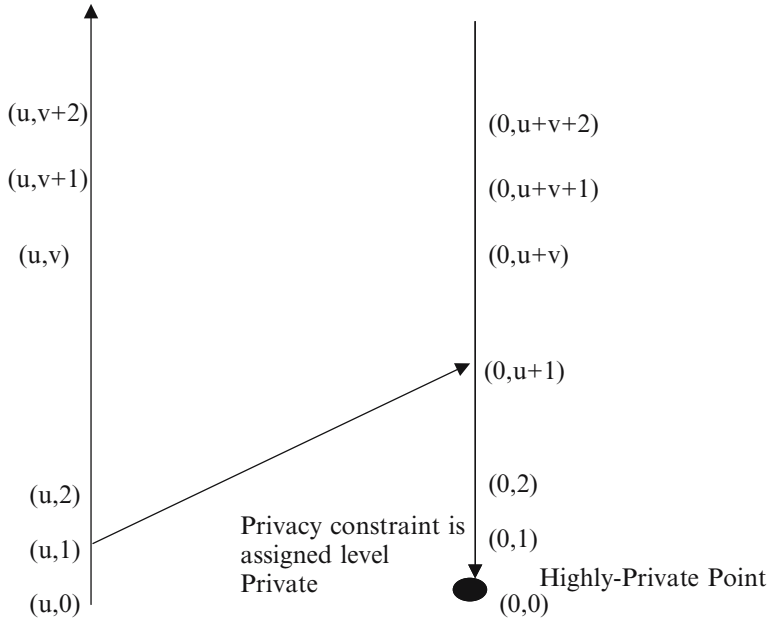


Fig. 6. Multilevel privacy constraints

3 Summary and Directions

In this paper we have defined the notion of privacy function and gave the formulation of the privacy problem in terms of the privacy function. Our formulation of the privacy problem defines it as a decision problem for the set of all non-privacy enhanced database designs. Furthermore, we obtain a different set for each privacy level. We then stated and proved several properties of these sets based on recursive functions and computability theory. Our ideas on privacy enhanced database designs as well as privacy constraints and the development of privacy controllers were discussed in an earlier paper (see [THUR05c]). In this paper we formalize our notions.

Our discussion of complexity is based on recursion theoretic complexity. The next step is to examine the computational complexity. For example while the general privacy problem is unsolvable (as we have shown in this paper), can we find classes of problems that are solvable? What is the computational complexity of the various classes of problems. When we say computational complexity, we mean the space and time complexity as discussed by Blum and others (see [BLUM67]). Can we obtain a characterization of the privacy problem in terms of the computational complexity?

The work in this paper is just the first step toward exploring the foundations of the privacy problem. There is still a lot to be done even for recursion theoretic complexity. The major challenge will also be to obtain a characterization of the privacy problem in terms of computational complexity.

References

[AFSB83I] Air Force Studies Board, "Multilevel Data Management Security," National Academy Press, 1983.

[BARR68] Barron, D., "Recursive Techniques in Programming," MacDonald and Co., London, England, 1968.

[BLUM67] Blum, M., "A Machine Independent Theory of the Complexity of Recursive Functions," Journal of the Association for Cornputin2 Machinery, Vol. 14, 1967, pp. 322–336.

[BRAI74] Brainerd, W., and Landweber, L., "Theory of Computation," Wiley, New York, 1974.

[CALU88] Calude, C., "Theories of Computational Complexity," Annals of Discrete Mathematics Series, ∼35, North Holland, Amsterdam, 1988.

[CHAN82] Chandra, A. and Harel, D., "Structure and Complexity of Relational Queries," Journal of Computer and Systems Sciences, Vol. 25, 1982, pp. 99–127.

[CLEA72] Cleave, 3. P., "Combinatorial Systems I: Cylindrical Problems," Journal of Computer and Systems Sciences, Vol. 6, 1972, pp. 254–266.

[CLEA73] Cleave, J.P., "Degrees of Decision Problems for Computing Machinery," Proceedings of the Ninth Summer School Symposium on Mathematical Logic and Foundations of Computer Science, High Tarras, September 1973, pp. 203–207.

[CLEA75] Cleave, J.P., "Combinatorial Systems II: Non-Cylindrical Problems," Logic Colloquium 1973, North Holland, 1975 (Ed: Rose, H. E and Shepherdson, 3. C.) (Extended version available as unpublished manuscript, University of Bristol, England).

[GALL78] Gallaire, H. and Minker, I., "Logic and Databases," Plenum, New York, 1978.

[HlNK88] Hinke, T., "Inference Aggregation Detection in Database Management Systems," Proceedings of the IEEE Symposium on Security and Privacy, 1988.

[HOPC79] Hopcroft J. and UlIman, J., "Introduction to Automata Theory, Languages and Computation," Addison Wesley, Reading, MA, 1979.

[KOLM65] Kolmogorov, A., "Three Approaches for Defining the Concept of Information Quantity," Problems Peredaci Informacii, 1, 1965. pp. 3–1 1.

[MATC78] Matchety, M. and Young, P.R., "An Introduction to the General Theory of Algorithms," North Holland, Amsterdam, 1978.

[MEND79] Mendleson, E., "Introduction to Mathematical Logic," Van Nostrand, Princeton, NJ, 1979.

[MORG87] Morgenstern, M., "Security' and Inference in Multilevel Database and Knowledge Base Systems," Proceedings of the ACM SIGM CD Conference, San Francisco, CA, May 1987.

[PETE8I] Peter, R., "Recursive Functions in Computer Theory," Ellis Horwood, Chichester, England, 1981.

[P0ST43] Post, E.L., "Formal Reductions of the General Combinatorial Decision Problem," American Journal of Mathematics, Vol. 65, 1943, pp. 197–215.

[P0ST44] Post, E.L., "Recursively Enumerable Sets of Positive Integers and their Decision Problems," Bulletin of the American Mathematical Society, Vol. 50, 1944, pp. 284–316.

[ROGE67] Rogers, H., Jr., "Theory of Recursive Functions and Effective Computability," McGraw-Hill, New York, 1967.

[THUR82] Thuraisingham, M.B., "Representation of One-One Degrees by Decision Problems," Journal of Computer and Systems Sciences, Vol. 24, 1982, pp. 373–377.

[THUR83] Thuraisingham, M.B., "Cylindrical Decision Problems," Notre Dame Journal of Formal Logic, Vol. 24, No. 2, 1983, pp. 188–198.

[THUR86] Thuraisingham, M.B., "The Concept of n-Cylinder and Its Application," Zeitschrift fur Mathematische Logik und Grundlagen der Mathematik. Vol. 32, 1986, pp. 211–219.

[THUR87] Thuraisingham, M.B., "Security Checking in Relational Database Management Systems Augmented with Inference Engines," Computers and Security, Vol. 6, No. 6, December 1987.

[THUR91] Thuraisingham, B., "Recursion Theoretic Properties of the Inference Problem, IEEE CIPHER, Winter, 1991.

[THUR03a] Thuraisingham, B., "Data Mining, National Security and Privacy," SIGKDD Explorations, 2003.

[THUR03b] Thuraisingham, B., "Web Data Mining: Technologies and Their Applications in Business Intelligence and Counter-terrorism," CRC Press, FL, 2003.

[THUR05a] Thuraisingham, B., "Database and Applications Security: Integrating Data Management and Information Security," CRC Press, 2005.

[THUR05b] Thuraisingham, B., "Privacy Preserving Data Mining: Developments and Directions," Accepted in the Journal of Database Management, 2005.

[THUR05c] Thuraisingham, B., "Privacy Constraint Processing in a Privacy Enhanced Database Management System," Accepted for Publication in Data and Knowledge Engineering Journal, 2005.

[THUR06] Thuraisingham, B., Semantic Data Model for Privacy Contol, UTD Technical Report, 2006.

[YOUN63] Young, P.R., "On Semicylinders, Splinters and Bounded Truth Table Reducibility," Transactions of the American Mathematical Society, Vol. 115, 1963, pp. 329–339.

[YOUN66] Young, P.R., "Linear Orderings Under One-One Reducibility," Journal of Symbolic Logic, Vol. 31, pp. 70–85.

Ensembles of Least Squares Classifiers with Randomized Kernels

Kari Torkkola[1] and Eugene Tuv[2]

[1] Motorola, Intelligent Systems Lab, Tempe, AZ, USA
kari.torkkola@motorola.com
[2] Intel, Analysis and Control Technology, Chandler, AZ, USA
eugene.tuv@intel.com

Summary. For the recent NIPS-2003 feature selection challenge we studied ensembles of regularized least squares classifiers (RLSC). We showed that stochastic ensembles of simple least squares kernel classifiers give the same level of accuracy as the best single RLSC. Results achieved were ranked among the best at the challenge. We also showed that performance of a single RLSC is much more sensitive to the choice of kernel width than that of an ensemble. As a continuation of this work we demonstrate that stochastic ensembles of least squares classifiers with randomized kernel widths and OOB-post-processing often outperform the best single RLSC, and require practically no parameter tuning. We used the same set of very high dimensional classification problems presented at the NIPS challenge. Fast exploratory Random Forests were applied for variable filtering first.

1 Introduction

Regularized least-squares regression and classification dates back to the work of Tikhonov and Arsenin [17], and has been re-advocated and revived recently by Poggio, Smale and others [6, 13–15]. Regularized Least Squares Classifier (RLSC) is an old combination of quadratic loss function combined with regularization in reproducing kernel Hilbert space, leading to a solution of a simple linear system. In many cases in the work cited above, this simple RLSC appears to equal or exceed the performance of support vector machines and other modern developments in machine learning.

The combination of RLSC with Gaussian kernels and the usual choice of spherical covariances gives an equal weight to every component of the feature vector. This poses a problem if a large proportion of the features consists of noise. With the datasets of the challenge this is exactly the case. In order to succeed in these circumstances, noise variables need to be removed or weighted down. We apply *ensemble-based variable filtering* to remove noise variables. A Random Forest (RF) is trained for the classification task, and an importance measure for each variable is derived from the forest [4]. Only highest ranking

K. Torkkola and E. Tuv: *Ensembles of Least Squares Classifiers with Randomized Kernels*, Studies in Computational Intelligence (SCI) **118**, 375–386 (2008)
www.springerlink.com

variables are then passed to RLSC. We chose Random Forests (RF) for this task for several reasons. RF can handle huge numbers of variables easily and global relative variable importance score is derived as a by-product of the forest construction with no extra computation involved.

In this chapter we study empirically how a stochastic ensemble of RLSCs with random kernel widths compares to a single optimized RLSC. Our motivation to do this is the well known fact that ensembles of simple weak learners are known to produce stable models that often significantly outperform an optimally tuned single base learner [3, 4, 9]. Another motivating factor is the elimination of the kernel width and regularization parameter selection procedures altogether. A further advantage of ensembles is the possibility of parallelization. Using much smaller sample sizes to train each expert of an ensemble may be faster than training a single learner using a huge data set.

For an ensemble to be effective, the individual experts need to have low bias and the errors they make should be uncorrelated [2,4]. Using no regularization with LSC reduces the bias of the learner making it a good candidate for ensemble methods. Diversity of the learners can be accomplished by training base learners using independent random samples of the training data and by using random kernel widths. The latter is the main topic of this chapter.

The structure of this chapter is as follows. We begin by briefly describing the RLSC, the theory behind it, and its connections to support vector machines. We discuss ensembles, especially ensembles of LSCs and the interplay of regularization and bias in ensembles. The scheme for variable filtering using ensembles of trees is presented, after which we describe experimentation with the NIPS2003 feature selection challenge data sets. We discuss our findings regarding ensembles of random kernel width LSCs, and conclude by touching upon some possible future directions.

2 Regularized Least-Squares Classification (RLSC)

In supervised learning the training data $(x_i, y_i)_{i=1}^m$ is used to construct a function $f : X \to Y$ that predicts or generalizes well. To measure goodness of the learned function $f(x)$ a loss function $L(f(x), y)$ is needed. Some commonly used loss functions for regression are as follows:

- Square loss or L_2: $L(f(x), y) = (f(x) - y)^2$ (the most common)
- Absolute value, or L_1 loss: $L(f(x), y) = |f(x) - y|$
- Vapnik's ϵ-insensitive loss: $L(f(x), y) = (|f(x) - y| - \epsilon)_+$
- Huber's loss function :
  $\begin{cases} |y - f(x)|^2, & \text{for } |f(x) - y| \le \delta \\ \delta(|y - f(x)| - \delta/2), & \text{otherwise} \end{cases}$

Examples of loss functions for classification are:

- Misclassification: $L(f(x), y) = I(sign(f(x)) \neq y)$
- Exponential (Adaboost): $L(f(x), y) = \exp(-yf(x))$

- Hinge loss (implicitly introduced by Vapnik) in binary SVM classification: $L(f(x), y) = (1 - yf(x))_+$
- Binomial deviance: $L(f(x), y) = \log(1 + \exp(-2yf(x)))$
- Squared error: $L(f(x), y) = (1 - yf(x))^2$

Given a loss function, the goal of learning is to find an approximation function $f(x)$ that minimizes the expected risk, or the generalization error

$$E_{P(x,y)} L(f(x), y) \tag{1}$$

where P(x,y) is the unknown joint distribution of future observations (x,y).

Given a finite sample from the (X,Y) domain this problem is ill-posed. The regularization approach championed by Poggio and rooted in Tikhonov regularization theory [17] restores well-posedness (existence, uniqueness, and stability) by restricting the hypothesis space, the functional space of possible solutions:

$$\hat{f} = \underset{f \in H}{\operatorname{argmin}} \frac{1}{m} \sum_{i=1}^{m} L(f(x_i), y_i) + \gamma \|f\|_K^2 \tag{2}$$

The hypothesis space H here is a Reproducing Kernel Hilbert Space (RKHS) defined by kernel K, and γ is a positive regularization parameter.

The mathematical foundations for this framework as well as a key algorithm to solve (2) are derived elegantly by Poggio and Smale [14] for the quadratic loss function. The algorithm can be summarized as follows:

1. Start with the data $(x_i, y_i)_{i=1}^m$.
2. Choose a symmetric , positive definite kernel, such as
$$K(x, x') = \exp(-\frac{||x - x'||^2}{2\sigma^2}). \tag{3}$$
3. Set
$$f(x) = \sum_{i=1}^{m} c_i K(x_i, x), \tag{4}$$
where $\mathbf{c}$ is a solution to
$$(m\gamma \mathbf{I} + \mathbf{K})\mathbf{c} = \mathbf{y}, \tag{5}$$
which represents well-posed ridge regression model [12].

The generalization ability of this solution, as well choosing the regularization parameter γ were studied in [6, 7]. Thus, using the square loss function with regularization leads to solving a simple well defined linear problem. This is the core of RLSC. The solution is a linear kernel expansion of the same form as the one given by support vector machines (SVM). Note also that the SVM formulation naturally fits in the regularization framework (2). Inserting the SVM hinge loss function $L(f(x), y) = (1 - yf(x))_+$ to (2) leads to a solution that is sparse in coefficients $\mathbf{c}$, but it introduces the cost of having to solve a quadratic optimization problem instead of the linear solution of the RLSC.

RLSC with the square loss function, which is more common for regression, has also proven to be very effective in binary classification problems [15, 16].

3 Model Averaging and Regularization

We discuss now what properties are required for the base learners to make an effective ensemble, and how those properties can be attained with least squares classifiers (LSC).

3.1 Stability

Generalization ability of a learned function is closely related to its stability. Stability of the solution could be loosely defined as a continuous dependence on the data. A stable solution changes very little for small changes in the data. A comprehensive treatment of this connection can be found in [2].

Furthermore, it is well known that bagging (bootstrap aggregation) can dramatically reduce variance of unstable learners providing some regularization effect [3]. Bagged ensembles do not overfit. The key to the performance is a low bias of the base learner, and a low correlation between base learners.

Evgeniou experimented with ensembles of SVMs [8]. He used a few datasets from UCI tuning all parameters separately for both a single SVM and for an ensemble of SVMs to achieve the best performance. He found that both perform similarly. However, he also found that generalization bounds for ensembles are tighter than for a single machine.

Poggio et al. studied the relationship between stability and bagging [13]. They showed that there is a bagging scheme, where each expert is trained on a disjoint subset of the training data, providing strong stability to ensembles of non-strongly stable experts, and therefore providing the same order of convergence for the generalization error as Tikhonov regularization. Thus, at least asymptotically, bagging strongly stable experts would not improve generalization ability of the individual member.

3.2 Ensembles of RLSCs

An ensemble should thus have diverse experts with low bias. For RLSC, the bias is controlled by the regularization parameter and by the σ in case of a Gaussian kernel. Instead of bootstrap sampling from training data which imposes a fixed sampling strategy, we found that often much smaller sample sizes of the order of 30–50% of the data set size improve performance. A further source of diversity is introduced by each expert having a different random kernel width.

Combining the outputs of the experts in an ensemble can be done in several ways. The simplest alternative is majority voting over the outputs of the experts. In binary classification this is equivalent to averaging the discretized $(+1,-1)$ predictions of the experts. In our experiments this performed better than averaging the actual numeric expert outputs before applying their decision function (sign).

A well known avenue to improve the accuracy of an ensemble is to replace the simple averaging of individual experts by a weighting scheme. Instead of giving equal weight to each expert, the outputs of more reliable experts are weighted up (even for a classification problem). Linear regression can be applied to learn these weights.

To avoid overfitting, the training material to learn this regression should be produced by passing only such samples through an expert, that did not participate in construction of the particular expert. Typically this is done by using a separate validation data set. Since some of the datasets used were very small in size, it was not useful to split the training sets further for this purpose. Instead, since each expert is constructed only from a fraction of the training data set, the rest of the data is available as "out-of-bag samples" (OOB).

We experimented with two schemes to construct the training data matrix in order to learn the weights. The matrix consists of outputs of each individual member of the ensemble, Each row corresponds to a data sample in the training set, and each column corresponds to one expert of the ensemble. Since each expert populates the column only with OOB-samples, the empty spaces corresponding to the training data of the expert can be filled in either with zeroes, or with the expert outputs by passing the training data through the expert. The latter is optimistically biased, and the former is biased toward zero (the "don't know" condition). In the latter case we also up-weighted the entries by the reciprocal of the fraction of missing entries in order to compensate for the inner product of the regression coefficients with the entries to sum to either plus or minus one.

Since expert outputs are correlated (although the aim is to have uncorrelated experts), PCA regression can be applied to reduce the number of regression coefficients. Partial Least Squares regression could also be used instead of PCA regression. We ended up using PCA regression in the final experiments.

4 Variable Filtering with Tree-Based Ensembles

Because the data sets contained unknown irrelevant variables (50–90% of the variables were noise), we noticed significant improvement in accuracy when only a small (but important) fraction of the original variables was used in the kernel construction.

We used fast exploratory tree-based models for variable filtering. One of many important properties of CART [5] is its embedded ability to select important variables during tree construction (greedy recursive partition, where impurity reduction is maximized at every step), and therefore resistance to noise. Variable importance then can be defined as

$$M(x_m, T) = \sum_{t \in T} \Delta I(x_m, t) \tag{6}$$

where $\Delta I(x_m, t)$ is the decrease in impurity due to an actual or potential split on variable x_m at a node t of the optimally pruned tree T. The sum in (6) is taken over all internal tree nodes where x_m was a primary splitter or a surrogate variable. Consequently, no additional effort is needed for its calculation.

Two recent advances in tree ensembles – Multivariate Adaptive Regression Trees (MART) [10, 11] and Random Forests (RF) [4] inherit all nice properties of a single tree, and provide more reliable estimate of this value, as the importance measure is averaged over the trees in the ensemble

$$M(x_i) = \frac{1}{M} \sum_{m=0}^{M} M(x_i, T_m). \tag{7}$$

MART builds shallow trees using all variables, and hence, can handle large datasets with moderate number of variables. RF builds maximal trees but chooses a small random subset of variables at every split, and easily handles thousands of variables in datasets of moderate size. For datasets massive in both dimensions a hybrid scheme with shallow trees and dynamic variable selection has been shown to have at least the same accuracy but to be much faster than either MART or RF [1].

Note that the index of variable importance defined in the above measures is the global contribution of a variable to the learned model. It is not just a univariate response-predictor relationship.

For the NIPS2003 challenge data we used RF to select important variables. A forest was grown using the training data until there was no improvement in the generalization error. Typically, this limit was around 100 trees. As an individual tree is grown, a random sample out of the N variables is drawn, out of which the best split is chosen (instead of considering all of the variables). The size of this sample was typically $\sqrt{N}$.

5 Experiments with the NIPS2003 Feature Selection Challenge Data Sets

The purpose of the NIPS2003 challenge in feature selection was to find feature selection algorithms that significantly outperform methods using all features, on all five benchmark datasets. The datasets and their (diverse) characteristics are listed in Table 1.

Of these data sets, only Dorothea was highly unbalanced with approximately 12% of samples in one class, and 88% in the other. The rest of the sets had an approximately balanced class distribution. All tasks are two-class classification problems.

Table 1. NIPS2003 feature selection challenge data

Dataset	Size (MB)	Type	Number of variables	Training examples	Validation examples
Arcene	8.7	Dense	10,000	100	100
Gisette	22.5	Dense	5,000	6,000	1,000
Dexter	0.9	Sparse integer	20,000	300	300
Dorothea	4.7	Sparse binary	100,000	800	350
Madelon	2.9	Dense	500	2,000	600

Table 2. Comparison of no variable selection to variable selection

Data set	Variables	Error rate using all variables	Selected variables	Error rate using selected variables
Madelon	500	0.254	19	0.093
Dexter	20,000	0.324	109	0.074

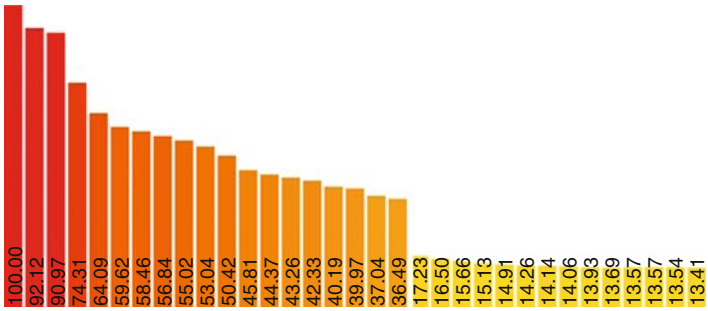


Fig. 1. The importance of the top 33 out of 500 variables of Madelon derived from a training set of 2,000 cases in 500 trees. Variable importance has a clear cut-off point at 19 variables

5.1 Variable Selection Experiments

Initial experimentation was performed to determine whether variable selection was necessary at all. We trained ensembles of LSCs (ELSC) for two of the data sets. Results are given in Table 2 as the averages of tenfold cross validation.

These results clearly indicated that RLSC/ELSC is sensitive to noise variables in data, and that variable selection based on importances derived from Random Forests works well.

For the rest of the experiments, we adopted the following variable selection procedure. Variables are ranked by a random forest as described in Sect. 4. If there are significant cut-off points in the ranked importance, the variable set before the cut-off point is selected. Figure 1 shows a clear example of such a cut-off point.

Table 3. Variable selection, standardization, and variable weighting decisions

Data set	Original variables	Selected variables	Selection method	Standardize?	Weighting?
Madelon	500	19	RF	Yes	No
Dexter	20,000	500	MI	Yes	By MI
Arcene	10,000	10,000	None	No	No
Gisette	5,000	307	RF	No	No
Dorothea	100,000	284	RF	No	No

For each data set, the smallest possible variable set as indicated by a cut-off point was tried first. If the results were unsatisfactory, the next cut-off point was searched, and so on, until satisfactory results were obtained. The maximum number of variables considered was about 500. Full cross-validation was thus not done over the whole possible range of the number of selected variables.

Variable set was thereafter fixed to the one that produced the smallest cross-validation error in the classification experiments, with two exceptions: Contrary to other data sets, on arcene the error rate using the validation set did not follow cross-validation error but was the smallest when all variables were used. Arcene is evidently such a small data set that variable selection and classifier training both using the 100 training samples, will overfit. The second exception is dexter, which gave the best results using 500 variables ranked by maximum mutual information (MI) with the class labels [18].

At this point we also experimented with variable standardization and weighting variables. Weighting here denotes multiplying variables by their importance scores given by the RF (or MI). Due to lack of space these experiments are not tabulated, but the decisions are summarized in Table 3.

5.2 Classification Experiments with ELSCs using Random Kernels

An individual RLSC has two parameters that need to be determined by cross-validation. These are the kernel width σ^2 and the regularization parameter γ. For a single RLSC, regularization is critical in order not to overfit. The choice of the parameters needs to be made by cross-validation, and appears to be very data dependent. This leads to optimization in a two-dimensional parameter space using cross-validation. As an example, we present this optimization for the Madelon data set in Fig. 2.

An ensemble of stochastic LSCs is less sensitive to kernel width, does not require search for the regularization parameter, is not sensitive to the ensemble size (once it is large enough), and is not very sensitive to the fraction

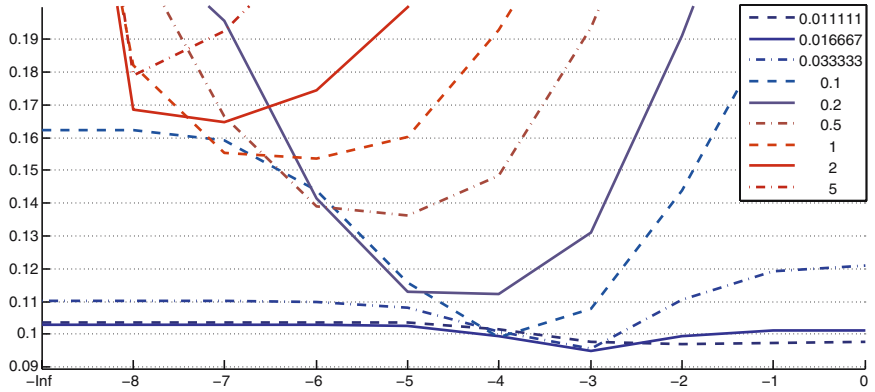


Fig. 2. Single RLSC: Cross-validation experimentation in order to find the optimal combination of kernel width and regularization parameter for madelon data set. Vertical axis is the tenfold cross-validation error rate on training data, horizontal axis is $\log_{10}(\gamma)$, and each curve corresponds to a specific kernel width as $w_j d_{av}^2$, where w_j are the numbers shown in the legend, and d_{av}^2 is the average squared distance between data samples. The optimal combination for this data set is $\gamma = 10^{-3}$ and $\sigma^2 = 0.016667 d_{av}^2$

of data sampled to train each LSC [19]. Our motivation in using random kernels, or more precisely, random kernel widths, was to get rid of all these tunable parameters in ensemble construction without sacrificing any of the generalization performance.

Naturally, the kernel width cannot be completely random, but in a reasonable range, which is determined by the data. We sampled the σ^2 uniformly in the range of $[d_{min}^2/4, d_{med}^2]$, where d_{med} is the median distance between samples, and d_{min} is the minimum distance between samples. This was found to be a reasonable range for all the five diverse challenge datasets.

The ensemble size was fixed to 200, and the fraction of training data to train each LSC was fixed to 0.5. These were near-optimal values for ELSCs according to our earlier experiments [19].

Ensemble output combination was done using PCA-regression. We experimented also with plain regression using a mixture of training/OOB samples or just the OOB-samples, but the differences were insignificant.

We present the final classification error rates in Table 4. Even though there is no significant difference in validation error rates between using a single RLSC with optimized parameters, an ELSCs with optimized parameters, or an ELSC with random kernel width, the fact that the latter can be trained without any necessary parameter/model selection makes it a desirable alternative.

Table 4. Error rates using the separate validation data set after optimizing σ^2 and γ for a single RLSC, and σ^2 and the fraction of data sampled for each LSC in an ensemble of 200 classifiers

Data set	Optimized RLSC	Optimized ELSC	Random kernel ELSC
Arcene	0.1331	0.1331	0.1130
Gisette	0.0210	0.0210	0.0200
Dorothea	0.1183	0.1183	0.1140
Madelon	0.0700	0.0667	0.0717
Dexter	0.0633	0.0633	0.0700

Random kernel ELSC required no parameter tuning

6 Future Directions

We describe an approach in this chapter that consists of two disjoint systems, Random Forests for variable selection, and ELSC for the actual classification. Even though the two systems complement each other nicely, RF providing fast embedded variable selection and ELSC providing highly capable base learners to compensate for the lack of smoothness of the trees of an RF, an integrated approach would be desirable. We describe an idea towards such a system.

RF could act as a supervised kernel generator using the pairwise similarities between cases. Breiman defined a coarse similarity measure between two observations for a single tree as one if the observations end up in the same terminal node in a tree, and zero otherwise [4]. For the whole forest, these similarities added up. Similarity with a finer granularity could be defined as the total number of common parent nodes, normalized by the level of the deepest case, and summed up for the ensemble. Minimum number of common parents to define non-zero similarity is another parameter that could be used like width in Gaussian kernels.

Figure 3 illustrates the difference between a Gaussian kernel and the proposed supervised kernel.[1]

An advantage of the method is that it works for any type of data, numeric, categorical, or mixed, even for data with missing values. This is because the base learners of the Random Forest can tolerate these. A further advantage is that explicit variable selection is bypassed altogether. Important variables will become used in the trees of the forest, and they thus participate implicitly in the evaluation of the kernel. New data to be classified must be run through all the trees of the forest in order to record the terminal nodes, but this is an extremely fast operation.

[1] Colormap is the Matlab standard "Jet": dark red to red to yellow to green to blue to dark blue denotes decreasing similarity.

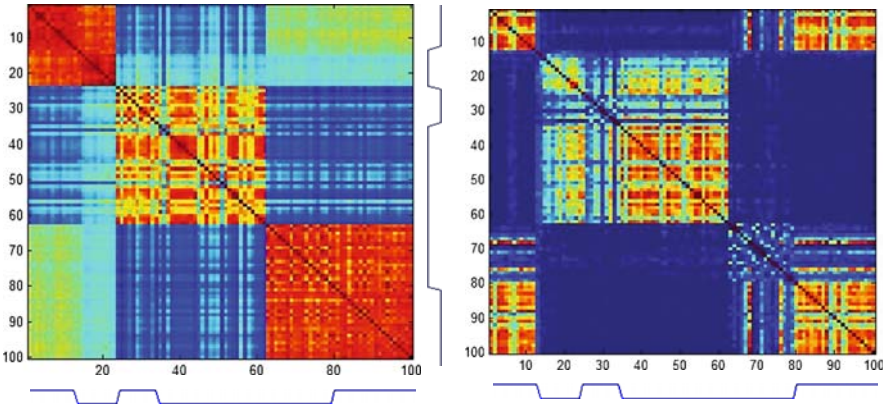


Fig. 3. Gaussian kernel compared to a supervised kernel using the Arcene dataset. *Left side* depicts the 100×100 Gaussian kernel matrix of the data set. The data can be seen to consist of three clusters. Each cluster has samples from both classes. Class identities of samples are depicted as the graphs below and between the kernel matrix images. For ideal classification purposes, the kernel matrix should reflect the similarity within a class and dissimilarity between classes. This can be seen on the *right side* of the figure, where the proposed supervised kernel has split the first cluster (*top left corner*) into the two classes nicely. Splits on the second and third clusters are not that clean but still visible, and much more so than what can be seen in the Gaussian kernel

7 Conclusion

We proposed a relatively straightforward approach to create powerful ensembles of simple least square classifiers with random kernels. We used NIPS2003 feature selection challenge data to evaluate performance of such ensembles. The binary classification data sets considered in the challenge originated in different domains with number of variables ranging from moderate to extremely large and moderate to very small number of observations. We used fast exploratory Random Forests for variable filtering as a preprocessing step. The individual learners were trained on small random sample of data with Gaussian kernel width randomly selected from relatively wide range of values determined only by basic properties of the corresponding dissimilarities matrix. The random sample of data used to build individual learner was relatively small. Modest ensemble size (less than 200) stabilized the generalization error. We used consistent parameter settings for all datasets, and achieved at least the same accuracy as the best single RLSC or an ensemble of LSCs with fixed tuned kernel width. Individual learners were combined through simple OOB post-processing PCA regression.

References

1. A. Borisov, V. Eruhimov, and E. Tuv. Dynamic soft feature selection for tree-based ensembles. In I. Guyon, S. Gunn, M. Nikravesh, and L. Zadeh, editors, *Feature Extraction, Foundations and Applications*. Springer, Berlin Heidelberg New York, 2005
2. O. Bousquet and A. Elisseeff. Algorithmic stability and generalization performance. In *NIPS*, pages 196–202, 2000
3. L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996
4. L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001
5. L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone. *Classification and Regression Trees*. CRC, Boca Raton, FL, 1984
6. F. Cucker and S. Smale. On the mathematial foundations of learning. *Bulletin of the American Mathematical Society*, 89(1):1–49, 2001
7. F. Cucker and S. Smale. Best choices for regularization parameters in learning theory: on the bias-variance problem. *Foundations of Computational Mathematics*, 2(4):413–428, 2003
8. T. Evgeniou. *Learning with Kernel Machine Architectures*. PhD thesis, Massachusetts Institute of Technology, EECS, July 2000
9. Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *European Conference on Computational Learning Theory*, pages 23–37, 1995
10. J.H. Friedman. Greedy function approximation: a gradient boosting machine. Technical report, Department of Statistics, Stanford University, 1999
11. J.H. Friedman. Stochastic gradient boosting. Technical report, Department of Statistics, Stanford University, 1999
12. A. Hoerl and R. Kennard. Ridge regression; biased estimation for nonorthogonal problems. *Technometrics*, 12(3):55–67, 1970
13. T. Poggio, R. Rifkin, S. Mukherjee, and A. Rakhlin. Bagging regularizes. CBCL Paper 214, Massachusetts Institute of Technology, Cambridge, MA, February 2002. AI Memo #2002–2003
14. T. Poggio and S. Smale. The mathematics of learning: Dealing with data. *Notices of the American Mathematical Society (AMS)*, 50(5):537–544, 2003
15. R. Rifkin. *Everything Old is New Again: A Fresh Look at Historical Approaches in Machine Learning*. PhD thesis, MIT, 2002
16. J.A.K. Suykens and J. Vandervalle. Least squares support vector machines. *Neural Processing Letters*, 9(3):293–300, June 1999
17. A.N. Tikhonov and V.Y. Arsenin. *Solutions of Ill-posed Problems*. W.H. Wingston, Washington DC, 1977
18. K. Torkkola. Feature extraction by non-parametric mutual information maximization. *Journal of Machine Learning Research*, 3:1415–1438, March 2003
19. K. Torkkola and E. Tuv. Ensembles of regularized least squares classifiers for high-dimensional problems. In I. Guyon, S. Gunn, M. Nikravesh, and L. Zadeh, editors, *Feature Extraction, Foundations and Applications*. Springer, Berlin Heidelberg New York, 2005

On Pseudo-Statistical Independence in a Contingency Table

Shusaku Tsumoto

Department of Medical Informatics, School of Medicine, Shimane University,
89-1 Enya-cho Izumo 693-8501 Japan
tsumoto@computer.org

Summary. A contingency table summarizes the conditional frequencies of two attributes and shows how these two attributes are dependent on each other with the information on a partition of universe generated by these attributes. Thus, this table can be viewed as a relation between two attributes with respect to information granularity. This chapter focuses on several characteristics of linear and statistical independence in a contingency table from the viewpoint of granular computing, which shows that statistical independence in a contingency table is a special form of linear dependence. The discussions also show that when a contingency table is viewed as a matrix, called a contingency matrix, its rank is equal to 1.0. Thus, the degree of independence, rank plays a very important role in extracting a probabilistic model from a given contingency table. Furthermore, it is found that in some cases, partial rows or columns will satisfy the condition of statistical independence, which can be viewed as a solving process of Diophatine equations.

1 Introduction

Statistical independence between two attributes is a very important concept in data mining and statistics. The definition $P(A, B) = P(A)P(B)$ show that the joint probability of A and B is the product of both probabilities. This gives several useful formula, such as $P(A|B) = P(A)$, $P(B|A) = P(B)$. In a data mining context, these formulae show that these two attributes may not be correlated with each other. Thus, when A or B is a classification target, the other attribute may not play an important role in its classification.

Although independence is a very important concept, it has not been fully and formally investigated as a relation between two attributes.

In this chapter, a statistical independence in a contingency table is focused on from the viewpoint of granular computing.

The first important observation is that a contingency table compares two attributes with respect to information granularity. It is shown from the definition that statistical independence in a contingency table is a special form of linear dependence of two attributes. Especially, when the table is viewed

S. Tsumoto: *On Pseudo-Statistical Independence in a Contingency Table*, Studies in Computational Intelligence (SCI) **118**, 387–403 (2008)
www.springerlink.com

as a matrix, the above discussion shows that the rank of the matrix is equal to 1.0. Also, the results also show that partial statistical independence can be observed.

The second important observation is that matrix algebra is a key point of analysis of this table. A contingency table can be viewed as a matrix and several operations and ideas of matrix theory are introduced into the analysis of the contingency table.

The chapter is organized as follows: Section 2 discusses the characteristics of contingency tables. Section 3 shows the conditions on statistical independence for a 2×2 table. Section 4 gives those for a $2 \times n$ table. Section 5 extends these results into a multiway contingency table. Section 6 discusses statistical independence from matrix theory. Sections 7 and 8 show pseudo-statistical independence. Finally, Sect. 9 concludes this chapter.

2 Contingency Table from Rough Sets

2.1 Rough Sets Notations

In the subsequent sections, the following notations is adopted, which is introduced in [2]. Let U denote a nonempty, finite set called the universe and A denote a nonempty, finite set of attributes, i.e., $a : U \rightarrow V_a$ for $a \in A$, where V_a is called the domain of a, respectively. Then, a decision table is defined as an information system, $\mathcal{A} = (U, A \cup \{D\})$, where $\{D\}$ is a set of given decision attributes. The atomic formulas over $B \subseteq A \cup \{\mathcal{D}\}$ and V are expressions of the form $[a = v]$, called descriptors over B, where $a \in B$ and $v \in V_a$. The set $F(B, V)$ of formulas over B is the least set containing all atomic formulas over B and closed with respect to disjunction, conjunction and negation. For each $f \in F(B, V)$, f_A denote the meaning of f in A, i.e., the set of all objects in U with property f, defined inductively as follows:

1. If f is of the form $[a = v]$ then, $f_A = \{s \in U | a(s) = v\}$
2. $(f \wedge g)_A = f_A \cap g_A$; $(f \vee g)_A = f_A \vee g_A$; $(\neg f)_A = U - f_a$

By using this framework, classification accuracy and coverage, or true positive rate is defined as follows.

Definition 1. *Let R and D denote a formula in $F(B, V)$ and a set of objects whose decision attribute is given as D, respectively. Classification accuracy and coverage(true positive rate) for $R \rightarrow D$ is defined as:*

$$\alpha_R(D) = \frac{|R_A \cap D|}{|R_A|} (= P(D|R)), \text{ and}$$

$$\kappa_R(D) = \frac{|R_A \cap D|}{|D|} (= P(R|D)),$$

where $|A|$ denotes the cardinality of a set A, $\alpha_R(D)$ denotes a classification accuracy of R as to classification of $\mathcal{D}$, and $\kappa_R(D)$ denotes a coverage, or a true positive rate of R to $\mathcal{D}$, respectively.

2.2 Contingency Table (2×2)

From the viewpoint of information systems, a contingency table summarizes the relation between two attributes with respect to frequencies. This viewpoint has already been discussed in [3, 4]. However, this study focuses on more statistical interpretation of this table.

Definition 2. *Let R_1 and R_2 denote binary attributes in an attribute space A. A contingency table is a table of a set of the meaning of the following formulas: $|[R_1 = 0]_A|$,$|[R_1 = 1]_A|$, $|[R_2 = 0]_A|$,$|[R_2 = 1]_A|$, $|[R_1 = 0 \wedge R_2 = 0]_A|$,$|[R_1 = 0 \wedge R_2 = 1]_A|$, $|[R_1 = 1 \wedge R_2 = 0]_A|$,$|[R_1 = 1 \wedge R_2 = 1]_A|$, $|[R_1 = 0 \vee R_1 = 1]_A|(= |U|)$. This table is arranged into the form shown in Table 1, where: $|[R_1 = 0]_A| = x_{11} + x_{21} = x_{\cdot 1}$, $|[R_1 = 1]_A| = x_{12} + x_{22} = x_{\cdot 2}$, $|[R_2 = 0]_A| = x_{11} + x_{12} = x_{1\cdot}$, $|[R_2 = 1]_A| = x_{21} + x_{22} = x_{2\cdot}$, $|[R_1 = 0 \wedge R_2 = 0]_A| = x_{11}$, $|[R_1 = 0 \wedge R_2 = 1]_A| = x_{21}$, $|[R_1 = 1 \wedge R_2 = 0]_A| = x_{12}$, $|[R_1 = 1 \wedge R_2 = 1]_A| = x_{22}$, $|[R_1 = 0 \vee R_1 = 1]_A| = x_{\cdot 1} + x_{\cdot 2} = x_{\cdot\cdot}(= |U|)$.*

From this table, accuracy and coverage for $[R_1 = 0] \rightarrow [R_2 = 0]$ are defined as:

$$\alpha_{[R_1=0]}([R_2 = 0]) = \frac{|[R_1 = 0 \wedge R_2 = 0]_A|}{|[R_1 = 0]_A|} = \frac{x_{11}}{x_{\cdot 1}},$$

and

$$\kappa_{[R_1=0]}([R_2 = 0]) = \frac{|[R_1 = 0 \wedge R_2 = 0]_A|}{|[R_2 = 0]_A|} = \frac{x_{11}}{x_{1\cdot}}.$$

2.3 Contingency Table ($m \times n$)

Two-way contingency table can be extended into a contingency table for multinominal attributes (Table 2).

Definition 3. *Let R_1 and R_2 denote multinominal attributes in an attribute space A which have m and n values. A contingency tables is a table of a set of the meaning of the following formulas: $|[R_1 = A_j]_A|$, $|[R_2 = B_i]_A|$, $|[R_1 = A_j \wedge R_2 = B_i]_A|$, $|[R_1 = A_1 \vee R_1 = A_2 \vee \cdots \vee R_1 = A_m]_A|$,$|[R_2 = B_1 \vee R_2 = A_2 \vee \cdots \vee R_2 = A_n]_A| = |U|$ $(i = 1, 2, 3, \cdots, n$ and $j = 1, 2, 3, \cdots, m)$. This table is arranged into the form shown in Table 1, where: $|[R_1 = A_j]_A| = \sum_{i=1}^{m} x_{1i} = x_{\cdot j}$, $|[R_2 = B_i]_A| = \sum_{j=1}^{n} x_{ji} = x_{i\cdot}$, $|[R_1 = A_j \wedge R_2 = B_i]_A| = x_{ij}$, $|U| = N = x_{\cdot\cdot}$ $(i = 1, 2, 3, \cdots, n$ and $j = 1, 2, 3, \cdots, m)$.*

Table 1. Two way contingency table

	$R_1 = 0$	$R_1 = 1$	
$R_2 = 0$	x_{11}	x_{12}	$x_{1\cdot}$
$R_2 = 1$	x_{21}	x_{22}	$x_{2\cdot}$
	$x_{\cdot 1}$	$x_{\cdot 2}$	$x_{\cdot\cdot}$ $(= \|U\| = N)$

Table 2. Contingency table $(m \times n)$

	A_1	A_2	$\cdots$	A_n	Sum
B_1	x_{11}	x_{12}	$\cdots$	x_{1n}	$x_{1\cdot}$
B_2	x_{21}	x_{22}	$\cdots$	x_{2n}	$x_{2\cdot}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
B_m	x_{m1}	x_{m2}	$\cdots$	x_{mn}	$x_{m\cdot}$
Sum	$x_{\cdot 1}$	$x_{\cdot 2}$	$\cdots$	$x_{\cdot n}$	$x_{\cdot\cdot} = \|U\| = N$

3 Statistical Independence in 2×2 Contingency Table

Let us consider a contingency table shown in Table 1. Statistical independence between R_1 and R_2 gives:

$$
\begin{aligned}
P([R_1 = 0], [R_2 = 0]) &= P([R_1 = 0]) \times P([R_2 = 0]) \\
P([R_1 = 0], [R_2 = 1]) &= P([R_1 = 0]) \times P([R_2 = 1]) \\
P([R_1 = 1], [R_2 = 0]) &= P([R_1 = 1]) \times P([R_2 = 0]) \\
P([R_1 = 1], [R_2 = 1]) &= P([R_1 = 1]) \times P([R_2 = 1])
\end{aligned}
$$

Since each probability is given as a ratio of each cell to N, the above equations are calculated as:

$$
\begin{aligned}
\frac{x_{11}}{N} &= \frac{x_{11} + x_{12}}{N} \times \frac{x_{11} + x_{21}}{N} \\
\frac{x_{12}}{N} &= \frac{x_{11} + x_{12}}{N} \times \frac{x_{12} + x_{22}}{N} \\
\frac{x_{21}}{N} &= \frac{x_{21} + x_{22}}{N} \times \frac{x_{11} + x_{21}}{N} \\
\frac{x_{22}}{N} &= \frac{x_{21} + x_{22}}{N} \times \frac{x_{12} + x_{22}}{N}
\end{aligned}
$$

Since $N = \sum_{i,j} x_{ij}$, the following formula will be obtained from these four formulae.

$$x_{11}x_{22} = x_{12}x_{21} \textit{ or } x_{11}x_{22} - x_{12}x_{21} = 0$$

Thus,

Theorem 1. *If two attributes in a contingency table shown in Table 1 are statistical indepedent, the following equation holds:*

$$x_{11}x_{22} - x_{12}x_{21} = 0 \tag{1}$$

It is notable that the above equation corresponds to the fact that the determinant of a matrix corresponding to this table is equal to 0. Also, when these four values are not equal to 0, the (1) can be transformed into:

$$\frac{x_{11}}{x_{21}} = \frac{x_{12}}{x_{22}}.$$

Let us assume that the above ratio is equal to $C(constant)$. Then, since $x_{11} = Cx_{21}$ and $x_{12} = Cx_{22}$, the following equation is obtained.

$$\frac{x_{11} + x_{12}}{x_{21} + x_{22}} = \frac{C(x_{21} + x_{22})}{x_{21} + x_{22}} = C = \frac{x_{11}}{x_{21}} = \frac{x_{12}}{x_{22}}. \tag{2}$$

This equation also holds when we extend this discussion into a general case. Before getting into it, let us consider a 2×3 contingency table.

4 Statistical Independence in 2×3 Contingency Table

Let us consider a 2×3 contingency table shown in Table 3. Statistical independence between R_1 and R_2 gives:

$$\begin{aligned}
P([R_1 = 0], [R_2 = 0]) &= P([R_1 = 0]) \times P([R_2 = 0]) \\
P([R_1 = 0], [R_2 = 1]) &= P([R_1 = 0]) \times P([R_2 = 1]) \\
P([R_1 = 0], [R_2 = 2]) &= P([R_1 = 0]) \times P([R_2 = 2]) \\
P([R_1 = 1], [R_2 = 0]) &= P([R_1 = 1]) \times P([R_2 = 0]) \\
P([R_1 = 1], [R_2 = 1]) &= P([R_1 = 1]) \times P([R_2 = 1]) \\
P([R_1 = 1], [R_2 = 2]) &= P([R_1 = 1]) \times P([R_2 = 2])
\end{aligned}$$

Since each probability is given as a ratio of each cell to N, the above equations are calculated as:

$$\frac{x_{11}}{N} = \frac{x_{11} + x_{12} + x_{13}}{N} \times \frac{x_{11} + x_{21}}{N} \tag{3}$$

$$\frac{x_{12}}{N} = \frac{x_{11} + x_{12} + x_{13}}{N} \times \frac{x_{12} + x_{22}}{N} \tag{4}$$

Table 3. Contingency table (2×3)

	$R_1 = 0$	$R_1 = 1$	$R_1 = 2$	
$R_2 = 0$	x_{11}	x_{12}	x_{13}	$x_{1\cdot}$
$R_2 = 1$	x_{21}	x_{22}	x_{23}	$x_{2\cdot}$
	$x_{\cdot 1}$	$x_{\cdot 2}$	$x_{\cdots 3}$	$x_{\cdot\cdot}$
			$(= \lvert U \rvert = N)$	

$$\frac{x_{13}}{N} = \frac{x_{11} + x_{12} + x_{13}}{N} \times \frac{x_{13} + x_{23}}{N} \tag{5}$$

$$\frac{x_{21}}{N} = \frac{x_{21} + x_{22} + x_{23}}{N} \times \frac{x_{11} + x_{21}}{N} \tag{6}$$

$$\frac{x_{22}}{N} = \frac{x_{21} + x_{22} + x_{23}}{N} \times \frac{x_{12} + x_{22}}{N} \tag{7}$$

$$\frac{x_{23}}{N} = \frac{x_{21} + x_{22} + x_{23}}{N} \times \frac{x_{13} + x_{23}}{N} \tag{8}$$

From (3) and (6),

$$\frac{x_{11}}{x_{21}} = \frac{x_{11} + x_{12} + x_{13}}{x_{21} + x_{22} + x_{23}}$$

In the same way, the following equation will be obtained:

$$\frac{x_{11}}{x_{21}} = \frac{x_{12}}{x_{22}} = \frac{x_{13}}{x_{23}} = \frac{x_{11} + x_{12} + x_{13}}{x_{21} + x_{22} + x_{23}} \tag{9}$$

Thus, we obtain the following theorem:

Theorem 2. *If two attributes in a contingency table shown in Table 3 are statistical indepedent, the following equations hold:*

$$\begin{aligned} x_{11}x_{22} - x_{12}x_{21} &= x_{12}x_{23} - x_{13}x_{22} \\ &= x_{13}x_{21} - x_{11}x_{23} = 0 \end{aligned} \tag{10}$$

It is notable that this discussion can be easily extended into a $2 \times n$ contingency table where $n > 3$. The important (9) will be extended into

$$\begin{aligned} \frac{x_{11}}{x_{21}} &= \frac{x_{12}}{x_{22}} = \cdots = \frac{x_{1n}}{x_{2n}} \\ &= \frac{x_{11} + x_{12} + \cdots + x_{1n}}{x_{21} + x_{22} + \cdots + x_{2n}} = \frac{\sum_{k=1}^{n} x_{1k}}{\sum_{k=1}^{n} x_{2k}} \end{aligned} \tag{11}$$

Thus,

Theorem 3. *If two attributes in a $2 \times k$ contingency table ($k = 2, \cdots, n$) are statistical indepedent, the following equations hold:*

$$\begin{aligned} x_{11}x_{22} - x_{12}x_{21} &= x_{12}x_{23} - x_{13}x_{22} = \cdots \\ &= x_{1n}x_{21} - x_{11}x_{n3} = 0 \end{aligned} \tag{12}$$

It is also notable that this equation is the same as the equation on collinearity of projective geometry [1].

5 Statistical Independence in $m \times n$ Contingency Table

Let us consider a $m \times n$ contingency table shown in Table 2. Statistical independence of R_1 and R_2 gives the following formulae:

$$P([R_1 = A_i, R_2 = B_j]) = P([R_1 = A_i])P([R_2 = B_j]) \\ (i = 1, \cdots, m, j = 1, \cdots, n).$$

According to the definition of the table,

$$\frac{x_{ij}}{N} = \frac{\sum_{k=1}^{n} x_{ik}}{N} \times \frac{\sum_{l=1}^{m} x_{lj}}{N}. \tag{13}$$

Thus, we have obtained:

$$x_{ij} = \frac{\sum_{k=1}^{n} x_{ik} \times \sum_{l=1}^{m} x_{lj}}{N}. \tag{14}$$

Thus, for a fixed j,

$$\frac{x_{i_a j}}{x_{i_b j}} = \frac{\sum_{k=1}^{n} x_{i_a k}}{\sum_{k=1}^{n} x_{i_b k}}$$

In the same way, for a fixed i,

$$\frac{x_{i j_a}}{x_{i j_b}} = \frac{\sum_{l=1}^{m} x_{l j_a}}{\sum_{l=1}^{m} x_{l j_b}}$$

Since this relation will hold for any j, the following equation is obtained:

$$\frac{x_{i_a 1}}{x_{i_b 1}} = \frac{x_{i_a 2}}{x_{i_b 2}} \cdots = \frac{x_{i_a n}}{x_{i_b n}} = \frac{\sum_{k=1}^{n} x_{i_a k}}{\sum_{k=1}^{n} x_{i_b k}}. \tag{15}$$

Since the right hand side of the above equation will be constant, thus all the ratios are constant. Thus,

Theorem 4. *If two attributes in a contingency table shown in Table 2 are statistical indepedent, the following equations hold:*

$$\frac{x_{i_a 1}}{x_{i_b 1}} = \frac{x_{i_a 2}}{x_{i_b 2}} \cdots = \frac{x_{i_a n}}{x_{i_b n}} = const. \tag{16}$$

for all rows: i_a and i_b ($i_a, i_b = 1, 2, \cdots, m$).

6 Contingency Matrix

The meaning of the above discussions will become much clearer when we view a contingency table as a matrix.

Definition 4. *A corresponding matrix $C_{T_{a,b}}$ is defined as a matrix the element of which are equal to the value of the corresponding contingency table $T_{a,b}$ of two attributes a and b, except for marginal values.*

Definition 5. *The rank of a table is defined as the rank of its corresponding matrix. The maximum value of the rank is equal to the size of (square) matrix, denoted by r.*

The contingency matrix of Table 2($T(R_1, R_2)$) is defined as $C_{T_{R_1,R_2}}$ as below:

$$\begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{pmatrix}$$

6.1 Independence of 2 × 2 Contingency Table

The results in Sect. 3 corresponds to the degree of independence in matrix theory. Let us assume that a contingency table is given as Table 1. Then the corresponding matrix ($C_{T_{R_1,R_2}}$) is given as:

$$\begin{pmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{pmatrix},$$

Then,

Proposition 1. *The determinant of $det(C_{T_{R_1,R_2}})$ is equal to $x_{11}x_{22} - x_{12}x_{21}$.*

Proposition 2. *The rank will be:*

$$rank = \begin{cases} 2, & if\ det(C_{T_{R_1,R_2}}) \neq 0 \\ 1, & if\ det(C_{T_{R_1,R_2}}) = 0 \end{cases}$$

From Theorem 1,

Theorem 5. *If the rank of the corresponding matrix of a 2×2 contingency table is 1, then two attributes in a given contingency table are statistically independent. Thus,*

$$rank = \begin{cases} 2, & dependent \\ 1, & statistical\ independent \end{cases}$$

This discussion can be extended into $2 \times n$ tables. According to Theorem 3, the following theorem is obtained.

Theorem 6. *If the rank of the corresponding matrix of a $2 \times n$ contingency table is 1, then two attributes in a given contingency table are statistically independent. Thus,*

$$rank = \begin{cases} 2, & dependent \\ 1, & statistical\ independent \end{cases}$$

6.2 Independence of 3×3 Contingency Table

When the number of rows and columns are larger than 3, then the situation is a little changed. It is easy to see that the rank for statistical independence of a $m \times n$ contingency table is equal 1.0 as shown in Theorem 4. Also, when the rank is equal to $\min(m, n)$, two attributes are dependent.

Then, what kind of structure will a contingency matrix have when the rank is larger than 1,0 and smaller than $\min(m, n) - 1$? For illustration, let us consider the following 3×3 contingency table.

Example 1. Let us consider the following corresponding matrix:

$$A = \begin{pmatrix} 1\ 2\ 3 \\ 4\ 5\ 6 \\ 7\ 8\ 9 \end{pmatrix}.$$

The determinant of A is:

$$\begin{aligned} det(A) &= 1 \times (-1)^{1+1} det \begin{pmatrix} 5\ 6 \\ 8\ 9 \end{pmatrix} \\ &\quad +2 \times (-1)^{1+2} det \begin{pmatrix} 4\ 6 \\ 7\ 9 \end{pmatrix} \\ &\quad +3 \times (-1)^{1+3} det \begin{pmatrix} 4\ 5 \\ 7\ 8 \end{pmatrix} \\ &= 1 \times (-3) + 2 \times 6 + 3 \times (-3) = 0 \end{aligned}$$

Thus, the rank of A is smaller than 2. On the other hand, since $(123) \neq k(456)$ and $(123) \neq k(789)$, the rank of A is not equal to 1.0 Thus, the rank of A is equal to 2.0. Actually, one of three rows can be represented by the other two rows. For example,

$$(4\ 5\ 6) = \frac{1}{2}\{(1\ 2\ 3) + (7\ 8\ 9)\}.$$

Therefore, in this case, we can say that two of three pairs of one attribute are dependent to the other attribute, but one pair is statistically independent of the other attribute with respect to the linear combination of two pairs. It is easy to see that this case includes the cases when two pairs are statistically independent of the other attribute, but the table becomes statistically dependent with the other attribute.

In other words, the corresponding matrix is a mixture of statistical dependence and independence. We call this case *contextual independent*. From this illustration, the following theorem is obtained:

Theorem 7. *If the rank of the corresponding matrix of a* 3×3 *contingency table is 1, then two attributes in a given contingency table are statistically independent. Thus,*

$$rank = \begin{cases} 3, & dependent \\ 2, & contextual\ independent \\ 1, & statistical\ independent \end{cases}$$

It is easy to see that this discussion can be extended into $3 \times n$ contingency tables.

6.3 Independence of $m \times n$ Contingency Table

Finally, the relation between rank and independence in a multiway contingency table is obtained from Theorem 4.

Theorem 8. *Let the corresponding matrix of a given contingency table be a $m \times n$ matrix. If the rank of the corresponding matrix is 1, then two attributes in a given contingency table are statistically independent. If the rank of the corresponding matrix is* $\min(m, n)$ *, then two attributes in a given contingency table are dependent. Otherwise, two attributes are contextual dependent, which means that several conditional probabilities can be represented by a linear combination of conditional probabilities. Thus,*

$$rank = \begin{cases} \min(m, n) & dependent \\ 2, \cdots, & \\ \quad \min(m, n) - 1 & contextual\ independent \\ 1 & statistical\ independent \end{cases}$$

7 Pseudo-Statistical Independence: Example

The next step is to investigate the characteristics of linear independence in a contingency matrix. In other words, a $m \times n$ contingency table whose rank is not equal to $\min(m, n)$. Since two-way matrix (2×2) gives a simple equation whose rank is equal to 1 or 2, let us start our discussion from 3×3-matrix, whose rank is equal to 2, first.

7.1 Contingency Table (3×3, Rank: 2)

Let $M(m, n)$ denote a contingency matrix whose row and column are equal to m and n, respectively. Then, a three-way contingency table is defined as:

$$M(3, 3) = \begin{pmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{pmatrix}$$

When its rank is equal to 2, it can be assumed that the third row is represented by the first and second row:

$$(x_{31}\ x_{32}\ x_{33}) = p(x_{11}\ x_{12}\ x_{13}) + q(x_{21}\ x_{22}\ x_{23}) \tag{17}$$

Then, we can consider the similar process in Sect. 5 (13). In other words, we can check the difference defined below.

$$\Delta(i,j) = N \times x_{ij} - \sum_{k=1}^{n} x_{ik} \times \sum_{l=1}^{m} x_{lj}. \tag{18}$$

Then, the following three types of equations are obtained by simple calculation.

$$\Delta(1,j) = (1+q)\left\{ x_{1j} \sum_{k=1}^{3} x_{2k} - x_{2j} \sum_{k=1}^{3} x_{1k} \right\}$$

$$\Delta(2,j) = (1+p)\left\{ x_{2j} \sum_{k=1}^{3} x_{1k} - x_{1j} \sum_{k=1}^{3} x_{2k} \right\}$$

$$\Delta(3,j) = (p-q)\left\{ x_{1j} \sum_{k=1}^{3} x_{2k} - x_{2j} \sum_{k=1}^{3} x_{1k} \right\}$$

According to Theorem 4, if $M(3,3)$ is not statistically independent, the formula: $x_{1j} \sum_{k=1}^{3} x_{2k} - x_{2j} \sum_{k=1}^{3} x_{1k}$ is not equal to 1.0. Thus, the following theorem is obtained.

Theorem 9. *The third row represented by a linear combination of first and second rows will satisfy the condition of statistical independence if and only if* $p = q$.

We call the above property *pseudo-statistical independence*. This means that if the third column satisfies the following constraint:

$$(x_{31}\ x_{32}\ x_{33}) = (x_{11}\ x_{12}\ x_{13}) + (x_{21}\ x_{22}\ x_{23}),$$

the third column will satisfy the condition of statistical independence. In other words, when we merge the first and second row and construct a 2×3 contingency table, it will become statistical independent. For example,

$$D = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 10 & 14 & 18 \end{pmatrix}$$

can be transformed into

$$D' = \begin{pmatrix} 5 & 7 & 9 \\ 10 & 14 & 18 \end{pmatrix},$$

where D' is statistically independent. Conversely, if D' is provided, it can be decomposed into D. It is notable that the decomposition cannot be uniquely determined. It is also notable that the above discussion does not use the information about the columns of a contingency table. Thus, this discussion can be extended into a $3 \times n$ contingency matrix.

7.2 Contingency Table (4×4, Rank: 3)

From four-way tables, the situation becomes more complicated. In the similar way to Sect. 7.1, a four-way contingency table is defined as:

$$M(4,4) = \begin{pmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \end{pmatrix}$$

When its rank is equal to 3, it can be assumed that the fourth row is represented by the first to third row:

$$\begin{aligned}(x_{41}\ x_{42}\ x_{43}\ x_{44}) = p(x_{11}\ x_{12}\ x_{13}\ x_{14}) \\ + q(x_{21}\ x_{22}\ x_{23}\ x_{24}) \\ + r(x_{31}\ x_{32}\ x_{33}\ x_{34})\end{aligned} \tag{19}$$

Then, the following three types of equations are obtained by simple calculation.

$$\begin{aligned}\Delta(1,j) &= (1+q)\left\{x_{1j}\sum_{k=1}^{4}x_{2k} - x_{2j}\sum_{k=1}^{4}x_{1k}\right\} \\ &\quad +(1+r)\left\{x_{1j}\sum_{k=1}^{4}x_{3k} - x_{3j}\sum_{k=1}^{4}x_{1k}\right\} \\ \Delta(2,j) &= (1+p)\left\{x_{2j}\sum_{k=1}^{4}x_{1k} - x_{1j}\sum_{k=1}^{4}x_{2k}\right\} \\ &\quad +(1+r)\left\{x_{2j}\sum_{k=1}^{4}x_{3k} - x_{3j}\sum_{k=1}^{4}x_{2k}\right\} \\ \Delta(3,j) &= (1+p)\left\{x_{2j}\sum_{k=1}^{4}x_{1k} - x_{1j}\sum_{k=1}^{4}x_{2k}\right\} \\ &\quad +(1+q)\left\{x_{1j}\sum_{k=1}^{4}x_{2k} - x_{2j}\sum_{k=1}^{4}x_{1k}\right\} \\ \Delta(4,j) &= (p-q)\left\{x_{1j}\sum_{k=1}^{4}x_{2k} - x_{2j}\sum_{k=1}^{4}x_{1k}\right\} \\ &\quad +(r-p)\left\{x_{3j}\sum_{k=1}^{4}x_{1k} - x_{1j}\sum_{k=1}^{4}x_{3k}\right\} \\ &\quad +(q-r)\left\{x_{2j}\sum_{k=1}^{4}x_{3k} - x_{3j}\sum_{k=1}^{4}x_{2k}\right\}\end{aligned}$$

Thus, the following theorem is obtained.

Theorem 10. *The fourth row represented by a linear combination of first to third rows (basis) will satisfy the condition of statistical independence if and only if* $\Delta(4, j) = 0$.

Unfortunately, the condition is not simpler than Theorem 9. It is notable $\Delta(4, j) = 0$ is a diophatine equation whose trivial solution is $p = q = r$. That is, the solution space includes not only $p = q = r$, but other solutions. Thus,

Corollary 1. *If* $p = q = r$*, then the fourth row satisfies the condition of statistical independence.*

The converse is not true.

Example 2. Let us consider the following matrix:

$$E = \begin{pmatrix} 1 & 1 & 2 & 2 \\ 2 & 2 & 3 & 3 \\ 4 & 4 & 5 & 5 \\ x_{41} & x_{42} & x_{43} & x_{44} \end{pmatrix}.$$

The question is when the fourth row represented by the other rows satisfies the condition of statistical independence. Since $x_{1j} \sum_{k=1}^{4} x_{2k} - x_{2j} \sum_{k=1}^{4} x_{1k} = -2$, $x_{1j} \sum_{k=1}^{4} x_{3k} - x_{3j} \sum_{k=1}^{4} x_{1k} = 6$ and $x_{2j} \sum_{k=1}^{4} x_{1k} - x_{1j} \sum_{k=1}^{4} x_{2k} = -4$, $\Delta(4, j)$ is equal to: $-2(p - q) + 6(r - p) - 4(q - r) = -8p - 2q + 10r$.

Thus, the set of solutions is $\{(p, q, r) | 10r = 8p + 2q\}$, where $p = q = r$ is included.

It is notable that the characteristics of solutions will be characterized by a diophantine equation $10r = 8p + 2q$ and a contingency table given by a tripule (p, q, r) may be represented by another tripule. For example, $(3, 3, 3)$ gives the same contingency table as $(1, 6, 2)$:

$$\begin{pmatrix} 1 & 1 & 2 & 2 \\ 2 & 2 & 3 & 3 \\ 4 & 4 & 5 & 5 \\ 21 & 21 & 30 & 30 \end{pmatrix}.$$

It will be our future work to investigate the general characteristics of the solution space.

7.3 Contingency Table (4×4, Rank: 2)

When its rank is equal to 2, it can be assumed that the third and fourth rows are represented by the first to third row:

$$\begin{aligned} (x_{41}\ x_{42}\ x_{43}\ x_{44}) &= p(x_{11}\ x_{12}\ x_{13}\ x_{14}) \\ &\quad + q(x_{21}\ x_{22}\ x_{23}\ x_{24}) \qquad (20) \\ (x_{31}\ x_{32}\ x_{33}\ x_{34}) &= r(x_{11}\ x_{12}\ x_{13}\ x_{14}) \\ &\quad + s(x_{21}\ x_{22}\ x_{23}\ x_{24}) \qquad (21) \end{aligned}$$

$$\Delta(1,j) = (1+q+s)\left\{x_{1j}\sum_{k=1}^{4}x_{2k} - x_{2j}\sum_{k=1}^{4}x_{1k}\right\}$$

$$\Delta(2,j) = (1+p+r)\left\{x_{2j}\sum_{k=1}^{4}x_{1k} - x_{1j}\sum_{k=1}^{4}x_{2k}\right\}$$

$$\Delta(3,j) = (p-q+ps-qr) \times \left\{x_{2j}\sum_{k=1}^{4}x_{1k} - x_{1j}\sum_{k=1}^{4}x_{2k}\right\}$$

$$\Delta(4,j) = (r-s+qr-ps) \times \left\{x_{1j}\sum_{k=1}^{4}x_{2k} - x_{2j}\sum_{k=1}^{4}x_{2k}\right\}$$

Since $p-q+ps-qr=0$ and $r-s+qr-ps=0$ gives the only reasonable solution $p=q$ and $r=s$, the following theorem is obtained.

Theorem 11. *The third and fourth rows represented by a linear combination of first and second rows (basis) will satisfy the condition of statistical independence if and only if $p=q$ and $r=w$.*

8 Pseudo-Statistical Independence

Now, we will generalize the results shown in Sect. 7. Let us consider the $m \times n$ contingency table whose r rows (columns) are described by $n-s$ rows (columns). Thus, we assume a corresponding matrix with the following equations.

$$\begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{pmatrix}$$

$$(x_{m-s+p,1}\ x_{m-s+p,2}\ \cdots\ x_{m-s+p,n}) = \sum_{i=1}^{m-s} k_{pi}(x_{i1}\ x_{i2}\ \cdots\ x_{in}) \times (1 \leq s \leq n-1, 1 \leq p \leq s) \quad (22)$$

Then, the following theorem about $\Delta(u,v)$ is obtained.

Theorem 12. *For a contingency table with size $m \times n$:*

$$\Delta(u,v) = \begin{cases} \sum_{i=1}^{m-s}(1+\sum_{r=1}^{m-s}k_{ri}) \\ \quad \times \left\{ x_{uv}(\sum_{j=1}^{n}x_{ij}) - x_{iv}(\sum_{j=1}^{n}x_{uj}) \right\} \\ \quad (1 \le u \le m-s,\ 1 \le v \le m) \\ \sum_{i=1}^{m-s}\sum_{j=1}^{n}\sum_{r=1}^{m-s} x_{r1}x_{ij} \\ \quad \times \{(k_{ur}-k_{ui}) \\ \quad +k_{ur}\sum_{r=1}^{m-s}k_{ri} - k_{ui}\sum_{r=1}^{m-s}k_{rq} \right\} \\ \quad (n-s+1 \le u \le m,\ 1 \le v \le m) \end{cases} \tag{23}$$

Thus, from the above theorem, if and only if $\Delta(u,v) = 0$ for all v, then the u-th row will satisfy the condition of statistically independence. Especially, the following theorem is obtained.

Theorem 13. *If the following equation holds for all $v(1 \le v \le n)$, then the condition of statistical independence will hold for the u-th row in a contingency table.*

$$\sum_{i=1}^{m-s}\sum_{r=1}^{m-s}\{(k_{ur}-k_{ui}) \quad +k_{ur}\sum_{r=1}^{m-s}k_{ri} - k_{ui}\sum_{r=1}^{m-s}k_{rq}\Big\} = 0 \tag{24}$$

It is notable that the above equations give diophatine equations which can check whether each row (column) will satisfy the condition of statistical independence. As a corollary,

Corollary 2. *If k_{ui} is equal for all $i = 1, \cdots, n-s$, then the u-th satisfies the condition of statistical independence.*

The converse is not true.

Example 3. Let us consider the following matrix:

$$F = \begin{pmatrix} 1 & 1 & 2 \\ 2 & 2 & 3 \\ 4 & 4 & 5 \\ x_{41} & x_{42} & x_{43} \\ x_{51} & x_{52} & x_{53} \end{pmatrix},$$

where the last two rows are represented by the first three columns. That is, the rank of a matrix is equal to 3. Then, according to Theorem 13, the following equations are obtained:

$$(5k_{53} - k_{52} - 4k_{51}) \times \{k_{41} - 2k_{43} + (k_{51} - 2k_{53} - 1)\} = 0 \quad (25)$$

$$(5k_{43} - k_{42} - 4k_{41}) \times \{k_{41} - 2k_{43} + (k_{51} - 2k_{53} - 1)\} = 0 \quad (26)$$

In case of $k_{41} - 2k_{43} + (k_{51} - 2k_{53} - 1) = 0$, simple calculations give several equations for those coefficients.

$$k_{41} + k_{51} = 2(k_{43} + k_{53}) + 1$$
$$k_{42} + k_{52} = -3(k_{43} + k_{53})$$

The solutions of these two equations give examples of pseudo-statistical independence.

9 Conclusion

In this chapter, a contingency table is interpreted from the viewpoint of granular computing and statistical independence. From the definition of statistical independence, statistical independence in a contingency table will holds when the equations of collinearity (14) are satisfied. In other words, statistical independence can be viewed as linear dependence. Then, the correspondence between contingency table and matrix, gives the theorem where the rank of the contingency matrix of a given contingency table is equal to 1 if two attributes are statistical independent. That is, all the rows of contingency table can be described by one row with the coefficient given by a marginal distribution. If the rank is maximum, then two attributes are dependent. Otherwise, some probabilistic structure can be found within attribute-value pairs in a given attribute, which we call contextual independence. Moreover, from the characteristics of statistical independence, a contingency table may be composed of statistical independent and dependent parts, which we call pseudo-statistical dependence. In such cases, if we merge several rows or columns, then we will obtain a new contingency table with statistical independence, whose rank of its corresponding matrix is equal to 1.0. Especially, we obtain Diophatine equations for a pseudo-statistical dependence. Thus, matrix algebra and elementary number theory are the key methods of the analysis of a contingency table and the degree of independence, where its rank and the structure of linear dependence as Diophatine equations play very important roles in determining the nature of a given table.

References

1. H. Coxeter, editor. *Projective Geometry*. Springer, Berlin Heidelberg New York, 2nd edition, 1987
2. A. Skowron and J. Grzymala-Busse. From rough set theory to evidence theory. In R. Yager, M. Fedrizzi, and J. Kacprzyk, editors, *Advances in the Dempster–Shafer Theory of Evidence*, pages 193–236. Wiley, New York, 1994
3. Y. Yao and S. Wong. A decision theoretic framework for approximating concepts. *International Journal of Man-machine Studies*, 37:793–809, 1992
4. Y. Yao and N. Zhong. An analysis of quantitative measures associated with rules. In N. Zhong and L. Zhou, editors, *Methodologies for Knowledge Discovery and Data Mining, Proceedings of the Third Pacific-Asia Conference on Knowledge Discovery and Data Mining LNAI* **1574**, pages 479–488. Springer, Berlin Heidelberg New York, 1999

Role of Sample Size and Determinants in Granularity of Contingency Matrix

Shusaku Tsumoto

Department of Medical Informatics,
Shimane University, School of Medicine
89-1 Enya-cho, Izumo 693-8501, Japan
tsumoto@med.shimane-u.ac.jp

Summary. This paper gives a empirical analysis of determinant, which empirically validates the trade-off between sample size and size of matrix. In the former studies, relations between degree of granularity and dependence of contingency tables are given from the viewpoint of determinantal divisors and sample size. The nature of determinantal divisors shows that the increase of the degree of granularity may lead to that of dependence. However, a constraint on the sample size of a contingency table is very strong, which leads to the evaluation formula where the increase of degree of granularity gives the decrease of dependency. This paper gives a further study of the nature of sample size effect on the degree of dependency in a contingency matrix. The results show that sample size will restrict the nature of matrix in a combinatorial way, which suggests that the dependency is closely related with integer programming.

1 Introduction

Although independence is a very important concept, it has not been fully and formally investigated as a relation between two attributes. Tsumoto introduces linear algebra into formal analysis of a contigency table [1]. The results give the following interesting results. First, a contingency table can be viewed as comparison between two attributes with respect to information granularity. Second, algebra is a key point of analysis of this table. A contingency table can be viewed as a matrix and several operations and ideas of matrix theory are introduced into the analysis of the contingency table. Especially, The degree of independence, rank plays a very important role in extracting a probabilistic model from a given contingency table.

Then, thirdly, the results of determinantal divisors show that it seems that the devisors provide information on the degree of dependencies between the matrix of the whole elements and its submatrices and the increase of the degree of granularity may lead to that of dependence [2]. This gives a

S. Tsumoto: *Role of Sample Size and Determinants in Granularity of Contingency Matrix*, Studies in Computational Intelligence (SCI) **118**, 405–421 (2008)
www.springerlink.com

contradictory view from the intuition that when two attributes has many values, the dependence between these two attributes becomes low.

The key for understanding these conflicts is to consider the constraint on the sample size.

In [3] we show that a constraint on the sample size of a contingency table is very strong, which leads to the evaluation formula where the increase of degree of granularity gives the decrease of dependency.

This paper confirms this constraint by using enumerative combinatorics.

The results show that sample size will restrict the nature of matrix in a combinatorial way, which suggests that the dependency is closely related with integer programming.

The paper is organized as follows: Section 2 shows preliminaries. Section 3 and 4 discusses the former results. Section 5 shows the effect of sample size on a matrix (2×2) theoretically. Section 6 introduces empirical validation of the results obtained in Sect. 5. Finally, Sect. 7 concludes this paper.

2 Preliminary Work

2.1 Notations

From Rough Sets

In the subsequent sections, the following notations is adopted, which is introduced in [4]. Let U denote a nonempty, finite set called the universe and A denote a nonempty, finite set of attributes, i.e., $a : U \rightarrow V_a$ for $a \in A$, where V_a is called the domain of a, respectively. Then, a decision table is defined as an information system, $A = (U, A \cup \{\mathcal{D}\})$, where $\{\mathcal{D}\}$ is a set of given decision attributes. The atomic formulas over $B \subseteq A \cup \{\mathcal{D}\}$ and V are expressions of the form $[a = v]$, called descriptors over B, where $a \in B$ and $v \in V_a$. The set $F(B, V)$ of formulas over B is the least set containing all atomic formulas over B and closed with respect to disjunction, conjunction and negation. For each $f \in F(B, V)$, f_A denote the meaning of f in A, i.e., the set of all objects in U with property f, defined inductively as follows.

1. If f is of the form $[a = v]$ then, $f_A = \{s \in U | a(s) = v\}$
2. $(f \wedge g)_A = f_A \cap g_A$; $(f \vee g)_A = f_A \vee g_A$; $(\neg f)_A = U - f_a$

Contingency Matrix

Definition 1. *Let R_1 and R_2 denote multinominal attributes in an attribute space A which have m and n values. A contingency tables is a table of a set of the meaning of the following formulas: $|[R_1 = A_j]_A|$, $|[R_2 = B_i]_A|$, $|[R_1 = A_j \wedge R_2 = B_i]_A|$, $|U|$ $(i = 1, 2, 3, \cdots, n$ and $j = 1, 2, 3, \cdots, m)$.*

Table 1. Contingency table ($n \times m$)

	A_1	A_2	$\cdots$	A_n	Sum
B_1	x_{11}	x_{12}	$\cdots$	x_{1n}	$x_{1\cdot}$
B_2	x_{21}	x_{22}	$\cdots$	x_{2n}	$x_{2\cdot}$
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
B_m	x_{m1}	x_{m2}	$\cdots$	x_{mn}	$x_{m\cdot}$
Sum	$x_{\cdot 1}$	$x_{\cdot 2}$	$\cdots$	$x_{\cdot n}$	$x_{\cdot\cdot} = \lvert U\rvert = N$

This table is arranged into the form shown in Table 1, where: $|[R_1 = A_j]_A| = \sum_{i=1}^{m} x_{1i} = x_{\cdot j}$, $|[R_2 = B_i]_A| = \sum_{j=1}^{n} x_{ji} = x_{i\cdot}$, $|[R_1 = A_j \wedge R_2 = B_i]_A| = x_{ij}$, $|U| = N = x_{\cdot\cdot}$ $(i = 1, 2, 3, \cdots, n$ *and* $j = 1, 2, 3, \cdots, m)$.

Definition 2. *A contigency matrix* $M_{R_1,R_2}(m, n, N)$ *is defined as a matrix, which is composed of* $x_{ij} = |[R_1 = A_j \wedge R_2 = B_i]_A|$, *extracted from a contigency table defined in definition 1.*

That is,

$$M_{R_1,R_2}(m, n, N) = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1n} & x_{1\cdot} \\ x_{21} & x_{22} & \cdots & x_{2n} & x_{2\cdot} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} & x_{m\cdot} \end{pmatrix}.$$

For simplicity, if we do not need to specify R_1 and R_2, we use $M(m, n, N)$ as a contingency matrix with m rows, n columns and N samples.

One of the important observations from granular computing is that a contingency table shows the relations between two attributes with respect to intersection of their supporting sets. When two attributes have different number of equivalence classes, the situation may be a little complicated. But, in this case, due to knowledge about linear algebra, we only have to consider the attribute which has a smaller number of equivalence classes. and the surplus number of equivalence classes of the attributes with larger number of equivalnce classes can be projected into other partitions. In other words, a $m \times n$ matrix or contingency table includes a projection from one attributes to the other one.

2.2 Rank of Contingency Matrix ($m \times n$)

In the former paper, Tsumoto obtained the following theorem [1].

Theorem 1. *Let the contingency matrix of a given contingency table be a* $m \times n$ *matrix. The rank of this matrix is less than* $\min(m, n)$. *If the rank of the corresponding matrix is 1, then two attributes in a given contingency table are statistically independent. If the rank of the corresponding matrix is* n, *then two attributes in a given contingency table are dependent. Otherwise,*

two attributes are contextual dependent, which means that several conditional probabilities can be represented by a linear combination of conditional probabilities. Thus,

$$rank = \begin{cases} \min(m,n) & dependent \\ 2, \cdots, & \\ \quad \min(m,n)-1 & contextual\ independent \\ 1 & statistical\ independent \end{cases}$$

In the cases of $m \neq n$, we need a discussion on submatrix and subderminant in the next section.

2.3 Submatrix and Subdeterminant

The next interest is the structure of a corresponding matrix with $1 \leq rank \leq n-1$. First, let us define a submatrix (a subtable) and subdeterminant.

Definition 3. *Let A denote a corresponding matrix of a given contigency table $(m \times n)$. A corresponding submatrix $A^{i_1 i_2 \cdots i_r}_{j_1 j_2 \cdots j_s}$ is defined as a matrix which is given by an intersection of r rows and s columns of A $(i_1 < i_2 < \cdots < i_r, j_1 < j_2 < \cdots < j_r)$.*

Definition 4. *A subdeterminant of A is defined as a determinant of a submatrix $A^{i_1 i_2 \cdots i_r}_{j_1 j_2 \cdots j_s}$, which is denoted by $det(A^{i_1 i_2 \cdots i_r}_{j_1 j_2 \cdots j_s})$.*

Let us consider the contingency table given as Table 1. Then, a subtable for $A^{i_1 i_2 \cdots i_r}_{j_1 j_2 \cdots j_s}$ is given as Table 2.

Rank and Subdeterminant

Let δ_{ij} denote a co-factor of a_{ij} in a square corresponding matrix of A. Then,

$$\Delta_{ij} = (-1)^{i+j} det\Big(A^{1,2,\cdots,i-1,i+1,\cdots,n}_{1,2,\cdots,j-1,j+1,\cdots,n}\Big).$$

It is notable that a co-factor is a special type of submatrix, where only ith-row and j-column are removed from a original matrix. By the use of co-factors, the determinant of A is defined as:

Table 2. A subtable $(r \times s)$

	A_{j_1}	A_{j_2}	$\cdots$	A_{j_r}	Sum
B_{i_1}	$x_{i_1 j_1}$	$x_{i_1 j_2}$	$\cdots$	$x_{i_1 j_r}$	$x_{i_1 \cdot}$
B_{i_2}	$x_{i_2 j_1}$	$x_{i_2 j_2}$	$\cdots$	$x_{i_2 j_r}$	$x_{i_2 \cdot}$
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
B_{i_r}	$x_{i_r j_1}$	$x_{i_r j_2}$	$\cdots$	$x_{i_r j_n}$	$x_{i_r \cdot}$
Sum	$x_{\cdot 1}$	$x_{\cdot 2}$	$\cdots$	$x_{\cdot n}$	$x_{\cdot\cdot} = \|U\| = N$

$$det(A) = \sum_{j=1}^{n} a_{ij} \Delta_{ij},$$

which is called *Laplace expansion.*

From this representation, if $det(A)$ is not equal to 0, then $\Delta_{ij} \neq 0$ for $\{a_{i1}, a_{i2}, \cdots, a_{in}\}$ which are not equal to 0. Thus, the following proposition is obtained.

Proposition 1. *If $det(A)$ is not equal to 0 if at least one co-factor of $a_{ij}(\neq 0)$, Δ_{ij} is not equal to 0.*

It is notable that the above definition of a determinant gives the relation between a original matrix A and submatrices (co-factors). Since cofactors gives a square matrix of size $n-1$, the above proposition gives the relation between a matrix of size n and submatrices of size $n-1$. In the same way, we can discuss the relation between a corresponding matrix of size n and submatrices of size $r(1 \leq r < n-1)$.

Rank and Submatrix

Let us assume that corresponding matrix and submatrix are square ($n \times n$ and $r \times r$, respectively).

Theorem 2. *If the rank of a corresponding matrix of size $n \times n$ is equal to r, at least the determinant of one submatrix of size $r \times r$ is not equal to 0. That is, there exists a submatrix $A^{i_1 i_2 \cdots i_r}_{j_1 j_2 \cdots j_r}$, which satisfies $det(A^{i_1 i_2 \cdots i_r}_{j_1 j_2 \cdots j_r}) \neq 0$*

Corollary 1. *If the rank of a corresponding matrix of size $n \times n$ is equal to r, all the determinants of the submatrices whose number of columns and rows are larger than $r+1(\leq n)$ are equal to 0.*

3 Degree of Dependence

3.1 Determinantal Divisors

From the subdeterminants of all the submatrices of size 2, all the subdeterminants of a corresponding matrix has the greatest common divisor, equal to 3.

From the recursive definition of the determinants, it is show that the subdeterminants of size $r+1$ will have the greatest common divisor of the subdeterminants of size r as a divisor. Thus,

Theorem 3. *Let $d_k(A)$ denote the greatest common divisor of all the subdeterminants of size k, $det(A^{i_1 i_2 \cdots i_k}_{j_1 j_2 \cdots j_r})$. $d_1(A), d_2(A), \cdots, d_n(A)$ are called determinantal divisors. From the definition of Laplace expansion,*

$$d_k(A) | d_{k+1}(A).$$

In the example of the above subsection, $d_1(A) = 1$, $d_2(A) = 3$ and $d_3(A) = 0$.

Example 1. Let us consider the following corresponding matrix:

$$B = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & \mathbf{11} & 9 \end{pmatrix}.$$

Calculation gives: $d_1(B) = 1$, $d_2(B) = 3$ and $d_3(B) = 18$.

It is notable that a simple change of a corresponding matrix gives a significant change to the determinant, which suggests a change of structure in dependence/independence.

The relation between $d_k(A)$ gives a interesting constraint.

Proposition 2. *Since $d_k(A)|d_{k+1}(A)$, the sequence of the devisors is monotonically increasing one:*

$$d_1(A) \leq d_2(A) \cdots \leq d_r(A),$$

where r denotes the rank of A.

The sequence of B illustrates this: $1 < 3 < 18$.

Let us define a ratio of $d_k(A)$ to $d_{k-1}(A)$, called *elementary divisors*, where C denotes a corresponding matrix and $k \leq rankA$:

$$e_k(C) = \frac{d_k(C)}{d_{k-1}(C)} (d_0(C) = 0).$$

The elementary divisors may give the increase of dependency between two attributes. For example, $e_1(B) = 1$, $e_2(B) = 3$, and $e_3(B) = 6$. Thus, a transition from 2×2 to 3×3 have a higher impact on the dependency of two attributes.

It is trivial to see that $det(B) = e_1 e_2 e_3$, which can be viewed as a decomposition of the determinant of a corresponding matrix.

3.2 Divisors and Degree of Dependence

Since the determinant can be viewed as the degree of dependence, this result is very important. If values of all the subdeterminants (size r) are very small (nearly equal to 0) and $d_r(A) \simeq 1$, then the values of the subdeterminants (size $r + 1$) are very small. This property may hold until the r reaches the rank of the corresponding matrix. Thus, the sequence of the divisors of a corresponding matrix gives a hidden structure of a contingency table.

Also, this results show that $d_1(A)$ and $d_2(A)$ are very important to estimate the rank of a corresponding matrix. Since $d_1(A)$ is only given by the greatest common divisor of all the elements of A, $d_2(A)$ are much more important

components. This also intuitively suggests that the subdeterminants of A with size 2 are principal components of a corresponding matrix from the viewpoint of statistical dependence.

Recall that statistical independence of two attributes is equivalent to a corresponding matrix with rank being 1. A matrix with rank being 2 gives a context-dependent independence, which means three values of two attributes are independent, but two values of two attributes are dependent.

3.3 Elementary Divisors and Elementary Transformation

Let us define the following three elementary (row/column)transformations of a corresponding matrix:

1. Exchange two rows (columns), i_0 and j_0 ($P(i_0, j_0)$).
2. Multiply -1 to a row (column) i_0 ($T(i_0; -1)$).
3. Multiply t to a row (column) j_0 (i_0) and add it to a row i_0 (j_0). ($W(i_0, j_0, t)$).

Then, three transformations have several interesting characteristics.

Proposition 3. *Matrices corresponding to three elementary transformations are regular.*

Proposition 4. *Three elementary transformations do not change the rank of a corresponding matrix.*

Proposition 5. *Let $\tilde{A}$ denote a matrix transformed by finite steps of three operations. Then,*

$$rank\tilde{A} = rankA, \quad d_r(\tilde{A}) = d_r(A),$$

where r denotes the rank of matrix A.

Then, from the results of linear algebra, the following interesting result is obtained.

Theorem 4. *With the finite steps of elementary transformations, a given corresponding matrix is transformed into*

$$\tilde{A} = \left(\begin{array}{cccc|c} e_1 & & & & \\ & e_2 & & & \\ & & \ddots & & O \\ & & & e_r & \\ \hline & & O & & O \end{array}\right),$$

where $e_j = \frac{d_j(A)}{d_{j-1}(A)} (d_0(A) = 1)$ and r denotes the rank of a corresponding matrix. Then, the determinant is decomposed into the product of e_j.

$$d_r(\tilde{A}) = d_r(A) = e_1 e_2 \cdots e_r.$$

4 Degree of Granularity and Dependence

From Theorem 4, it seems that the increase of the degree of granularity gives that of the dependence between two attributes.

However, our empirical observations are different from the above intuitive analysis. Thus, there should be a strong constraint which suppress the above effects on the degree of granularity.

Let us assume that the determinant of a give contingency matrix gives the degree of the dependence of the matrix. Then, from the results of linear algebra, we obtain the following theorem.

Theorem 5. *Let A denote a $n \times n$ contingency matrix, which includes N samples. If the rank of A is equal to n, then there exists a matrix B $(n \times n)$ which satisfies*

$$BA = \begin{pmatrix} \rho_1 & & & \\ & \rho_2 & & O \\ & & \ddots & \\ & O & & \rho_n \end{pmatrix} = P,$$

where $\rho_1 + \rho_2 + \cdots + \rho_n = N$.

It is notable that the value of determinants of P is larger than A:

$$detA \leq detP$$

Example 2. Let us consider B as an example (Example 1). Let C denote the orthogonal matrix for transformation of B. Since the cardinality of B is equal to 48, the diagonal matrix which gives the maximum determinant is equal to:

$$\begin{pmatrix} 16 & 0 & 0 \\ 0 & 16 & 0 \\ 0 & 0 & 16 \end{pmatrix}.$$

On the other hand, the determinant of B is equal to 18. Thus, $detB = 18 < 16^3 = 4096$. Then, C is obtained from the following equation.

$$C \times \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 11 & 9 \end{pmatrix} = \begin{pmatrix} 16 & 0 & 0 \\ 0 & 16 & 0 \\ 0 & 0 & 16 \end{pmatrix}.$$

Thus,

$$C = \begin{pmatrix} -56/3 & 40/3 & -8/3 \\ 16/3 & -32/3 & 16/3 \\ 8 & 8/3 & -8/3 \end{pmatrix}$$

It is notable that the determinant of C is equal to 2048/9. Also, since $detB = 18$, we do not have any diagonal matrix whose determinant is equal to 18 and the sum of all the elements is equal to 48.

It is easy to see that the transformed matrix P has a very nice property to calculate the determinant.

Proposition 6. *The determinant of the transformed matrix P is equal to the multiplication of ρ_1 to ρ_n. That is,*

$$detP = \rho_1 \rho_2 \cdots \rho_n$$

Then, the following constraint will be have the special meaning:

$$\rho_1 + \rho_2 + \cdots + \rho_n = N, \tag{1}$$

because the following inequality holds in general:

$$\frac{\rho_1 + \rho_2 + \cdots + \rho_n}{n} \geq \sqrt[n]{\rho_1 \rho_2 \cdots \rho_n}, \tag{2}$$

where the equality holds when $\rho_1 = \rho_2 = \cdots = \rho_n$. Since the above inequality can be transformed into:

$$\rho_1 \rho_2 \cdots \rho_n \leq \left(\frac{\rho_1 + \rho_2 + \cdots + \rho_n}{n} \right)^n,$$

the following inequality is obtained:

$$detP = \rho_1 \rho_2 \cdots \rho_n \leq \left(\frac{\rho_1 + \rho_2 + \cdots + \rho_n}{n} \right)^n, \tag{3}$$

where the equality holds when $\rho_1 = \rho_2 = \cdots = \rho_n$. From the theorem 5 and equation 1, the following theorem is obtained.

Theorem 6. *When a contingency matrix A holds $AB = P$, where P is a diagonal matrix, the following inequality holds:*

$$detA \leq \left(\frac{N}{n} \right)^n,$$

Proof.

$$\begin{aligned} detA &= det(PB^{-1}) \\ &\leq detP \\ &= \rho_1 \rho_2 \cdots \rho_n \\ &\leq \left(\frac{\rho_1 + \rho_2 + \cdots + \rho_n}{n} \right)^n = \left(\frac{N}{n} \right)^n, \end{aligned} \tag{4}$$

where the former equality holds when $detB^{-1} = detB = 1$ and the latter equality holds when $\rho_1 = \rho_2 = \cdots = \rho_n = \frac{N}{n}$. □

Example 3. Let us consider the following contingency matrices D and E:

$$D = \begin{pmatrix} 1 & 2 & 3 & 0 \\ 4 & 5 & 6 & 0 \\ 7 & \mathbf{11} & 9 & 0 \\ 0 & 0 & 0 & \mathbf{1} \end{pmatrix}$$

$$E = \begin{pmatrix} 1 & 2 & 3 & 0 \\ 4 & 5 & 6 & 0 \\ 7 & \mathbf{10} & 9 & 0 \\ 0 & 0 & 0 & \mathbf{1} \end{pmatrix}$$

The numbers of examples of D and E are 49 and 48, respectively, which can be comparable to that of B. Then, from Theorem 6,

$$detD = 18 < (49/4)^4 = \frac{5764801}{256} \sim 22518$$
$$detE = 12 < (48/4)^4 = 20736$$

Thus, the maximum value of the determinant of A is at most $\left(\frac{N}{n}\right)^n$. Since N is constant for the given matrix A, the degree of dependence will decrease very rapidly when n becomes very large. That is,

$$detA \sim n^{-n}.$$

Thus,

Corollary 2. *The determinant of A will converge into 0 when n increases into infinity.*

$$\lim_{n \to \infty} detA = 0.$$

This results suggest that when the degree of granularity becomes higher, the degree of dependence will become lower, due to the constraints on the sample size.

However, it is notable that N/n is very important. If N is very large, the rapid decrease will be observed N is close to n. Even N is 48 as shown in Example 3, $n = 3, 4$ may give a strong dependency between two attributes. For the behavior of $(N/n)^n$, we can apply the technique of real analysis, which will our future work.

5 Distribution of Determinant

As shown in the former section, the determinant of D and E is significantly smaller than the maximum value of the determinant of a set of matrix $\{M(4, 4, 49)\}$ or $\{M(4, 4, 48)\}$. Then, the next interest is how is the statistical nature of the determinant for $M(m, n, N)$.

First, since a 2×2 matrix is a basic one, let us examine the nature of $\det M(2,2,N)$.

5.1 Total Number of $M(2,2,N)$

Let the four elements of $M(2,2,N)$ be denoted as a,b,c,d. That is, $x_{11} = a$, $x_{12} = b$, $x_{21} = c$, and $x_{22} = d$. Then, $a+b+c+d = N$.

Let us assume that $a = 0$. Then, $b+c+d = N$. Recursively, we can assume that $b = 0$. Then, for this pair $(a,b) = 0$, we have $(N+1)$ pairs which satisfies $c+d = N$. In this way, the total number of $M(2,2,N)$ is obtained as:

$$\sum_{i=0}^{N} \frac{(N+1-i) \times (N+2-i)}{2}.$$

Simple calculation shows that the above formula is equal to:

$$\frac{1}{6}(N+1)(N+2)(N+3).$$

That is,

Theorem 7. *The total number of a contingency matrix M(2,2,N) is equal to:*

$$\frac{1}{6}(N+1)(N+2)(N+3).$$

(*Proof Sketch*)

The total combination of $M(2,2,N)$ *is given as:*

$$\begin{aligned} \sum_{i=0}^{N} \left(\sum_{k=1}^{(N-i)+1} k \right) &= \sum_{i=0}^{N} \frac{(N+1-i) \times (N+2-i)}{2} \\ &= \sum_{i=0}^{N} \left\{ \frac{1}{2}(N+1)(N+2) - \frac{1}{2}(2N+3)i + \frac{1}{2}i^2 \right\} \\ &= \frac{1}{6}(N+1)(N+2)(N+3) \end{aligned} \tag{5}$$

Intuitively, this formula can be interpreted as follows. We have four parameters, a,b,c,d, which will take a value between 0 and N. Thus, the original freedom is 4, and the order of total number can be N^4. However, since a constraint $a+b+c+d = N$ is given, we have only three free parameters, thus the order of total number of $M(2,2,N)$ is approximately of N^3:

$$\#\ of\ M(2,2,N) \approx \mathcal{O}(N^3).$$

5.2 Total Number of $Det = 0$

Enumeration of total number of $det = 0$ is very difficult. However, upper bound can be calculated as follows. When a and d is fixed, we have obtained two constraints:

$$b + c = N - (a + d)$$
$$bc = ad$$

Thus, (b, c) can be obtained as a solution for quadratic equations. If the pair (b, c) is integer, we will have obtained two solutions $(ad - bc = 0)$ for each pair: (b,c) and (c,b).

Therefore, the upper bound of the number of solutions is equal to:

$$\sum_{i=0}^{N} \left(\sum_{k=1}^{(N-i)+1} 2 \right) = (N + 1)(N + 2)$$

Theorem 8. *The upper bound of total number of a contingency matrix M(2,2,N) with determinant being 0 is equal to:*

$$(N + 1)(N + 2)$$

Thus, the probability that the determinant of a matrix $M(2, 2, N)$ is equal to 0 is at most:

$$\frac{(N + 1)(N + 2)}{\frac{1}{6}(N + 1)(N + 2)(N + 3)} = \frac{6}{N + 3}.$$

Then, how is the lower bound ? This is the case when (b,c) does not have any integer solution for a given quadratic equations except for trivial solutions. The simple trivial solutions are: $a = 0$ or $d = 0$ with $b = 0$ or $c = 0$. Then, for $a = 0$, $b = 0$, we may have a solution for $c + d = N$, N pairs $(c \neq 0, d \neq 0)$. Totally, 4N pairs. If we consider the cases when three values are equal to 0, such as $a = b = c = 0$, we have four pairs. Thus, totally. we have 4(N+1) pairs.

Theorem 9. *The lower bound of total number of a contingency matrix M(2,2,N) with determinant being 0 is equal to:*

$$4(N + 1)$$

Thus, the probability that the determinant of a matrix $M(2, 2, N)$ is equal to 0 is at least:

$$\frac{4(N + 1)}{\frac{1}{6}(N + 1)(N + 2)(N + 3)} = \frac{24}{(N + 2)(N + 3)}.$$

Thus, it is expected that the number of matrices with 0 determinant vibrates between $4(N+1)$ and $(N+1)(N+2)$. The variance will become larger when N grows. In other words, the probability of $det = 0$ will vibrate between $\mathcal{O}(1/N^2)$ and $\mathcal{O}(1/N)$. The variance will become larger when N grows.

It is notable that the above discussion can be applied to a general case, such as $ad - bc = k$, or other constraint. For example, if we have a constraint such as $a/(a+b)$ or $a/(a+c)$, then we can analyze a constraint for accuracy or coverage. It will be our future work to investigate such cases.

6 Empirical Validations

For empirical validations, we calculate the whole combination of a 2×2 matrix with fixed sample size $(0 \leq N \leq 100)$ $M(2, 2, N)$.

6.1 Total Number of $M(2, 2, N)$

Figure 1 plots the relation between sample size N and the total number of $M(2, 2, N)$. This figure clearly shows that the relation is polynomial.

On the other hand, Fig. 2, which plots the relation between sample size and the total number of matrices with zero determinant, gives an interesting feature. As discussed in Sect. 5, the total number vibrates and the amplitude

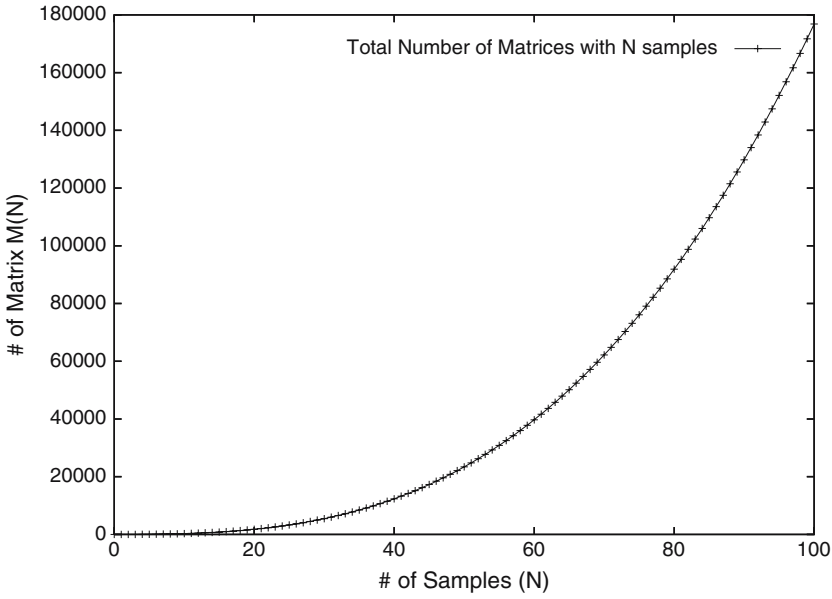


Fig. 1. Total number of $M(2, 2, N)$

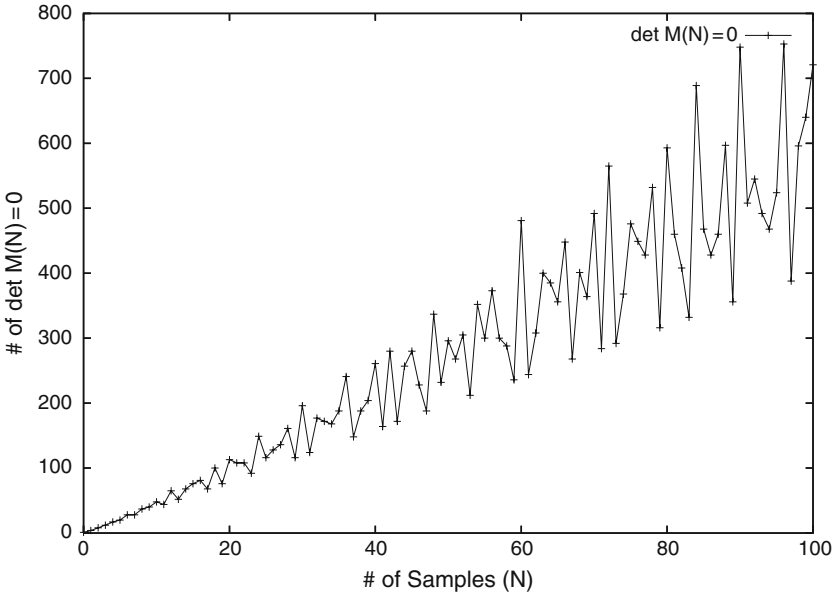


Fig. 2. Number of matrices with [Det = 0] in $M(2, 2, N)$

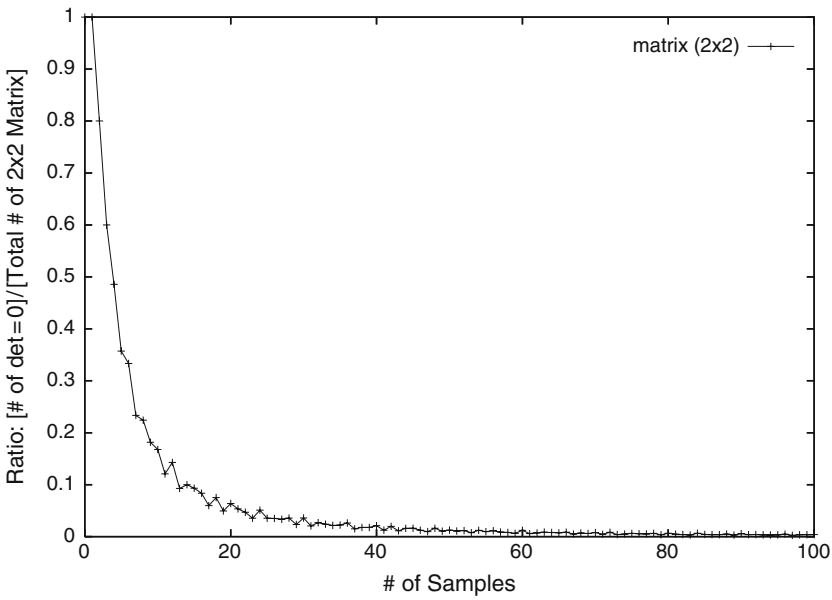


Fig. 3. Ratio of [Det = 0] in $M(2, 2, N)$

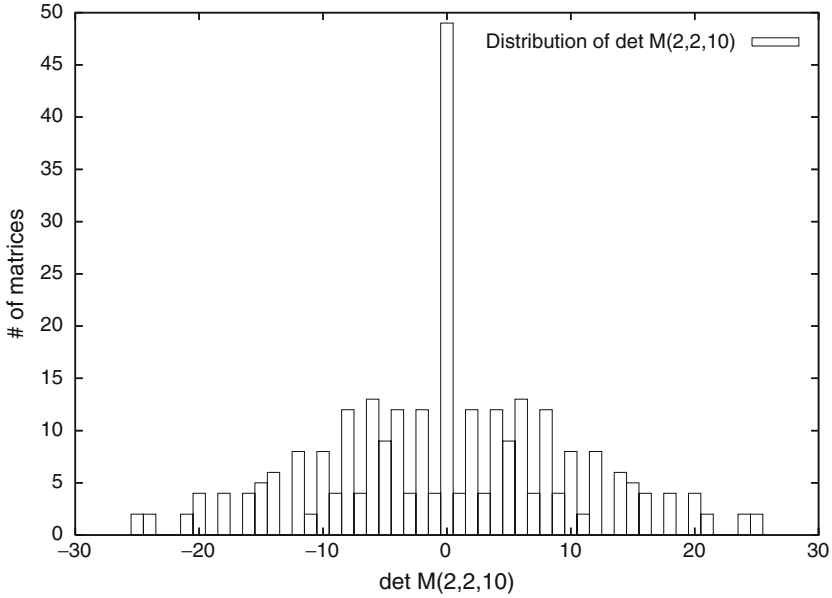


Fig. 4. Distribution of $\det M(2, 2, 10)$

of the vibration becomes larger when N grows. Furthermore, the lower bound of the total number can be approximately equal to a linear function, whereas the upper bound is to a quadratic function.

Finally, the ratio of the number of matrices with zero determinant to the total number of $M(2, 2, N)$ is plotted as Fig. 3. This figure also confirms the results obtained in Sect. 5.

6.2 Statistics of Determinant

Figures 4 and 5 show the distributions of the determinant of $M(2, 2, 10)$ and $M(2, 2, 50)$. The distribution are symmetric, and the median and average are exactly equal to 0. Furthermore, the number of matrices with 0 determinant is very high, compared with other values.

Figure 6 plots the distribution of $|\det M(2, 2, 50)|$, which suggests that the distribution is like $1/N$. However, it is notable that the vibration is observed for a given determinant value.

It is also notable that since the ratio of $det = 0$ rapidly decreases as N grows, the number of matrices with 0 determinant becomes smaller.

Tables 3 and 4 shows the statistics of those matrices.

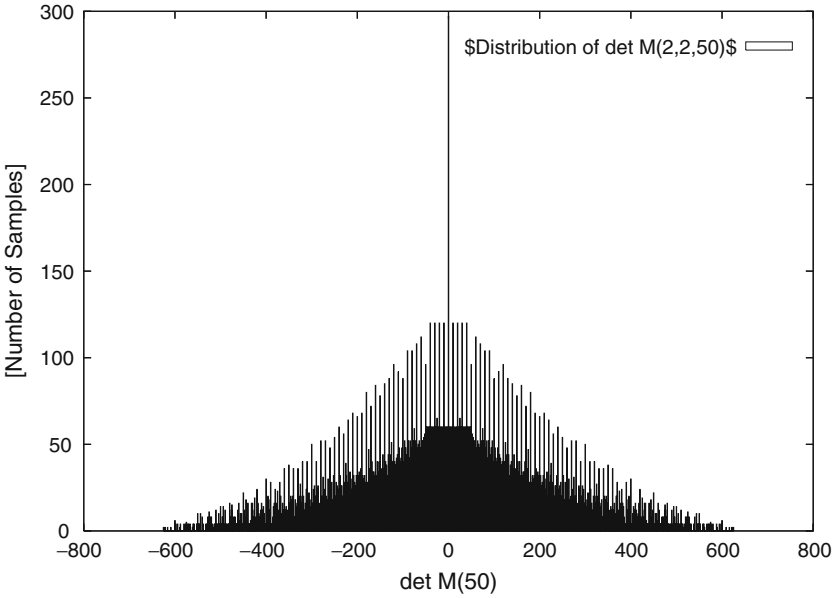


Fig. 5. Distribution of $\det M(2, 2, 50)$

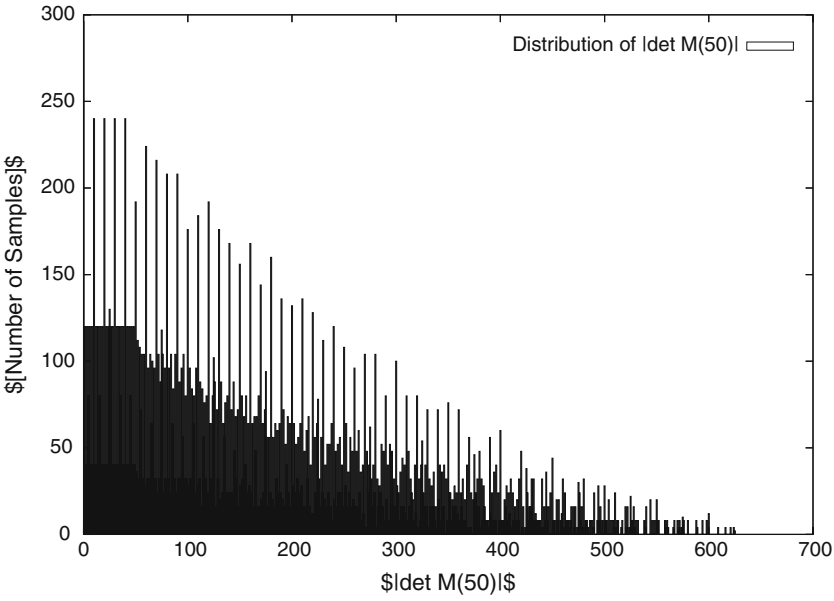


Fig. 6. Distribution of $|\det M(2, 2, 50)|$

Table 3. Statistics of $M(2, 2, 10)$

	det	|det|
min	−25	0
25%	−6	2
50%	0	6
75%	6	11.75
max	25	25

Table 4. Statistics of $M(2, 2, 50)$

	det	|det|
min	−625	0
25%	−140	60
50%	0	140
75%	140	256
max	625	625

7 Conclusion

In this paper, the nature of the dependence of a contingency matrix and the statistical nature of the determinant are examined.

Especially, the constraint on the sample size N of a contingency table will determine the number of 2×2 matrices. As N grows, the ratio of matrices with zero determinant rapidly decreases, which shows that the number of matrix with statistical dependence will increase. However, due to the nature of the determinant, the average of absolute value of the determinant also increase with the order of N^2, whereas the increase in the size of total number of matrix is of N^3.

This is a preliminary work on the statistical nature of the determinant, and it will be our future work to investigate the nature of 3×3 or higher dimensional contingency matrices.

References

1. Tsumoto, S.: Statistical independence as linear independence. In Skowron, A., Szczuka, M., eds.: Electronic Notes in Theoretical Computer Science. Volume 82., Elsevier, Amsterdam (2003)
2. Tsumoto, S., Hirano, S.: Determinantal divisors for the degree of independence of a contingency matrix. In: Proceedings of NAFIPS 2004, IEEE press (2004)
3. Tsumoto, S., Hirano, S.: Degree of dependence as granularity in contingency table. In Hu, T., Lin, T., eds.: Proceedings of IEEE GrC 2005, IEEE press (2005)
4. Skowron, A., Grzymala-Busse, J.: From rough set theory to evidence theory. In Yager, R., Fedrizzi, M., Kacprzyk, J., eds.: Advances in the Dempster-Shafer Theory of Evidence. Wiley, New York (1994) 193–236

Generating Concept Hierarchies from User Queries

Bob Wall[1], Neal Richter[2], and Rafal Angryk[3]

[1] RightNow Technologies, Bozeman, MT, USA
bwall@rightnow.com
[2] RightNow Technologies, Bozeman, MT, USA
nealr@rightnow.com
[3] Montana State University, Bozeman, MT, USA
angryk@cs.montana.edu

Summary. Most information retrieval (IR) systems are comprised of a focused set of domain-specific documents located within a single logical repository. A mechanism is developed by which user queries against a particular type of IR repository, a frequently asked question (FAQ) system, are used to generate a concept hierarchy pertinent to the domain. First, an algorithm is described which selects a set of user queries submitted to the system, extracts terms from the repository documents matching those queries, and then reduces this set of terms to a manageable length. The resulting terms are used to generate a feature vector for each query, and the queries are clustered using a hierarchical agglomerative clustering (HAC) algorithm. The HAC algorithm generates a binary tree of clusters, which is not particularly amenable to use by humans and which is slow to search due to its depth, so a subsequent processing step applies min–max partitioning to form a shallower, bushier tree that is a more natural representation of the hierarchy of concepts inherent in the system. Two alternative versions of the partitioning algorithm are compared to determine which produces a more usable concept hierarchy.

The goal is to generate a concept hierarchy that is built from phrases that users actually enter when searching the repository, which should make the hierarchy more usable for all users. While the algorithm presented here is applied to an FAQ system, the techniques can easily be extended to any IR system that allows users to submit natural language queries and that selects documents from the repository that match those queries.

1 Introduction

As the World Wide Web is assimilated more completely into our culture, people are increasingly willing and able to help themselves by finding answers to questions and solutions to problems online. Companies are able to realize considerable savings in their customer support costs by implementing an

B. Wall et al.: *Generating Concept Hierarchies from User Queries*, Studies in Computational Intelligence (SCI) **118**, 423–441 (2008)
www.springerlink.com

easy-to-use customer self-help system. The easier it is to navigate the system and to find and extract information, the greater the benefit to the company.

An information retrieval (IR) system serving as a customer self-help (frequently asked questions or FAQ) system is a key component of the customer service product offered by RightNow Technologies, Inc., a leading provider of on-demand software to help companies manage their customer interactions. The RightNow self-help system and similar products from other vendors are becoming ubiquitous on support sites throughout the Web. A diverse group of manufacturers, retailers, and service providers, including the Social Security Administration, Nikon, Dell, Qwest, and Florida State University, utilize FAQ systems as integral components of their customer support offerings.

In addition to searching the FAQ repository for relevant documents using keyword searches or natural language queries (via a standard IR system), product and category associations, and other filtering techniques, the RightNow FAQ system also includes an unsupervised machine learning algorithm that clusters the documents in the system, grouping documents containing similar sets of terms and phrases, so that users can browse the document collection in an organized fashion without necessarily having an exact question clearly specified.

RightNow Technologies' research shows that a significant enhancement is to cluster users' search phrases, with a similar goal. The query taxonomy created by this process can help demonstrate to users the types of queries that can be effectively answered by the FAQ system and can illustrate the progression of detail from general concepts to more specific information. The hierarchy can also help the system's user interface to adapt to typical use, indicating not only the contents of the system but how users are attempting to retrieve information. Members of the clusters can potentially serve as a source of additional search terms to help focus user queries and retrieve smaller sets of more relevant documents. Also, system administrators can examine the topic or concept hierarchy to evaluate the quality and usefulness of the documents contained in the system. If queries that are deemed too dissimilar are clustered together, additional documents more specific to the questions being asked should be added. (This process of analyzing the repository to find content shortcomings is known as gap analysis [1].)

Note that this clustering of user queries is not intended to replace the clustering of FAQs; the documents in the FAQ repository are still clustered according to their content. The user query hierarchy should augment this information, describing not what is contained within the repository but what users are expecting to find there and how their questions get related by the FAQ content and search process. Also, this technique is not limited to FAQ systems. Any information retrieval system in which user queries are used to return a subset of a document repository could benefit from this technique.

This paper describes the HAC + P + FSR (for HAC clustering plus partitioning plus feature selection and reduction) algorithm, which is an extension to a previously designed algorithm, HAC+P. It also presents a similar

algorithm, HAC+P2+FSR, that uses an alternative partitioning technique. Our new algorithms take advantage of the controlled environment of the FAQ repository and the direct, well-defined relationship between user queries and the FAQ documents to cluster the user queries. Since the algorithms have access to the internal details of the system, including accurate frequency counts for terms within documents and computed relevance scores for searches, they are able to utilize this information to accurately cluster queries. The output of the clustering process serves as a hierarchy of the concepts contained within the system.

The remainder of this paper is organized as follows: Section 2 reviews related work in the field, and in particular the HAC + P algorithm, Sect. 3 describes our HAC+P+FSR and HAC + P2 + FSR algorithms in detail, Sect. 4 summarizes the results of experiments conducted on several RightNow FAQ systems to evaluate the quality of the clustering, and Sect. 5 presents conclusions and discusses related future work.

2 Background

A significant amount of research has been conducted into methods of clustering documents hierarchically. In [2], Sanderson and Croft describe a technique of automatically generating the concept hierarchy by extracting terms and phrases from a set of documents and generating the hierarchy. This technique is called subsumption, where the parent of a set of clustered documents must match a list of keywords that is a superset of the keywords that were matched by the documents in the cluster.

However, this method and most others described in the literature are focused on clustering documents, not the user queries. Clustering queries is inherently more complicated than clustering documents, because of the abbreviated nature of the source material. Individual queries do not include enough context in and of themselves to provide sufficient input for a clustering algorithm. In order to cluster short text phrases, they must be associated with some additional source of meaning. Cilibrasi and Vitanyi [3] propose the idea of interpreting the search results returned by Google when a word or phrase is submitted as a search query. They claim that the huge body of text that is indexed by Google represents an amassing of human knowledge, and that this technique can be used to automatically extract the meaning of words.

The technique developed by Chuang and Chien [4, 5] is based on a similar concept – a phrase is submitted to a search engine, and keywords are extracted from the snippets of text returned in the search results and used to form the context for the query. These keywords are weighted to form feature vectors that are used to cluster the queries using a Hierarchical Agglomerative Clustering (HAC) algorithm.

A drawback of HAC is that the hierarchy of clusters formed, referred to as a dendrogram, tends to be tall and narrow. In the extreme case, the dendrogram can have $n-1$ levels for n queries. Such a hierarchy is not well suited to interpretation and understanding. Chuang and Chien augment the HAC algorithm with a subsequent min–max partitioning algorithm to flatten the hierarchy, producing a shallower multi-way tree that is much more readily understandable.

This algorithm, which the authors refer to as HAC + P [5], was shown to produce fairly good results across several different data sets. The authors evaluated the algorithm using an analytic measure, the F-measure, that is commonly used to measure how well a classification algorithm matches a priori classifications for its input data. The authors used a version of the F-measure first introduced by Larsen and Aone that is particularly appropriate for text classification [6]. They also included two subjective measures; the first was a user survey in which the users were asked to assign numeric scores for six different criteria to the results produced by the algorithm. The second was a usability test that attempted to gauge how much the use of the concept hierarchy generated by the algorithm could assist human experts in reconstructing an existing topic hierarchy.

The HAC + P algorithm was shown to produce impressive scores compared to an HK-Means clustering algorithm (a modification of standard k-means clustering that uses a top-down approach to build a hierarchy of the clusters) for the analytical tests. Subjective results were also good for the most part; comparing the results to an existing topic hierarchy generated for one of the data sets showed a reasonable approximation for five of the six criteria evaluated, and the usability test showed that the automatically generated concept hierarchy was of significant aid to people in constructing an accurate topic hierarchy. This algorithm seems to be the best candidate for use in query clustering and concept hierarchy generation within the RightNow system, and we used it as a starting point for our research. Our new algorithms, HAC + P + FSR and HAC + P2 + FSR, are described in detail, and their performance on some actual production RightNow FAQ systems is evaluated.

3 The HAC + P + FSR Algorithms

This section describes the HAC+P+FSR algorithms in detail. The basic algorithm (Fig. 1) is common to both HAC + P + FSR and HAC + P2 + FSR. It is designed to interact with the RightNow Technologies system, but it should be easy to adapt to the internals of a different FAQ system.

3.1 Feature Selection and Reduction (FSR)

The remainder of the section is separated into four sections: feature selection and reduction (FSR) and feature vector generation, HAC, min–max partitioning, and an alternate technique for partitioning. Section 3.1 describes how the

1. select queries to cluster
2. select keywords associated with queries
3. reduce set of keywords to manageable length
4. generate feature vector for each query
5. use HAC algorithm to cluster queries
6. apply min–max partitioning algorithm to flatten dendrogram

Fig. 1. HAC + P + FSR algorithm

database is used to retrieve the list of queries to cluster, the list of documents that form the context for each query, and the list of keywords from those documents that are used to form feature vectors for the queries (steps one through four of Fig. 1). Section 3.2 describes HAC, which is a well known clustering algorithm. The next section describes how min–max partitioning is used to flatten the cluster hierarchy into a shallow multi-way tree. The last section describes an alternative mechanism for partitioning the tree; it uses the same partitioning algorithm but applies different metrics to the generated clusters while partitioning.

The problems encountered extracting keywords from text snippets returned by a search engine are all avoided by the HAC + P + FSR algorithm. It has access to the internal data of the RightNow system; one key advantage of this is that the documents in the FAQ repository have already been processed using stop word lists and stemming, and the resulting keyword phrases have been extracted and stored in a document index. Each phrase includes a count of the number of times it appeared in different sections of the document (the FAQ documents are structured and contain sections such as the title, keywords, question, and answer). Currently, the HAC+P+FSR algorithm only uses single-word phrases in its feature vectors.

The user queries have also been filtered by the stop word list, stemmed, and stored along with a count of the documents matching the queries and a list of the most relevant documents' IDs. The algorithm groups the search queries by stemmed, filtered phrase and selects the most frequently occurring queries. Queries are only included for which at least one matching document was found; queries with no matches have no context and therefore cannot be clustered.

The feature selection phase of the algorithm is shown in Fig. 2. On completion there are N unique queries to be clustered and at most K distinct keywords. K will be the length of the feature vector for each query; it is desirable to limit K to a manageable value. This is essential for a FAQ repository; even if the value of M, the number of documents per query, is 20, if the system stores 500 keywords per document, there could potentially be 10,000 keywords per query.

There are a number of available techniques for reducing the dimensionality of a data set; see the analysis by Forman for a comparison of several methods specifically for text classification problems [7]. However, these techniques all

1. Select N most frequently occurring stemmed queries from list of user queries
2. For each query, select first M matching documents (processing each entry to accumulate unique document IDs)
3. For list of n unique documents ($1 < n < N * M$), find list of all k associated single-word keywords in the document phrase index. Accumulate frequency count of each keyword in each document
4. Generate mapping from queries to distinct keywords and inverted mapping from keywords to queries
5. If $k > K$ (max # keywords), then
6. Sort keywords in ascending order by number of associated queries
7. Select and discard $k - K$ keywords

Fig. 2. Feature selection algorithm

1. discard all keywords that are associated with only a single query
2. if there are still keywords to discard
3. $C = \emptyset, D = \emptyset$
4. while $|C| < N$ and keywords remaining to examine
5. if next keyword's queries $\cap\, C ==$ next keyword's queries
6. $D = D \cup$ keyword's index
7. else
8. $C = C \cup$ indices of keyword's queries
9. for each keyword from the end of the list to the last one added to C, while still keywords to discard
10. discard keyword
11. while still keywords to discard
12. discard keyword with largest index in D

Fig. 3. Feature reduction algorithm

require that each document has a class label. Jain et al. suggest that in an unsupervised clustering situation, where the documents are unlabeled, only ad hoc feature selection methods are possible [8]. Feature extraction and dimension reduction methods such as Principle Component Analysis can be used on unclassified data (see the survey by Fodor [9] for details), and there are a number of other methods targeted specifically for use in text classification problems, including sequential search techniques such as sequential forward selection and sequential backward elimination (see the survey by Liu and Yu [10] for details). A promising approach based on expectation maximization is developed by Dy and Brodley [11].

However, these techniques are all computationally expensive. It is desirable to reduce the number of keywords via a method that can be evaluated quickly. A simple feature reduction algorithm is shown in Fig. 3; this algorithm comprises step 7 in Fig. 3. It is loosely similar to the feature selection method of Forman [7]. Forman used a "prune rare and common words" method based on the Zipf distribution.

The algorithm prunes the size of the list of keywords to the desired value K. The algorithm make use of the inverted mapping from keywords to queries created in step 4 of Fig. 2 and the sorted list of keywords created in step 6. Here C is the set of covered queries, or queries that are associated with a keyword that will be retained for the final feature vector. D is the set of keywords that can potentially be discarded.

The first heuristic employed by the algorithm is to immediately discard keywords that match only a single query. These rare keywords will have nearly no impact during the clustering process, because they cannot increase similarity with other queries. The second heuristic is to ensure that each query has at least one associated keyword. The keywords are evaluated starting with those associated with the fewest queries, and a covering set is accumulated. If any keyword is associated with a set of queries already in the covering set, it is a candidate for removal. The third heuristic is to discard additional keywords after enough have been checked to include all the queries in the covering set. These additional common keywords are discarded in descending order by their count of associated queries. The rational behind discarding keywords that are common across large number of queries is that these key-words will provide little differentiation during the clustering process.

After the feature reduction phase is complete, there are at most K keywords remaining. Once the set of keywords is known, a feature vector is created for each query. The vector contains a weight for each keyword relative to the query. There are a number of mechanisms for computing the weights; HAC + P + FSR uses a measure common to many other text clustering algorithms, the *tf-idf (term frequency/inverse document frequency)* metric. It is calculated using (1):

$$v_{i,j} = \begin{cases} (1 + log_2 tf_{i,j}) log_2 \frac{N}{n_j}, tf_{i,j} > 0 \\ 0, tf_{i,j} = 0 \end{cases} \quad (1)$$

where $v_{i,j}$ is the element corresponding to the jth keyword in the feature vector for the ith query, $tf_{i,j}$ is the number of times that keyword j occurred in the list of documents associated with query i, and n_j is the number of queries that are associated with keyword j. HAC + P + FSR weights keywords differently according to their position in the document; words in the title are counted 25 times, words in the keyword section are counted ten times, and all other words are counted once.

3.2 HAC Component

Once the feature vectors are computed, the HAC algorithm shown in Fig. 4 is used to create the dendrogram. Here, N is the number of queries to cluster, v_i is the array of feature vectors, C_i is the list of clusters, $f(i)$ is a flag indicating whether cluster i can be merged, and $S_{i,j}$ is the upper-triangular similarity matrix.

1. for $1 \leq i \leq N$
2. $\quad C_i = \{v_i\}$
3. $\quad f(i) = true$
4. for $1 \leq i \leq N$
5. $\quad$ for $i \leq j \leq N$
6. $\quad\quad S_{i,j} = sim_{AL}(v_i, v_j)$
7. for $1 \leq i \leq N$
8. $\quad$ choose most similar pair $\{C_a, C_b\}$ with $f(a) \wedge f(b)$ true
9. $\quad C_{n+i} = C_a \cup C_b, left(C_{n+i}) = C_a, right(C_{n+i}) = C_b$
10. $\quad f(n+i) = true, f(a) = false, f(b) = false$
11. for $1 \leq k \leq N + i - 1$
12. $\quad S_{k,n+i} = sim_{AL}(C_k, C_{n+i})$

Fig. 4. HAC clustering algorithm

The algorithm is simple: place each query into its own single-element cluster, then recursively select the two most similar clusters, join them together into a new cluster, and update the similarity matrix with the new cluster. The clusters are treated as the nodes in a binary tree, where the children of a cluster are the two clusters that were merged to form it.

There are a number of different mechanisms for computing the similarity between two clusters; HAC + P + FSR uses the average linkage inter-cluster similarity, which is defined in (2):

$$sim_{AL}(C_i, C_j) = \frac{1}{|C_i||C_j|} \sum_{v_a \in C_i} \sum_{v_b \in C_j} sim(v_a, v_b) \tag{2}$$

This similarity measure, also referred to in the literature as UPGMA, has been experimentally shown by Steinbach et al. to be the best choice for HAC clustering [12].

This calculation requires the computation of the similarity between two feature vectors; HAC+P+FSR uses the cosine distance metric, defined in (3):

$$sim(v_a, v_b) = \begin{cases} 0, \mathbf{a} = \mathbf{0}\ or\ \mathbf{b} = \mathbf{0} \\ \frac{\sum_{t_j \in T} \mathbf{v}_{a,j}, \mathbf{v}_{b,j}}{\sqrt{\sum_{t_j \in T} \mathbf{v}_{a,j}^2} \sqrt{\sum_{t_j \in T} \mathbf{v}_{b,j}^2}}, otherwise \end{cases} \tag{3}$$

The values of this metric are in the range $[0, 1]$, where 0 indicates no similarity and 1 indicates an exact match. This distance measure has been experimentally determined by Steinbach et al. [12] to be well suited for measuring the similarity between text document feature vectors.

Note that, as suggested by Dhillon et al. [13], and by Jain et al. [8], it is only necessary to evaluate the cosine similarity between each pair of documents once; these can be stored in an upper triangular matrix and used for all successive computations of the average link similarity.

3.3 Min–Max Partitioning (P)

Once the HAC algorithm has formed the dendrogram, the min–max partitioning algorithm in Fig. 5 is applied to flatten the tree. ϵ and ρ are parameters limiting the size of a cluster and the depth of the tree, respectively. These parameters are selected to tune the usability of the results; for instance, a depth between three and six levels is probably reasonable for most moderately complicated knowledge bases, and for a fairly large knowledge base, a minimum cluster size between three and ten queries might be reasonable.

$LC(l)$ is the set of clusters produced after cutting the dendrogram at level l, $CH(C_i)$ is the cluster hierarchy rooted at cluster C_i, $Q(C)$ is the cluster set *quality function*, and $N(C)$ is the *cluster number preference function*.

The set $LC(l)$ can be easily computed if the clusters are manipulated using their indices in the list C created by the HAC algorithm. In this list, clusters with larger indices are created after clusters with lower indices, and are therefore higher in the dendrogram tree. Starting with a set containing only the index of the root node of a subtree, at each cut level l the set $LC(l)$ is generated by replacing the largest index from the set with the indices of its left and right children.

Figure 6 shows a small dendrogram and illustrates how the tree would be cut at each level. Clusters are numbered in order of their creation – the root of the tree, $C9$, was created last, so the first cut level would produce $LC(1) = \{C7, C8\}$. Similarly, $LC(3) = \{C1, C2, C3, C6\}$. For each of these cut levels for a particular subtree, the q metric is computed, and after all cuts have been evaluated, the one that yields the most usable grouping is selected. The process is repeated recursively, with each cluster in the selected cut serving as the root of a new subtree to be partitioned.

1. **MinMaxPartition**$(d, C_1, \ldots, C_{2n-1})$
2. if $n < \epsilon$ or $d > \rho$ then
3. return $C_1, \ldots, C_n$
4. $minq = \infty, bestcut = 0$
5. $LC = \{C_{2n-1}\}$
6. for all cut levels $l, 1 \leq l < n$
7. $top = i \triangle \{i\} \in LC, i > j \forall j, \{j\} \in LC \wedge j \neq i$
8. $LC = LC - \{top\} + \{left(top), right(top)\}$
9. $q = \frac{Q(LC)}{N(LC)}$
10. if $q < minq$ then
11. $minq = q, bestcut = l, bestLC = LC$
12. for all $C_i \in LC(bestcut)$
13. $children(C_i) =$ **MinMaxPartition**$(d + 1, CH(C_i))$
14. return $bestLC$

Fig. 5. Min–max partitioning algorithm

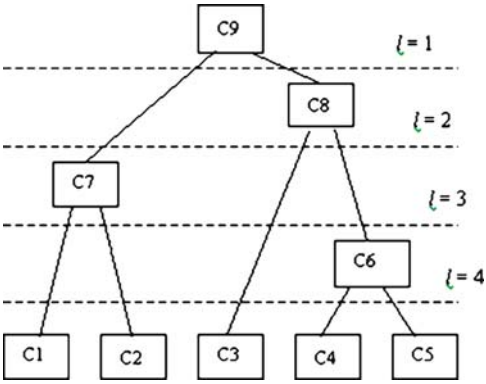


Fig. 6. Cut levels in partitioning

The cluster set quality function, $Q(C)$, is defined by (4):

$$Q(C) = \frac{1}{|C|} \sum_{C_i \in C} \frac{sim_{AL}(C_i, \bar{C}_i)}{sim_{AL}(C_i, C_i)}, \ \bar{C}_i = \bigcup_{k \neq i} C_k \tag{4}$$

The quality function in (4) is chosen to create cohesion between the clusters that are grouped together. The expression $sim_{AL}(C_i, \bar{C}_i)$ represents the inter-cluster pairwise similarity, while $sim_{AL}(C_i, C_i)$ represents the intra-cluster pairwise similarity. The algorithm chooses a cut of the tree that minimizes the inter-cluster similarity while maximizing the intra-cluster similarity, which should result in the highest cohesion within clusters.

The cluster number preference function, $N(C)$, is intended to help the algorithm form clusters that are of a size easily digestible by humans using the resulting hierarchy. It is defined by (5):

$$N(C) = f(|C|), \ f(x) = \frac{1}{\alpha! \beta^{\alpha}} x^{\alpha - 1} e^{-x/\beta} \tag{5}$$

This is just a simplified gamma distribution. The parameters α and β tune the smoothness of the preference function; HAC + P + FSR uses the values suggested in [5], $\alpha = 3$ and $\beta = Nclus/2$, where $Nclus$ is the ideal number of generated clusters per layer (empirically set to the square root of the number of objects in each partitioning step).

Alternative Quality Function

Although the average similarity metric has been experimentally shown to be a good choice for HAC clustering by Chuang and Chien [5], other similarity measures could potentially produce better results for partitioning the hierarchy. Two common alternatives are the single-linkage (SL) and complete-linkage (CL) functions, which are defined in (6) and (7), respectively.

$$sim_{SL}(C_i, C_j) = \max_{v_a \in C_i, v_b \in C_j} sim(v_a, v_b) \tag{6}$$

$$sim_{CL}(C_i, C_j) = \min_{v_a \in C_i, v_b \in C_j} sim(v_a, v_b) \tag{7}$$

The single-linkage similarity between two clusters is the similarity between the closest pair of points where one is in each cluster. The complete-linkage similarity is the similarity between the most distant pair of points where one is in each cluster.

The characteristic behavior of clustering using each of these similarity measures has been analyzed by Yager in [14]. He notes that the complete-linkage measure tends to generate larger clusters, with outlying elements placed in isolated small clusters. The single-linkage measure tends to form small, uniform-sized clusters and then join them together, potentially forming long odd-shaped clusters.

Given that the goal of the partitioning algorithm is to minimize inter-cluster similarity while maximizing intra-cluster similarity, an alternative quality function can be formulated as defined in (8).

$$Q(C) = \sum_{C_i \in C} \frac{\frac{1}{|C|} \sum_{C_i \in C, C_j \neq C_i} sim_{SL}(C_i, C_j)}{sim_{CL}(C_i, C_i)} \tag{8}$$

This function finds the average minimum distance between each cluster and each of the other clusters, divided by the average minimum pairwise similarity within each cluster. The denominator is inversely proportional to the average diameter of the clusters.

As with (4), this function has its minimum value when clusters are very cohesive and are well-separated. It can thus be used interchangeably with (4) in the min–max partitioning algorithm given in Fig. 5. In the following analysis, if the algorithm is run with this alternate quality function, it is referred to as HAC + P2 + FSR.

4 Evaluation

The HAC (hierarchical agglomerative clustering) and P (min–max partitioning) components of HAC+P and HAC+P+FSR are very similar, although the technique for determining $LC(l)$ in our partitioning algorithm has been clarified. The evaluation criteria are different, due to our goal of producing a hierarchy from a real production data set with no pre-assigned classifications.

The key difference between HAC + P and HAC + P + FSR is the feature selection and feature reduction mechanism, which is essential for our professional implementation. HAC + P + FSR takes advantage of the availability of all necessary context information for each query in the FAQ repository, rather than relying on the extraction of information from an external source

as assumed in [5]. This generates clusters that are more domain-specific. The repository contains all keywords extracted from each document, so the feature vectors contain a much more complete context for each query. Furthermore, HAC+P+FSR is not sensitive to the length of the text segments returned by a search engine for each query, and it does not have to process each text segment to extract keywords.

Also, HAC + P does not include a method for feature reduction. Our experience indicates that in an FAQ system of even moderate size, this step is essential for HAC + P + FSR to maintain a reasonable length for the feature vectors. The feature reduction algorithm takes advantage of some straightforward heuristics to reduce the feature set with very modest computational requirements.

In [5], Chuang and Chien used a combination of techniques to evaluate the performance of their algorithm. The performance of the algorithm was analyzed using the *F-measure*, which is a common technique for computing a performance measure for classification. However, computing this analytic measure for the clusters generated by the HAC + P + FSR algorithm from a real production data set is difficult if not impossible; the F-measure requires that the documents are pre-classified. This expert categorization is not available for a general RightNow FAQ system and in fact occurs rarely in real-word data repositories, due to the size of the data sets and their dynamic nature. This was the motivation for the development of clustering algorithms as an alternative to classification methods for data sets that don't include class information.

In the absence of any external category information, such as class labels, the cohesiveness of clusters can be used as a measure of cluster similarity. One measure of cluster cohesiveness is the intra-cluster similarity or self-similarity, $sim_{AL}(C, C)$, as computed in the cluster set quality function. This value for the cluster is equivalent to the squared length of the cluster centroid, $||C||^2$ [12].

In a production system, a subjective evaluation of the results produced by the algorithm is a more important measure of performance. The goal of the algorithm is to produce a concept hierarchy that users will find useful in describing the contents of the FAQ repository. In order to generate a subjective evaluation, HAC + P + FSR was used to cluster queries submitted to several RightNow systems. Table 1 includes some summary statistics on the analyzed repositories.

The algorithm was executed on each repository using the following parameter values: N (number of queries) = 1,000, K (number of keywords) = 2,500, ϵ (minimum cluster size) = 5, ρ (maximum hierarchy depth) = 6, α (smoothing parameter) = 3, β (smoothing parameter) = -1, which causes it to be computed dynamically during clustering. The value of β at each level of recursion is set to the square root of the number of queries included in the leaves of the subtree being partitioned.

Table 1. Analyzed system statistics

FAQ DB name	Fed. Gov. Dept.	Consumer electronics	Consumer software
Dates	3/18-6/07	5/08-6/07	3/28-6/15
# of Documents	657	2,418	368
# Words in docs	38,426	231,089	24,461
# Unique words in docs	4,008	7,231	3,951
# of searches	779,578	5,701	106,006
# Unique search words	83,105	2,545	15,946

Table 2. Clustering of 1,000 queries

Level	# clusters	Avg. self-similarity	Min. self-similarity
		Federal Government Department	
1	17	0.777	0.057
2	33	0.355	0.117
3	128	0.542	0.173
4	207	0.642	0.246
5	165	0.720	0.343
6	60	0.770	0.463
		Consumer Electronics Manufacturer	
1	35	0.465	0.038
2	117	0.459	0.121
3	256	0.655	0.269
4	173	0.748	0.333
5	55	0.762	0.462
6	7	0.813	0.502
		Consumer Software Producer	
1	24	0.524	0.094
2	62	0.526	0.160
3	135	0.570	0.251
4	187	0.679	0.338
5	138	0.744	0.417
6	48	0.816	0.527

Table 2 includes statistics for the final concept hierarchies produced by the algorithm. Note that when accumulating the maximum self-similarity, values of 1.0 were ignored; these similarity values were produced for any single-query clusters.

As expected, as depth in the tree increased, the cohesiveness of the clusters also increased. The slight anomalies at the top level were due to the decreasing number of clusters at that level and to the handling of single-query clusters; the unity self-similarity values had an adverse effect on the results.

In addition to the statistics, the resulting hierarchies were subjectively evaluated. The results were deemed very useful; the algorithm indeed clusters

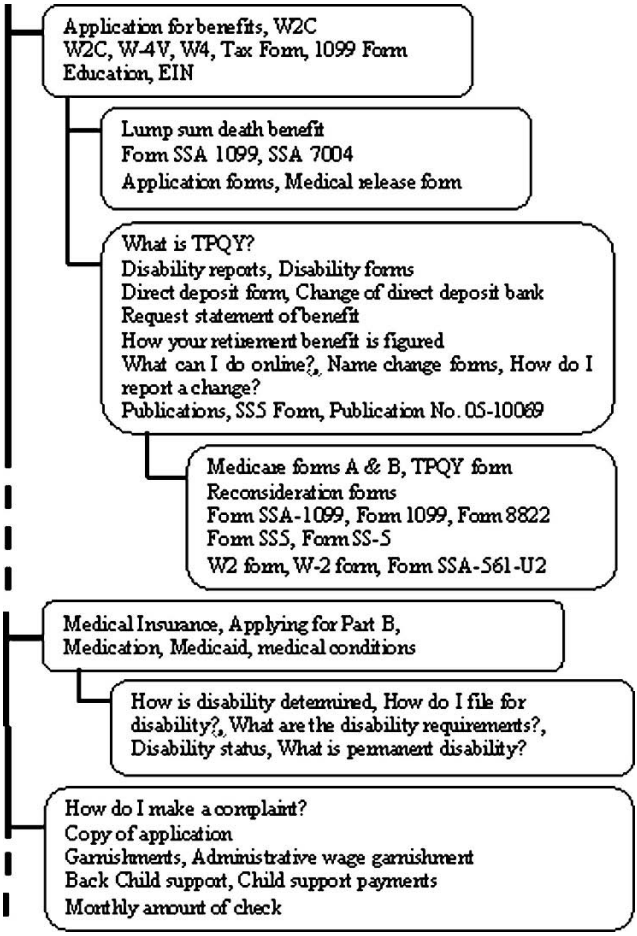


Fig. 7. Example showing one branch of hierarchy for federal government department's system

closely related queries at the deepest levels of the hierarchy, and shallower levels become more general. Figure 7 shows portions of the hierarchy that was generated from the first repository.

Two queries within one of the clusters, "Disability determination" and "What is permanent disability", were run against the knowledge base, and of the 20 most relevant documents returned in each result set, 90% were the same documents. Similarly, comparing the top 20 documents returned by a search on "TPQY Form" and the 15 documents returned by a search on "Reconsideration Form" shows that all 15 of the documents in the second set are in the first set.

4.1 Results Generated by Alternate Partitioning Algorithm

The HAC + P2 + FSR algorithm was executed on the same data sets with the same parameters to obtain a comparison of the two partitioning techniques. The results are shown in Table 3.

Note that although different cluster similarity measures were used in the partitioning process, the average and minimum self-similarities in the following table were using the average similarity measure. This allows for direct comparison with the results given for HAC + P + FSR.

Note that although the maximum depth parameter for these runs was set to six, just like the runs of the HAC+P+FSR algorithm, the use of the alternate quality function naturally led the algorithm to produce a flatter, bushier hierarchy without reaching the depth limit. As expected, the average and minimum self-similarity for each cluster increased with depth in the tree. However, comparison of Tables 2 and 3 show that the similarity measures for the clusters produced by HAC+P2+FSR were better overall at each level of the generated hierarchies, except for the first level. This is explained by the anomalies introduced in the original runs with HAC + P + FSR by single-query clusters. HAC + P2 + FSR generated more first-level clusters, so the effect of single-query clusters was diluted in the average values.

HAC + P2 + FSR tends to create a larger number of clusters at each level, making the tree shallower and bushier than HAC+P+FSR. In addition, the implementation of HAC+P2+FSR takes significantly less CPU time during the partitioning step.

Table 3. Alternate clustering

Level	# clusters	Avg. self-similarity	Min. self-similarity
	Federal Government Department		
1	48	0.511	0.117
2	184	0.583	0.204
3	292	0.742	0.331
4	100	0.863	0.557
	Consumer Electronics Manufacturer		
1	48	0.441	0.064
2	234	0.603	0.221
3	300	0.810	0.333
4	72	0.879	0.518
	Consumer Software Producer		
1	48	0.502	0.131
2	179	0.654	0.251
3	248	0.755	0.375
4	159	0.850	0.417
5	31	0.918	0.589

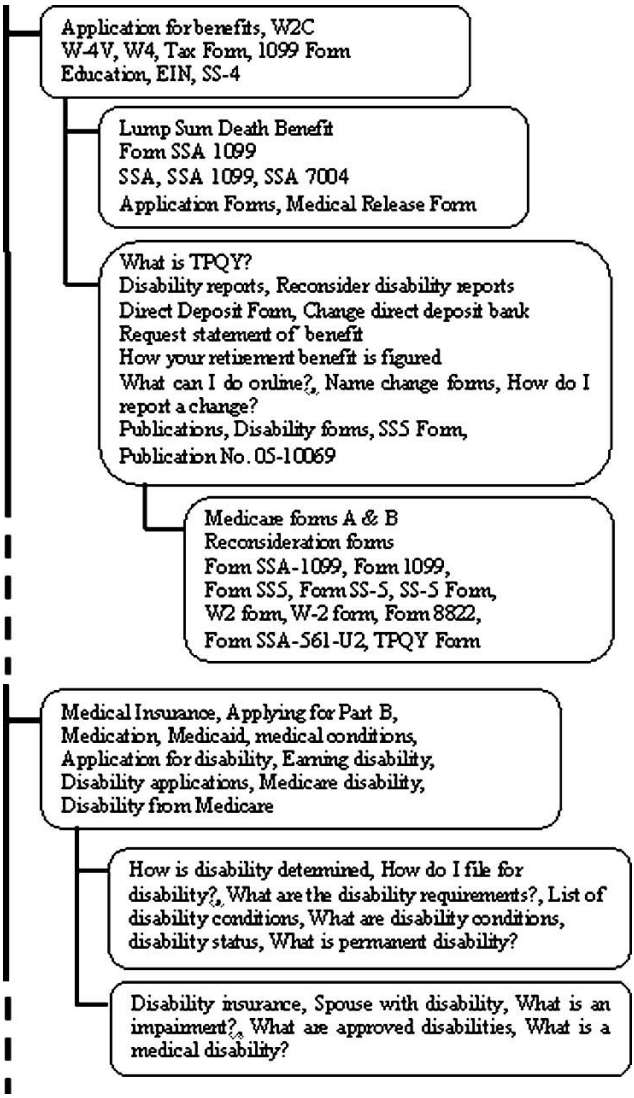


Fig. 8. Alternate branch of hierarchy for federal government system

Subjective analysis of the clustering created by HAC + P2 + FSR indicates that it is as usable as the clustering produced by HAC + P + FSR. Figure 8 shows a portion of the hierarchy generated from the government agency data; the top branch is the same as the branch produced by the HAC+P+FSR algorithm, but there are some slight differences in the lower branch. Some additional related queries were included in the clusters. Subjective review suggests that the clustering is somewhat more inclusive.

The power of using the documents returned by queries to cluster them is well demonstrated by this portion of the hierarchy. For instance, queries on "Lump sum death benefit" and "SSA 7004" produce result sets with only one overlap in the twenty most relevant documents returned. The same is true for the queries "Medical release form" and "Lump sum death benefit". However, the most relevant documents returned by each of these queries discuss the forms required when a Social Security recipient has a status-changing event.

The differences between the last two clusters shown here are not readily apparent, without actually retrieving the query documents and breaking them into their associated feature vectors. However, they do get merged into the same parent cluster, which adds other queries related to disability benefits. If a user is drilling down through the hierarchy, this branch would definitely be helpful in answering questions related to disabilities.

5 Conclusions and Future Work

Preliminary results show that the HAC+P+FSR algorithm performs well, producing very usable concept hierarchies by clustering the user queries from a RightNow FAQ system. The alternative partitioning technique used in HAC + P2 + FSR appears to produce a preferable clustering and requires less processor resources, suggesting that it is clearly the better choice. There is still work to do to fine-tune the algorithm, but this prototype definitely served as a proof of concept and will be used as the starting point for a production implementation.

There are a few interesting avenues for further exploration: one of the primary needs for a production presentation is labeling or otherwise identifying the clusters. Text summarization is a very complex problem, but the usability of the generated concept hierarchy would be enhanced by concise naming of the nodes in the hierarchy. Some method other than picking the most frequently occurring terms in a cluster is desirable. In [15], Mandhani et al. suggest a technique to co-cluster documents and words; it may be possible to maintain a similar keyword density matrix that could suggest the best terms to use to label each cluster.

Another interesting research challenge, more theoretical in nature, is to find a suitable objective measure to evaluate clustering performance in the absence of pre-classified data. Also, alternative techniques for feature selection and reduction would likely improve clustering. Finally, it would be interesting to investigate adapting the work described by Frigui and Masraoui in [16] to the query clustering algorithm. They discuss a technique for clustering documents and simultaneously adjusting a set of keyword weights for the cluster; upon completion, the associated weights indicate the relevance of each keyword for the cluster, and may help to select terms to describe the cluster. The algorithm can also be adjusted to perform fuzzy clustering; adapting this technique would allow each query to be included in multiple clusters.

Acknowledgements

Bob Wall and Neal Richter would like to thank RightNow Technologies and particularly the other members of the knowledge base group at RightNow (Dr. Steve Durbin, Zuzana Gedeon, and Doug Warner) for their suggestions, assistance, and support.

Rafal Angryk would like to thank the Montana NASA EPSCoR Grant Consortium for sponsoring this research. This work was partially supported by the NASA Grant Consortium Award No. M166-05-Z3184.

References

1. Spangler S, Kreulen J (2001) Knowledge base maintenance using knowledge gap analysis. In: Proceedings of SIGKDD'01, San Francisco, CA, August, 2001, pp. 462–466
2. Sanderson M, Croft B (1999) Deriving concept hierarchies from text. In: Proceedings of SIGIR'99, Berkeley, CA, August, 1999, pp. 206–213
3. Cilibrasi R, Vitanyi P. Automatic meaning discovery using Google. Published on Web, available at http://arxiv.org/abs/cs/0412098
4. Chuang S-L, Chien L-F (2002) Towards automatic generation of query taxonomy: a hierarchical query clustering approach. In: Proceedings of ICDM'02, Maebashi City, Japan, December 9–12, 2002, pp. 75–82
5. Chuang S-L, Chien L-F (2004) A practical web-based approach to generating topic hierarchy for text segments. In: Proceedings of CIKM'04, Washington DC, November, 2004, pp. 127–136
6. Larsen B, Aone C (1999) Fast and effective text mining using linear-time document clustering. In: Proceedings of SIGKDD'99, San Diego, CA, August, 1999, pp. 16–22
7. Forman G (2003) An extensive empirical study of feature selection metrics for text classification. In: Journal of machine learning research, vol. 3, 2003, pp. 1289–1305
8. Jain A, Murty M, Flynn P (1999) Data clustering: a review. In: ACM computing surveys, vol. 31, no. 3, September, 1999, pp. 264–323
9. Fodor IK (2002) A survey of dimension reduction techniques. LLNL technical report, June 2002, UCRL-ID-148494 (available at http://www.llnl.gov/CASC/sapphire/pubs/148494.pdf)
10. Liu H, Yu L (2005) Toward integrating feature selection algorithms for classification and clustering. In: IEEE transactions on knowledge and data engineering, vol. 17, no. 4, April, 2005, pp. 491–502
11. Dy JG, Brodley CE (2005) Feature selection for unsupervised learning. In: Journal of machine learning research, vol. 5, 2005, pp. 845–889
12. Steinbach M, Karypis G, Kumar V (2000) A comparison of document clustering techniques. In: KDD workshop on text mining, 2000
13. Dhillon I, Fan J, Guan Y (2001) Efficient clustering of very large document collections. In: Grossman R, Kamath G, Naburu R (eds) Data mining for scientific and engineering applications, Kluwer, Boston

14. Yager RR (2000) Intelligent control of the hierarchical agglomerative clustering process. In: IEEE transactions on systems, man, cybernetics, part B, vol. 30, no. 6, December 2000, pp. 835–845
15. Mandhani B, Joshi S, Kummamuru K (2003) A matrix density based algorithm to hierarchically co-cluster documents and words. In: Proceedings of WWW2003, May 20–24, 2003, Budapest, Hungary, pp. 511–518
16. Frigui H, Masraoui O (2004) Simultaneous clustering and dynamic keyword weighting for text documents. In: Berry, MW (ed) Survey of text mining: clustering, classification, and retrieval, Springer, Berlin Heidelberg New York, 2004, pp. 45–72

Mining Efficiently Significant Classification Association Rules

Yanbo J. Wang[1], Qin Xin[2], and Frans Coenen[1]

[1] Department of Computer Science, University of Liverpool, Ashton Building, Ashton Street, Liverpool, L69 3BX, UK
jwang@csc.liv.ac.uk, frans@csc.liv.ac.uk
[2] Department of Informatics, University of Bergen, P.B.7800, N-5020 Bergen, Norway
xin@ii.uib.no

Summary. Classification Rule Mining (CRM) is a well-known Data Mining technique for the extraction of hidden Classification Rules (CRs) from a given database that is coupled with a set of pre-defined classes, the objective being to build a classifier to classify "unseen" data-records. One recent approach to CRM is to employ Association Rule Mining (ARM) techniques to identify the desired CRs, i.e. Classification Association Rule Mining (CARM). Although the advantages of accuracy and efficiency offered by CARM have been established in many papers, one major drawback is the large number of Classification Association Rules (CARs) that may be generated – up to a maximum of "$2^n - n - 1$" in the worst case, where n represents the number of data-attributes in a database. However, there are only a limited number, say at most $\hat{k}$ in each class, of CARs that are required to distinguish between classes. The problem addressed in this chapter is how to efficiently identify the $\hat{k}$ such CARs. Having a CAR list that is generated from a given database, based on the well-established "Support-Confidence" framework, a rule weighting scheme is proposed in this chapter, which assigns a score to a CAR that evaluates how significantly this CAR contributes to a single pre-defined class. Consequently a rule mining approach is presented, that addresses the above, that operates in time $O(k^2n^2)$ in its deterministic fashion, and $O(kn)$ in its randomised fashion, where k represents the number of CARs in each class that are potentially significant to distinguish between classes and $k \geq \hat{k}$; as opposed to exponential time $O(2^n)$ – the time required in score computation to mine all $\hat{k}$ CARs in a "one-by-one" manner. The experimental results show good performance regarding the accuracy of classification when using the proposed rule weighting scheme with a suggested rule ordering mechanism, and evidence that the proposed rule mining approach performs well with respect to the efficiency of computation.

Y.J. Wang et al.: *Mining Efficiently Significant Classification Association Rules*, Studies in Computational Intelligence (SCI) **118**, 443–467 (2008)
www.springerlink.com

1 Introduction

Data Mining [29,30] is a promising area of current research and development in Computer Science, which is attracting more and more attention from a wide range of different groups of people. Data Mining aims to extract various types of hidden and interesting knowledge (i.e., rules, patterns, regularities, customs, trends, etc.) from databases, where the volume of a collected database can be very large. In Data Mining, common types of mined knowledge include: Association Rules (ARs) [1], Classification Rules (CRs) [45], Classification Association Rules (CARs) [3], Prediction Rules (PRs) [28], Clustering Rules (CTRs) [42], Sequential Patterns (SPs) [52], Emerging Patterns (EPs) [21], etc.

An AR describes a co-occurring relationship between binary-valued data-attributes, expressed in the form of an "antecedent ⇒ consequent" rule. Association Rule Mining (ARM) [2], with its wide range of applications, has been well-established in Data Mining in the past decade. It aims to identify all ARs in a given transaction-database D_T. One application of ARM is to define CRs, from a training-dataset D_R that is coupled with a set of pre-defined classes $C = \{c_1, c_2, \ldots, c_{|C|-1}, c_{|C|}\}$, which can be used to classify the data-records in a test-dataset D_E. This kind of AR based CR is referred to as CAR. In general the process to build a classifier using identified CRs is called Classification Rule Mining (CRM) [45], which is another well-known Data Mining technique parallelling ARM. In CRM, a class-database D_C is given as $D_R \cup D_E$, where D_R and D_E share the same data-attributes but the class-attribute (the last data-attribute in D_R) is "unseen" in D_E.

Classification Association Rule Mining (CARM) [3] is a recent CRM approach that builds an ARM based classifier using CARs, where these CARs are generated from a given transaction-training-dataset $D_{TR} \subset D_{TC}$, and D_{TC} is a D_C in a "transactional" manner. In [18], Coenen et al. suggest that results presented in [36,38] show that CARM seems to offer greater accuracy, in many cases, than other methods such as C4.5 [45]. However, one major drawback of this approach is the large number of CARs that might be generated – up to a maximum of "$2^n - n - 1$" in the worst case, where n represents the number of data-attributes in D_{TC}. In [53], Yin and Han believe that there are only a limited number, say at most $\hat{k}$ in each class, of CARs that are required to distinguish between classes and should be thus used to make up a classifier. They suggest a value of 5 as an appropriate value for $\hat{k}$, and employ the *Laplace expected error estimate* [10] to estimate the accuracy of CARs. In [15] Coenen and Leng evaluated a number of alternative rule ordering and case satisfaction strategies, and conclude that for lower confidence thresholds (i.e., 50–75%) CSA (Confidence-Support-size_of_Antecedent) and *Laplace* ordering coupled with a "best first" case satisfaction mechanism can achieve better accuracy than comparable alternatives.

1.1 Contribution

Given a CAR list $\mathcal{R} = \{R_1, R_2, \ldots, R_{N-1}, R_N\}$ that is generated from a given D_{TR} based on the well-established "Support-Confidence" framework, where

$\mathcal{R}$ is presented using CSA ordering, and N represents the size of $\mathcal{R}$ that can be as large as "$2^n - n - 1$", a rule weighting scheme is proposed in this chapter, which assigns a score to a CAR $R_j \in \mathcal{R}$ that represents how significantly R_j contributes to a single class $c_i \in C$. An alternative rule ordering mechanism is consequently introduced, based on the proposed rule weighting scheme, that aims to improve the performance of the well-established CSA ordering regarding the accuracy of classification. In [19] a general framework for selectors, namely (k, m, n)-selectors, was proposed with applications in optimal group testing. In this chapter, a similar concept of selectors is further considered in a randomised setting. This randomised selector can be proved to exist with a high probability. With regards to the concept of selectors, a novel rule mining approach is presented that addresses the problem of mining the $\hat{k}$ "significant rules" (see Definition 9 in Sect. 3.2) in $\mathcal{R}$. The rule mining approach operates in time $O(k^2n^2)$ in its deterministic fashion, and $O(kn)$ in its randomised fashion, where k represents the number of CARs in each class that can potentially be used to distinguish between classes and $k \geq \hat{k}$; as opposed to exponential time $O(2^n)$ – the time required in score computation to find all $\hat{k}$ significant (the "best $\hat{k}$") rules in $\mathcal{R}$ in a "one-by-one" manner. The experimental results show that the proposed rule weighting and rule ordering approaches perform well regarding the accuracy of classification, and evidence the fast computational efficiency of running the randomised rule mining approach. Note that the deterministic rule mining approach is theoretical only.

1.2 Chapter Organisation

The following section describes some Data Mining work that relate to CARM. In Sect. 3 we first introduce the rule weighting scheme together with a rule ordering mechanism based on the rule weighting scheme; and sketch the concept of deterministic selectors and give an introduction to the concept of randomised selectors. In Sect. 4 we propose a rule mining approach to efficiently mine the "best $\hat{k}$" CARs in $\mathcal{R}$. In Sect. 5 we present experimental results obtained using the TFPC (Total From Partial Classification) CARM algorithm [15, 18] coupled with a "best first" case satisfaction approach. Finally we discuss our conclusions in Sect. 6, and a number of open issues for further research.

2 Related Work

2.1 Association Rule Mining

ARM extracts a set of ARs from D_T, first introduced in [1]. Let $I = \{a_1, a_2, \ldots, a_{n-1}, a_n\}$ be a set of items (data-attributes), and $\mathcal{T} = \{T_1, T_2, \ldots, T_{m-1}, T_m\}$ be a set of transactions (data-records), D_T is described by $\mathcal{T}$, where

each $T_i \in \mathcal{T}$ contains a set of items $I' \subseteq I$. In ARM, two threshold values are usually used to determine the significance of an AR:

- *Support:* The frequency that the items occur or co-occur in $\mathcal{T}$. A support threshold σ, defined by the user, is used to distinguish frequent items from the infrequent ones. A set of items S is called an itemset, where $S \subseteq I$, and $\forall a_i \in S$ co-occur in $\mathcal{T}$. If the occurrences of some S in $\mathcal{T}$ exceeds σ, we say that S is a Frequent Itemset (FI).
- *Confidence:* Represents how "strongly" an itemset X implies another itemset Y, where $X, Y \subseteq I$ and $X \cap Y = \{\oslash\}$. A confidence threshold α, supplied by the user, is used to distinguish high confidence ARs from low confidence ARs.

An AR $X \Rightarrow Y$ is valid when the support for the co-occurrence of X and Y exceeds σ, and the confidence of this AR exceeds α. The computation of support is $\frac{X \cup Y}{|\mathcal{T}|}$, where $|\mathcal{T}|$ is the size function of the set $\mathcal{T}$. The computation of confidence is $\frac{Support(X \cup Y)}{Support(X)}$. Informally, $X \Rightarrow Y$ can be interpreted as "if X exists, it is likely that Y also exists". With regards to the history of ARM investigation, three major categories of serial (non-parallel) ARM algorithms can be identified: (1) mining ARs from all possible FIs, (2) mining ARs from Maximal Frequent Itemsets (MFIs), and (3) mining ARs from Frequent Closed Itemsets (FCIs).

Mining ARs from FIs

In the past decade, many algorithms have been introduced that mine ARs from identified FIs. These algorithms can be further grouped into different "families", such as Pure-apriori like, Semi-apriori like, Set Enumeration Tree like, etc.

- *Pure-apriori like* where FIs are generated based on the generate-prune level by level iteration that was first promulgated in the Apriori algorithm [2]. In this "family" archetypal algorithms include: Apriori, AprioriTid and AprioriHybrid [2], Partition [49], DHP [43], Sampling [50], DIC [7], CARMA [31], etc.
- *Semi-apriori like* where FIs are generated by enumerating candidate itemsets but do not apply the Apriori generate-prune iterative approach founded on (1) the join procedure, and (2) the prune procedure that employs the closure property of itemsets – if an itemset is frequent then all its subsets will also be frequent; if an itemset is infrequent then all its supersets will also be infrequent. In this "family" typical algorithms include: AIS [1], SETM [33], OCD [40], etc.
- *Set Enumeration Tree like* where FIs are generated through constructing a set enumeration tree structure [48] from D_T, which avoids the need to enumerate a large number of candidate itemsets. In this "family" a number of approaches can be further divided into two main streams: (1)

Apriori-TFP[1] based (i.e., [11–13, 16, 17], etc.), and (2) FP-tree based (i.e., [9, 24, 27, 39], etc.).

Mining ARs from MFIs

It is apparent that the size of a complete set of FIs can be very large. The concept of MFI [47] was proposed to find several "long" (super) FIs in D_T, which avoids the redundant work required to identify "short" FI. The concept of vertical mining has also been effectively promoted in this category [54]. Vertical mining, first mentioned in [32], deals with a vertical transaction database D_T^V, where each data-record represents an item that is associated with a list of its relative transactions (the transactions in which it is present). Typical MFI algorithms include: MaxEclat / Eclat [54], MaxClique / Clique [54], Max-Miner [47], Pincer-Search [37], MAFIA [8], GenMax [26], etc.

Mining ARs from FCIs

Algorithms belonging to this category extract ARs through generating a set of FCIs from D_T. In fact the support of some sub-itemsets of an MFI might be hard to identified resulting in a further difficulty in the computation of confidence. The concept of FCI [44] is proposed to improve this property of MFI, which avoids the difficulty of identifying the support of any sub-itemsets of a relatively "long" FI. A FCI f is an itemset $S \in D_T$, where f is frequent and $\neg\exists$ itemset $f' \supset f$ and f' shares a common support with f. The relationship between FI, MFI and FCI is that MFI $\subseteq$ FCI $\subseteq$ FI [8]. In this category typical algorithms include: CLOSET [44], CLOSET+ [51], CHARM [55], MAFIA [8], etc.

2.2 Classification Rule Mining

CRM deals with D_C, where D_C is founded as $D_R \cup D_E$. It discovers a set of CRs in D_R from which to build a classifier to classify "unseen" data records in D_E. A D_R consists of n data-attributes and m data records. By convention the last data-attribute in each data-record usually indicates its pre-defined class, noted as the class attribute. CRM can thus be described as the process of assigning a Boolean value to each pair $(d_j, c_i) \in D_E \times C$, where each $d_j \in D_E$ is an "unseen" data-record, C as declared in Sect. 1 is a set of pre-defined classes, and (d_j, c_i) is a data-record in D_E to be labeled.

The Cover Algorithm

In CRM a number of approaches have been proposed to generate a classifier from a set of training data-records. For example the Cover Algorithm [41] takes D_R as its input and aims to generate a complete set of minimal non-redundant CRs. We define the Cover algorithm in Fig. 1 as follows.

[1] Apriori-TFP and its related softwares may be obtained from http://www.csc.liv.ac.uk/~frans/KDD/Software.

Algorithm COVER;
input: D_R (a training-dataset);
output: the set $\mathcal{S}_{\mathcal{CR}}$ (the complete set of minimal non-redundant CRs);
(1)begin
(2) $\mathcal{S}_{\mathcal{CR}} := \{\oslash\}$;
(3) while $D_R \neq \{\oslash\}$ do
(4) find a CR cr from D_R heuristically;
(5) remove all records identified by cr from D_R;
(6) $\mathcal{S}_{\mathcal{CR}} \leftarrow \mathcal{S}_{\mathcal{CR}} \cup cr$;
(7) end while
(8) return ($\mathcal{S}_{\mathcal{CR}}$);
(9)end

Fig. 1. The cover algorithm

Existing CRM Approaches

With regards to the history of CRM investigation, various mechanisms on which CRM algorithms have been based include: Decision Trees [45], Bayesian Approach [20], K-Nearest Neighbour [34], Support Vector Machine [6], Association Rules [38], Emerging Patterns [22], Genetic Algorithm [25], Neural Networks [28], Case-based Reasoning [28], Rough Set [28], Fuzzy Set [28], Simple Approach [23], etc. In this section, we briefly describe four of the most well-known mechanisms used in CR generation as follows.

- *Decision Trees:* Where CRs are mined based on a greedy algorithm. The approach can be separated into two stages where a flow chart like tree structure is constructed from D_R first (stage 1) followed by a tree pruning phase; the pruned tree is then used in CR generation (stage 2). C4.5 [45] is the most famous Decision Tree based CRM method and operates by recursively splitting D_R on the attribute that produces the *maximum gain* to generate the decision tree. This tree is then pruned according to an error estimate. The result is used to classify "unseen" data.
- *Bayesian Approach:* The typical mechanism found in Byesian CRM approaches is naive bayes, which has been widely applied in Machine Learning. The general idea of naive bayes is to make use of knowledge of the joint probabilities that exist between attributes in training-dataset so as to produce a model of some machine learning application that can be applied to "unseen" data. The term naive is used to refer to the assumption that the conditional probability between data-attributes is independent of the conditional probability between other data-attributes. A naive bayes classifier is built using D_R, which comprises a set of conditional probabilities for each data-attribute $a_h \in I_R$ (the set of attributes in D_R) and each class $c_i \in C$, so that there are $|I_R| \times |C|$ probabilities. This set of conditional probabilities is then used to classify "unseen" data-records in D_E.
- *K-Nearest Neighbour:* K-Nearest Neighbour (K-NN) is a well-known statistical approach used in CRM, which classifies an "unseen" data-record

$d_{Ei} \in D_E$, by summarising a common pre-defined class from its K most similar instances, identified in D_R. To identify the K most similar training-instances for d_{Ei}, calculating the Euclidean distance value between each training data-record $d_{Ri} \in D_R$ and d_{Ei} has been commonly used: $Distance(d_{Ri}, d_{Ei}) = \sqrt{(\sum_{j=1}^{n}(d_{Ri_j} - d_{Ei_j})^2)}$, where d_{Ri_j} and d_{Ei_j} are the values of the jth data-attribute in D_C for d_{Ri} and d_{Ei}.

- *Support Vector Machine:* The objective of using Support Vector Machine (SVM) [6] is to find a hypothesis $\tilde{h}$ which minimises the *true error* defined as the probability that $\tilde{h}$ produces an erroneous result. SVM make use of linear functions of the form: $f(x) = w^T x + b$, where w is the weight vector, x is the input vector, and $w^T x$ is the inner product between w and x. The main concept of SVM is to select a *hyperplane* that separates the positive and negative examples while maximising the smallest margin. Standard SVM techniques produce binary classifiers. Two common approaches to support the application of SVM techniques to the multi-class problem are One Against All (OAA) and One Against One (OAO).

2.3 Classification Association Rule Mining

An overlap between ARM and CRM is CARM, which strategically solves the traditional CRM problem by applying ARM techniques. It mines a set of CARs from D_{TR}. A CAR is an AR of the form $X \Rightarrow c_i$, where X is an FI mined from D_{TR}, and c_i is a pre-defined class in C to which data-records can be assigned. The idea of CARM was first presented in [3]. Subsequently a number of alternative approaches have been described. Broadly CARM algorithms can be categorised into two groups according to the way that the CRs are generated:

- *Two stage algorithms* where a set of CARs are produced first (stage 1), which are then pruned and placed into a classifier (stage 2). Examples of this approach include CBA [38] and CMAR [36]. CBA (Classification Based on Associations), developed by Liu et al. in 1998, is an Apriori [2] based CARM algorithm, which (1) applies its CBA-GR procedure for CAR generation; and (2) applies its CBA-CB procedure to build a classifier based on the generated CARs. CMAR (Classification based on Multiple Association Rules), introduced by Han and Jan in 2001, is similar to CBA but generates CARs through a FP-tree [27] based approach.
- *Integrated algorithms* where the classifier is produced in a single processing step. Examples of this approach include TFPC[2] [15,18], and induction systems such as FOIL [46], PRM and CPAR [53]. TFPC (Total From Partial Classification), proposed by Coenen et al. in 2004, is a Apriori-TFP [16] based CARM algorithm, which generates CARs through efficiently constructing both P-tree and T-tree set enumeration tree structures. FOIL

[2]TFPC may be obtained from http://www.csc.liv.ac.uk/~frans/KDD/Software.

(First Order Inductive Learner) is an inductive learning algorithm for generating CARs developed by Quinlan and Cameron-Jones in 1993. This algorithm was later developed by Yin and Han to produce the PRM (Predictive Rule Mining) CAR generation algorithm. PRM was then further developed, by Yin and Han in 2003 to produce CPAR (Classification based on Predictive Association Rules).

Case Satisfaction Approaches

Regardless of which particular methodology is used to build it, a classifier is usually presented as an ordered CAR list $\mathcal{R}$. In [15] Coenen and Leng summarised three case satisfaction approaches that have been employed in different CARM algorithms for utilising the resulting classifier to classify "unseen" data. These three case satisfaction approaches are itemised as follows (given a particular case):

- *Best First Rule:* Select the first "best" rule that satisfies the given case according to some ordering imposed on $\mathcal{R}$. The ordering can be defined according to many different ordering schemes, including: (1) CSA (Confidence-Support-size_of_Antecedent) – combinations of confidence, support and size of antecedent, with confidence being the most significant factor (used in CBA, TFPC and the early stages of processing of CMAR); (2) WRA (Weighted Relative Accuracy) – which reflects a number of rule "interestingness" measures as proposed in [35]; (3) Laplace Accuracy – as used in PRM and CPAR; (4) χ^2 Testing – χ^2 values as used, in part, in CMAR; (5) ACS (size_of_Antecedent-Confidence-Support) – an alternative to CSA that considers the size of the rule antecedent as the most significant factor; etc.
- *Best $\mathcal{K}$ Rules:* Select the first "best $\mathcal{K}$" rules (in this chapter we denote $\mathcal{K}$ by $\hat{k}$ as mentioned above) that satisfy the given case and then select a rule according to some averaging process as used for example, in CPAR. The term "best" in this case is defined according to an imposed ordering of the form described in *Best First Rule*.
- *All Rules:* Collect all rules in the classifier that satisfy the given case and then evaluate this collection to identify a class. One well-known evaluation method in this category is WCS (Weighted χ^2) testing as used in CMAR.

Rule Ordering Approaches

As noted in the previous section five existing rule ordering mechanisms are identified to support the "best first rule" case satisfaction strategy. Each can be further separated into two stages: (1) a rule weighting stage where each $R_j \in \mathcal{R}$ is labeled with a weighting score that represents the significance of R_j indicates a single class c_i; and (2) a rule re-ordering stage, which sorts the original $\mathcal{R}$ in a descending manner, based on the score assigned in stage (1), of

each R_j. With regards to both stages of rule weighting and rule re-ordering, each rule ordering mechanism can be described in more detail as follows:

- *CSA:* The CSA rule ordering mechanism is based on the well-established "Support-Confidence" framework (see Sect. 2.1). It does not assign an additional weighting score to each $R_j \in \mathcal{R}$ in its rule weighting stage, but simply gathers the values of confidence and support, and the size of the rule antecedent to "express" a weighting score for each $R_j \in \mathcal{R}$. In the rule re-ordering stage, CSA generally sorts the original $\mathcal{R}$ in a descending order based on the value of confidence of each R_j. For these rules in $\mathcal{R}$ that share a common value of confidence, CSA sorts them in a descending order based on their support value. Furthermore for these rules in $\mathcal{R}$ that share common values for both confidence and support, CSA sorts them in an ascending order based on their size of the rule antecedent. In this chapter, the "pure confidence" approach is applied as a simplified version of the CSA rule ordering mechanism, which sorts the original $\mathcal{R}$ in a descending order based on the value of confidence of each R_j only.
- *WRA:* The use of WRA can be found in [35], where this technique is used to determine an expected accuracy for each generated CR. In its rule weighting stage, WRA assigns a weighting score to each $R_j \in \mathcal{R}$. The calculation of the value of R_j, confirmed in [15], is: $wra(R_j) = support(R_j.antecedent) \times (confidence(R_j)$ - $support(R_j.consequent)$. In the rule re-ordering stage the original $\mathcal{R}$ is simply sorted in a descending order based on the assigned *wra* value of each R_j.
- *Laplace Accuracy:* The use of the *Laplace expected error estimate* [10] can be found in [53]. The principle of applying this rule ordering mechanism is similar to WRA. The calculation of the *Laplace* value of R_j is: $Laplace(R_j) = \frac{support(R_j.antecedent \cup R_j.consequent) + 1}{support(R_j.antecedent + |C|)}$, where $|C|$ is the size function of the set C.
- χ^2 *Testing:* χ^2 Testing is a well known technique in statistics, which can be used to determine whether two variables are independent of one another. In χ^2 Testing a set of observed values (O) is compared against a set of expected values (E) – values that would be estimated if there were no associative relationship between the variables. The value of χ^2 is calculated as: $\sum_{i=1}^{n} \frac{(O_i - E_i)^2}{E_i}$, where n is the number of observed/expected values, which is always 4 in CARM. If the χ^2 value between two variables (the antecedent and consequent of $R_j \in \mathcal{R}$) above a given threshold value (for CMAR the chosen threshold is 3.8415), thus it can be concluded that there is a relation between the rule antecedent and consequent, otherwise there is not a relation. After assigning a χ^2 value to each $R_j \in \mathcal{R}$, it can be used to re-order the $\mathcal{R}$ in a descending basis.
- *ACS:* The ACS rule ordering mechanism is a variation of CSA. It takes the size of the rule antecedent as its major factor (using a descending order) followed by the rule confidence and support values respectively. This rule ordering mechanism ensures that "specific rules have a higher precedence than more general rules" [15].

3 Preliminaries

As noted in Sect. 1.1, $\mathcal{R} = \{R_1, R_2, \ldots, R_{2^n-n-2}, R_{2^n-n-1}\}$ represents a complete set of possible CARs that are generated from D_{TR}, and R_j represents a rule in set $\mathcal{R}$ with label j.

3.1 Proposed Rule Weighting Scheme

Item Weighting Score

There are n items involved in D_{TR}. For a particular pre-defined class A (as $c_i \in C$), a score is assigned to each item in D_{TR} that distinguishes the significant items for class A from the insignificant ones.

Definition 1. *Let $\varsigma^A(Item_h)$ denote the contribution of each $item_h \in D_{TR}$ for class A, which represents how significantly $item_h$ determines A, where $0 \leq \varsigma^A(Item_h) \leq |C|$, and $|C|$ is the size function of the set C.*

The calculation of $\varsigma^A(Item_h)$ is given as follows:

$$\varsigma^A(Item_h) = (TransFreq(Item_h, A)) \times (1 - TransFreq(Item_h, \bar{A})) \times \frac{|C|}{ClassCount(Item_h,\ C)},$$

where

1. The $TransFreq(Item_h,\ A\ or\ \bar{A})$ function computes how frequently that $Item_h$ appears in class A or the group of classes $\bar{A}$ (the complement of A). The calculation of this function is: $\frac{number\ of\ transactions\ with\ Item_h\ in\ the\ class(es)}{number\ of\ transactions\ in\ the\ class(es)}$.
2. The $ClassCount(Item_h, C)$ function simply counts the number of classes in C which contain $Item_h$.

The rationale of this item weighting score is demonstrated as follows:

1. The weighting score of $Item_h$ for class A tends to be high if $Item_h$ is frequent in A.
2. The weighting score of $Item_h$ for class A tends to be high if $Item_h$ is infrequent in $\bar{A}$.
3. The weighting score of $Item_h$ for any class tends to be high if $Item_h$ is involved in a small number of classes in C. In [5], a similar idea can be found in feature selection for text categorisation.

Rule Weighting Score

Based on the item weighting score, a weighting score is assigned to the rule antecedent of each $R_j \in \mathcal{R}$.

Definition 2. *Let* $\varsigma^A(R_j)$ *denote the contribution of each CAR* $R_j \in \mathcal{R}$ *for class A, which represents how significantly* R_j *determines A.*

The calculation of $\varsigma^A(R_j)$ is given as follows:

$$\varsigma^A(R_j) = \sum_{h=1}^{|R_j|} \varsigma^A(Item_h \in R_j).$$

3.2 Some Definitions

Definition 3. *If* $\varsigma^A(Item_h) < \varepsilon$, *we recognise* $Item_h \in D_{TR}$ *as a light-weighted item for class A, where* ε *is a user-defined constant and* $0 \leq \varepsilon \leq 1$. *We use* $I'(A) = \{Item'_1, Item'_2, \ldots, Item'_{|I'(A)|-1}, Item'_{|I'(A)|}\}$ *to denote the sufficient set of light-weighted items for A, identified in* D_{TR}.

Definition 4. *If a CAR* $R_j \in \mathcal{R}$ *significantly satisfies the following inequality,* $\varsigma^A(R_j) > \sum_{h=1}^{|I'(A)|} \varsigma^A(Item_h \in I'(A))$, *where the contribution of* R_j *for class A is significantly greater than the sum of the contributions of all light-weighted items for class A, we recognise* R_j *as a heavy-weighted rule for A. We use* $\mathcal{R}'(A) = \{R'_1, R'_2, \ldots, R'_{t-1}, R'_t\}$ *to denote the set of selected heavy-weighted rules for A, identified in* $\mathcal{R}$. *In* $\mathcal{R}$, *we always select the top-t heavy-weighted rules to construct* $\mathcal{R}'(A)$, *where integer t is a user-defined constant.*

Definition 5. *We recognise an item* $Item_h \in D_{TR}$ *as a heavy-weighted rule-item for class A if* $Item_h \in (\exists R'_j \in \mathcal{R}'(A))$. *We use* $I''(A) = \{Item''_1, Item''_2, \ldots, Item''_{|I''(A)|-1}, Item''_{|I''(A)|}\}$ *to denote the sufficient set of heavy-weighted rule-items for A, identified in* D_{TR}.

Definition 6. *If a CAR* $R_j \in \mathcal{R}$ *does not contain any item* $Item''_h \in I''(A)$, *we recognise* R_j *as a noisy rule for class A. We use* $\mathcal{R}''(A) = \{R''_1, R''_2, \ldots, R''_{k'-1}, R''_{k'}\}$ *to denote the sufficient set of noisy rules for A, identified in* $\mathcal{R}$.

Definition 7. *We recognise an item* $Item_h \in D_{TR}$ *as a noisy rule-item for class A if* $Item_h \in (\exists R''_j \in \mathcal{R}''(A))$. *We use* $I'''(A) = \{Item'''_1, Item'''_2, \ldots, Item'''_{|I'''(A)|-1}, Item'''_{|I'''(A)|}\}$ *to denote the sufficient set of noisy rule-items for A, identified in* D_{TR}.

Definition 8. *If a CAR* $(R_j \in \mathcal{R}) \notin \mathcal{R}''(A)$, *we recognise* R_j *as a potential significant rule for class A. We use* $\mathcal{R}'''(A) = \{R'''_1, R'''_2, \ldots, R'''_{k-1}, R'''_k\}$ *to denote the sufficient set of potential significant rules for class A, identified in* $\mathcal{R}$ *as* $\mathcal{R} - \mathcal{R}''(A)$, *where integer* $k \geq t$ *(See Definition 4).*

Definition 9. *If a CAR* $(R_j \in \mathcal{R}) \in \mathcal{R}'''(A)$ *satisfies the following inequality,* $\varsigma^A(R_j) > \sum_{h=1}^{|I'''(A)|} \varsigma^A(Item'''_h \in I'''(A))$, *where the contribution of* R_j *to class A is greater than the sum of the contributions of all noisy rule-items for class A, we recognise* R_j *as a significant rule for A. We say there are at most* $\hat{k}$ *significant rules for A in* $\mathcal{R}$, *where integer* $\hat{k}$ *is a user-defined constant* $\leq k$.

3.3 Proposed Rule Ordering Mechanism

In Sect. 2.3.2 five existing rule ordering strategies were presented. Each is separated into both rule weighting and rule re-ordering stages. From the previous section, a list of CARs $\mathcal{R}^{\diamond} \subset \mathcal{R}$ has been generated that only consists of the "best $\hat{k}$" rules for each class $c_i \in C$, identified in $\mathcal{R}$. A rule re-ordering strategy is then required in the process of rule ordering. The rule re-ordering mechanism proposed herein contains three steps as follows:

1. $\mathcal{R}^{\diamond}$ is ordered using the well-established CSA ordering strategy.
2. The original $\mathcal{R}$ is linked at back of $\mathcal{R}^{\diamond}$, as $\mathcal{R}^{\diamond} + \mathcal{R}$.
3. Reassign: $\mathcal{R} \leftarrow (\mathcal{R}^{\diamond} + \mathcal{R})$.

3.4 Deterministic Selectors

We say that a set P hits a set Q on element q, if $P \cap Q = \{q\}$, and a family $\mathcal{F}$ of sets hits a set Q on element q, if $P \cap Q = \{q\}$ for at least one $P \in \mathcal{F}$. De Bonis et al. [19] introduced a definition of a family of subsets of set $[N] \equiv \{0, 1, \cdots, N-2, N-1\}$ which hits each subset of $[N]$ of size at most k on at least m distinct elements, where N, k and m are parameters, $N \geq k \geq m \geq 1$. They proved the existence of such a family of size $O((k^2/(k-m+1)) \log N)$. For convenience of our presentation, we prefer the following slight modification of this definition, obtained by using the parameter $r = k - m$ instead of the parameter m. For integers N and k, and a real number r such that $N \geq k \geq r \geq 0$, a family $\mathcal{F}$ of subsets of $[N]$ is a (N, k, r)-selector, if for any subset $Q \subset [N]$ of size at most k, the number of all elements q of Q such that $\mathcal{F}$ does not hit Q on q is at most r. That is,

$$|\{q \in Q : \forall P \in \mathcal{F}, P \cap Q \neq \{q\}\}| \leq r.$$

In terms of this definition, De Bonis et al. [19] showed the existence of a (N, k, r)-selector of size $T(N, k, r) = O((k^2/(r+1)) \log N)$. In particular, there exists a $(N, k, 0)$-selector of size $O(k^2 \log N)$ such a "strong" selector hits each set $Q \subset [N]$ of size at most k on each of its elements.

3.5 Proposed Randomised Selectors

A randomised k-selector $\mathcal{F}$ is a family of subsets of set $[N] \equiv \{0, 1, \ldots, N-2, N-1\}$ which hits each element q of the subset $Q \subset [N]$ of size at most k with a high probability.

Theorem 1. *There exists a randomised k-selector $\mathcal{F}$ of size $O(k)$ such that $\mathcal{F}$ hits each set $Q \subset [N]$ of size at most k on each of its elements with a constant probability $p \geq 1/8$.*

Proof. Let each element $v \in [N]$ with a uniformed probability $1/k$ to be the part of the element of $\mathcal{F}$. Let H denote the number of different elements in $Q \subset [N]$ that have been hit by $\mathcal{F}$ after repeating the same procedure k times. The probability p that $\mathcal{F}$ hits each set Q of size at most k on each of its elements could be bounded by

$$\begin{aligned} p &= E(H)/E(k) \\ &= \{\textstyle\sum_{i=1}^{k} [(k-i+1)/k] \times k \times (1/k) \times (1-1/k)^{k-1}\}/k \\ &= \textstyle\sum_{i=1}^{k} \{[(k-i+1)/k^2] \times (1-1/k)^{k-1}\} \\ &> \textstyle\sum_{i=1}^{k} \{[(k-i+1)/k^2] \times (1-1/k)^{k}\} \\ &> \textstyle\sum_{i=1}^{k} \{[(k-i+1)/k^2] \times (1/4)\} \qquad (\star) \\ &= (1/4) \times [(1+k) \times (k/2)/k^2] \\ &= (1/4) \times [(1+k)/(2k)] \\ &> (1/4) \times [k/(2k)] \\ &> 1/8 \ , \end{aligned}$$

where inequality $(\star)$ follows from the fact that the sequence $(1-1/k)^k$ is monotonously increasing.

4 Proposed Rule Mining Approach

4.1 The Strategy of the Deterministic Approach

To identify the significant CARs for class A (as $c_i \in C$) in $\mathcal{R}$, we provide a deterministic approach that employs a single application of a "strong" $(2^n, k, 0)$-selector. This approach ensures that every potential significant rule for A will be hit at least once. To apply a family $\mathcal{F}$ of subsets of $[2^n]$ means first to arrange the sets of $\mathcal{F}$ into a sequence $F_1, F_2, \ldots, F_{|\mathcal{F}|}$. In the ith step, only the CARs in $\mathcal{R}$ with labels in F_i will be involved in the procedure SIGNIFICANCE-TEST, while other CARs can be ignored. Thus, we have an $O(k^2 \log 2^n)$-complexity to hit each of the k potential significant rules independently at least once, due to the property of the "strong" selector. If the current test for F_i contains only one potential significant rule R_j and R_j is also a significant rule for A, then we call the function LOG-TEST, which is based on a binary search and finally find R_j. With a "smaller" list of rules (all significant rules and some insignificant rules), we then compute the weighting score of each rule for class A, and finally catch the "best $\hat{k}$" (top-$\hat{k}$ score) rules for A.

4.2 The Strategy of the Randomised Approach

In this section, we use the randomised k-selector to substitute the "strong" selector in Sect. 4.1. This randomised approach ensures that every potential significant rule for class A will be hit at least once with a high probability.

To apply a family $\mathcal{F}$ of subsets of $[2^n]$ means first to arrange the sets of $\mathcal{F}$ into a sequence $F_1, F_2, \cdots, F_{|\mathcal{F}|}$. In the ith step, each element of $[2^n]$ will be contained in F_i with a uniformed probability $1/k$ and only the CARs in $\mathcal{R}$ with labels in F_i will be involved in the procedure SIGNIFICANCE-TEST, while other CARs can be ignored. Thus, we have an $O(k)$-complexity to hit each of the k potential significant rules independently once with a high probability, due to the property of the randomised k-selector (see Theorem 1). With a list of the extracted $O(k)$ rules (all significant rules and some insignificant rules), the weighting score of each rule for A is computed. The "best $\hat{k}$" (top-$\hat{k}$ score) rules are the significant rules for class A identified in $\mathcal{R}$.

4.3 Rule Mining Algorithm

The function (Fig. 2) identifies a rule with a possible large score for class A.

Lemma 1. *If there exists only one potential significant rule R_j in the current test F_i and R_j is also a significant rule, the function* LOG-TEST *will return R_j.*

Proof. According to Definition 9, we know that $\varsigma^A(R_j) > \sum_{h=1}^{|I'''(A)|} \varsigma^A(Item'''_h \in I'''(A))$. Thus, the subset of $\mathcal{R}$ which contains R_j is always chosen for further binary test if R_j is the only potential significant rule including in the current test F_i.

The procedure (Fig. 3) identifies all $\hat{k}$ significant rules for class A in $\mathcal{R}$.

Theorem 2. *The procedure* SIGNIFICANCE-TEST *will catch all $\hat{k}$ significant rules.*

Function LOG-TEST(F_i, $\mathcal{R}$);
input: F_i (the ith element in $\mathcal{F}$) and set $\mathcal{R}$;
output: Rw (a rule in $\mathcal{R}$);
(1)begin
(2) $Rw := null$;
(3) $Temp := F_i$;
(4) while $|Temp| > 1$ do
(5) choose an arbitrary subset $Temp_0$ with half CARs in $Temp$ to test;
(6) if $\sum_{\forall Item_h \in Temp_0} \varsigma^A(Item_h) \geq \sum_{\forall Item_{h'} \in Temp - Temp_0} \varsigma^A(Item_{h'})$
(7) then $Temp \leftarrow Temp_0$;
(8) else $Temp \leftarrow Temp - Temp_0$;
(9) end while
(10) $Rw \leftarrow Temp$;
(11) return (Rw);
(12)end

Fig. 2. The LOG-TEST function

Procedure SIGNIFICANCE-TEST;
input: $\mathcal{F}$ ($(2^n, k, 0)$-selector/randomised k-selector for $[2^n]$), set $\mathcal{R}$, and integer $\hat{k}$;
output: the set $\mathcal{SR}$ (the set of significant rules);
(1)begin
(2) $\mathcal{SR} := \{\oslash\}$;
(3) $\mathcal{PSR} := \{\oslash\}$;
(4) for $i = 1$ to $|\mathcal{F}|$ do
(5) if the label of a CAR R_j in F_i
(6) then R_j will be involved in current test;
(7) else R_j will be ignored in current test;
(8) $\mathcal{PSR} \leftarrow \mathcal{PSR} \cup \{\text{LOG-TEST}(F_i, \mathcal{R})\}$;
(9) end for
(10) $\mathcal{SR} \leftarrow$ catch the top-$\hat{k}$ rules $R_j \in \mathcal{PSR}$ (according to $\varsigma^A(R_j)$);
(11) return ($\mathcal{SR}$);
(12)end

Fig. 3. The SIGNIFICANCE-TEST procedure

Proof. According to the properties of the "selectors" (both deterministic and randomised selectors), we know that the selector $\mathcal{F}$ hits all k potential significant rules at least once. Note that a significant rule is also a potential significant rule. Lemma 1 states that if the current test F_i hits only one significant rule R_j, the function LOG-TEST will figure out R_j, which completes the proof of the theorem.

Lemma 2. *A $(2^n, k, 0)$-selector has size at most $O(k^2 n)$.*

Proof. It directly comes from the property of the selectors.

Theorem 3. *The problem of mining $\hat{k}$ significant rules in $\mathcal{R}$ can be solved in time $O(k^2 n^2)$ in a deterministic manner, where k is the number of potential significant rules in $\mathcal{R}$.*

Proof. The function LOG-TEST takes at most $\log 2^n$ time to find a rule with a possible large score from a subset of $\mathcal{R}$. From Lemma 2, we know that a $(2^n, k, 0)$-selector has the size at most $O(k^2 n)$. Consequently, the total time spent to figure out at most k potential significant rules can be bounded by $O(k^2 n^2)$. The problem of finding the top-$\hat{k}$ significant rules in at most $O(k^2 n)$ rules can be solved in time $O(k^2 n \log(k^2 n) + \hat{k})$, which is $O(k^2 n^2)$ due to $\hat{k} \leq k \leq 2^n$.

Theorem 4. *The problem of mining $\hat{k}$ significant rules in $\mathcal{R}$ can be solved in time $O(kn)$ in a randomised manner, with a constant probability $p \geq 1/8$, where k is the number of potential significant rules in $\mathcal{R}$.*

Proof. The function LOG-TEST takes at most $\log 2^n$ time to find a rule with a possible large score from a subset of $\mathcal{R}$. From Theorem 1, we know that a randomised k-selector has the size $O(k)$ with a high probability to succeed.

Consequently, the total time spent to figure out at most k potential significant rules can be bounded by $O(kn)$ with a constant probability $p \geq 1/8$. The problem of finding the top-$\hat{k}$ significant rules in at most $O(k)$ rules can be solved in time $O(k \log(k) + \hat{k})$, which is $O(kn)$ due to $\hat{k} \leq k \leq 2^n$.

5 Experimental Results

In this section, we aim to evaluate: (1) the proposed rule weighting and rule ordering strategies with respect to the accuracy of classification, where significant rules are mined in both (i) "one-by-one" or deterministic manner and (ii) randomised manner; and (2) the proposed rule mining approach (in its randomised fashion) with respect to the efficiency of computation by comparing it with the "one-by-one" mining approach. All evaluations were obtained using the TFPC CARM algorithm coupled with the "best first" case satisfaction strategy, although any other CARM classifier generator, founded on the "best first" strategy, could equally well be used. Experiments were run on a 1.20 GHz Intel Celeron CPU with 256 Mbyte of RAM running under Windows Command Processor.

The experiments were conducted using a range of datasets taken from the LUCS-KDD discretised/normalised ARM and CARM Data Library [14]. The chosen datasets are originally taken from the UCI Machine Learning Repository [4]. These datasets have been discretised and normalised using the LUCS-KDD DN software,[3] so that data are then presented in a binary format suitable for use with CARM applications. It should be noted that the datasets were rearranged so that occurrences of classes were distributed evenly throughout the datasets. This then allowed the datasets to be divided in half with the first half used as the training-dataset and the second half as the test-dataset. Although a "better" accuracy figure might have been obtained using Ten-Cross Validation [25], it is the relative accuracy that is of interest here and not the absolute accuracy.

The first set of evaluations undertaken used a confidence threshold value of 50% and a support threshold value of 1% (as used in the published evaluations of CMAR [28], CPAR [53], TFPC [15, 18]). The results are presented in Table 1 where 120 classification accuracy values are listed based on 24 chosen datasets. The row labels describe the key characteristics of each dataset: for example, the label *adult.D97.N48842.C2* denotes the "adult" dataset, which includes 48,842 records in two pre-defined classes, with attributes that for the experiments described here have been discretised and normalised into 97 binary categories.

From Table 1 it can be seen that with a 50% confidence threshold and an 1% support threshold the proposed rule weighting and rule ordering mechanisms worked reasonably well. When choosing a value of 1 as the value for $\hat{k}$

[3]The LUCS-KDD DN is available at http://www.csc.liv.ac.uk/~frans/KDD/Software/LUCS-KDD-DN/.

Table 1. Classification accuracy ($\alpha = 50\%$, $\sigma = 1\%$)

Datasets	*CSA*	*One-by-one approach*		*Randomised selector*	
		$\hat{k} = 1$	$\hat{k} = 10$	$\hat{k} = 1$ $k = 5$	$\hat{k} = 10$ $k = 50$
adult.D97.N48842.C2	80.83	83.87	76.88	81.95	81.85
anneal.D73.N898.C6	91.09	89.31	91.09	90.20	91.31
auto.D137.N205.C7	61.76	64.71	59.80	64.71	58.82
breast.D20.N699.C2	89.11	87.68	89.11	90.83	92.55
connect4.D129.N67557.C3	65.83	66.78	65.87	66.34	66.05
cylBands.D124.N540.C2	65.93	69.63	63.70	67.41	67.78
flare.D39.N1389.C9	84.44	84.01	84.29	84.44	84.29
glass.D48.N214.C7	58.88	64.49	52.34	64.49	64.49
heart.D52.N303.C5	58.28	58.28	56.29	60.26	59.60
hepatitis.D56.N155.C2	68.83	68.83	66.23	75.32	72.72
horseColic.D85.N368.C2	72.83	77.72	80.43	80.43	81.52
ionosphere.D157.N351.C2	85.14	84.00	90.29	88.57	93.14
iris.D19.N150.C3	97.33	97.33	97.33	97.33	97.33
led7.D24.N3200.C10	68.38	62.94	68.38	68.89	69.94
letRecog.D106.N20000.C26	30.29	29.41	31.19	29.36	30.92
mushroom.D90.N8124.C2	99.21	98.45	98.82	99.21	98.45
nursery.D32.N12960.C5	80.35	76.85	76.17	80.20	81.11
pageBlocks.D46.N5473.C5	90.97	91.74	90.97	91.74	90.97
pima.D38.N768.C2	73.18	73.18	73.18	73.44	73.44
soybean-large.D118.N683.C19	85.92	81.23	86.51	84.46	84.75
ticTacToe.D29.N958.C2	71.61	68.48	72.03	71.19	73.28
waveform.D101.N5000.C3	61.60	58.92	55.96	59.52	57.20
wine.D68.N178.C3	53.93	83.15	71.91	83.15	85.39
zoo.D42.N101.C7	76.00	86.00	78.00	90.00	86.00
Average	73.82	75.29	74.03	76.81	76.79

(only the most significantly CAR for each class is mined) and applying the "one-by-one" rule mining approach, the average accuracy of classification throughout the 24 datasets is 75.29%. When substituting the value of 1 by a value of 10 (the best ten significant CARs for each class are identified), the average accuracy, using the "one-by-one" rule mining approach, is 74.03%. Note that the average accuracies are higher than the average accuracy of classification obtained by the well-established CSA ordering approach, which is 73.82%. Furthermore when dealing with the randomised selector based rule mining approach, and choosing a value of 1 as the value for $\hat{k}$ and a value of 5 as the value for k (only the most significantly CAR for each class is mined, based on the existence of five potential significant CARs for each class in $\mathcal{R}$), the average accuracy throughout the 24 datasets can be obtained as 76.81%. Note that in the randomised experiment process, we always run several tests (i.e., 8–10 tests) for each dataset, and catch the best result. When substituting the value of 1 by a value of 10, and the value of 5 by a value of 50 (the best ten significant

CARs for each class are mined, based on the existence of 50 potential significant CARs for each class in $\mathcal{R}$), the average accuracy was found as 76.79%.

The second set of evaluations undertaken used a confidence threshold value of 50%, a set of decreasing support threshold values from 1 to 0.03%, and the letter recognition dataset. The "large" letter recognition dataset (*letRecog.D106.N20000.C26*), comprises 20,000 records and 26 pre-defined classes. For the experiment the dataset has been discretised and normalised into 106 binary categories. From the experiment it can be seen that a relationship exists between: the selected value of support threshold (σ or *min.support*), the number of generated CARs (—$\mathcal{R}$—), the accuracy of classification (*Accy*), and the time in seconds spent on computation ($Time$). Clearly, $\downarrow \sigma \Rightarrow \uparrow |\mathcal{R}| \Rightarrow (\uparrow Accy \wedge \uparrow Time)$.

Table 2 demonstrate that with a 50% confidence threshold and a value of 1 as the value for $\hat{k}$ (only the most significantly CAR for each class is mined in $|\mathcal{R}|$), the proposed rule mining approach (its randomised fashion) performs well with respect to both accuracy of classification and efficiency of computation. When applying the "one-by-one" rule mining approach, as σ decreasing from 1 to 0.03%, $|\mathcal{R}|$ (before mining the "best $\hat{k}$" rules) is increased from 149 to 6,341; and $|\mathcal{R}|$ (after mining the "best $\hat{k}$" rules and re-ordering all rules) is increased from 167 to 6,367. Consequently accuracy has been increased from 29.41 to 48.22%, and $Time$ (the time spent on mining the $\hat{k}$ significant rules) has been increased from 0.08 to 12.339 s. In comparison when applying the proposed randomised rule mining approach with a value of 50 as the value for k (there exist 50 potential significant rules for each class in $|\mathcal{R}|$), as σ decreasing from 1 to 0.03%, $|\mathcal{R}|$ (before mining the "best $\hat{k}$" rules) is increased from 149 to 6,341; and $|\mathcal{R}|$ (after mining the "best $\hat{k}$" rules and

Table 2. Computational efficiency and classification accuracy ($\alpha = 50\%$)

Dataset	*One-by-one approach*				*Randomised selector*			
letRecog	$\hat{k} = 1$				$\hat{k} = 1$, $k = 50$			
D106. N20000. C26	Rule number (before)	Rule number (after)	Time (s)	Accuracy (%)	Rule number (before)	Rule number (after)	Time (s)	Accuracy (%)
1	149	167	0.080	29.41	149	166	0.160	29.60
0.75	194	212	0.110	29.94	194	211	0.160	29.92
0.50	391	415	0.200	35.67	391	411	0.251	35.78
0.25	1118	1143	1.052	40.36	1118	1139	0.641	41.26
0.10	2992	3018	4.186	44.95	2992	3016	0.722	45.18
0.09	3258	3284	4.617	45.21	3258	3282	1.913	45.42
0.08	3630	3656	6.330	45.88	3630	3655	2.183	45.43
0.07	3630	3656	6.360	45.88	3630	3656	2.163	46.02
0.06	4366	4392	5.669	46.70	4366	4391	2.754	46.45
0.05	4897	4923	7.461	47.28	4897	4922	3.235	47.65
0.04	5516	5542	9.745	47.67	5516	5542	3.526	47.53
0.03	6341	6367	12.339	48.22	6341	6365	4.296	48.79

re-ordering all rules) is increased from 166 to 6,365. Consequently accuracy has been increased from 29.60 to 48.79%, and $Time$ (the time spent on mining the $\hat{k}$ significant rules) has been increased from 0.16 to 4.296 s.

Figures 4 and 5 demonstrate (respectively) that there is no significant difference between accuracies of classification obtained by the "one-by-one"

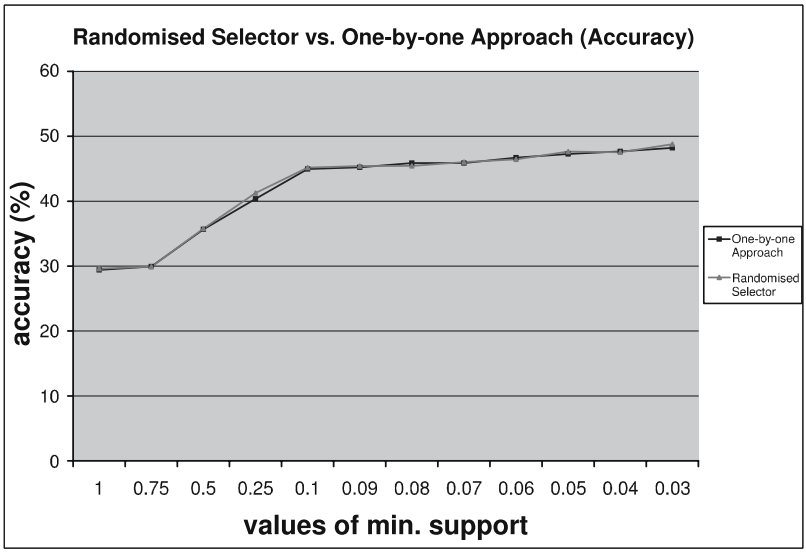


Fig. 4. Randomised selector vs. one-by-one approach (accuracy)

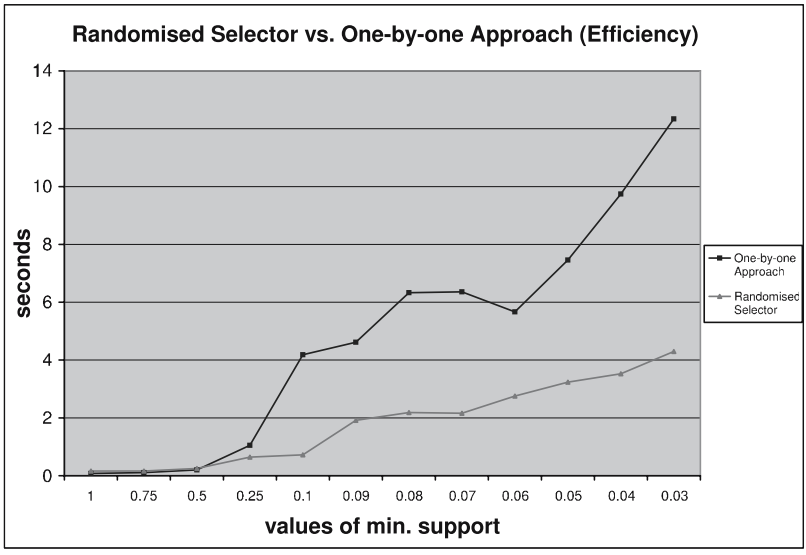


Fig. 5. Randomised selector vs. one-by-one approach (efficiency)

rule mining approach and the randomised selector based rule mining approach, whereas a significant difference in times spent on mining significant rules can be seen.

6 Conclusion

This chapter is concerned with an investigation of CARM. A overview of existing CARM algorithms was provided in Sect. 2 where five existing rule weighting schemes used in CARM algorithms were reviewed. A rule weighting scheme was proposed in Sect. 3 that was used to distinguish the significant CARs from the insignificant ones. Consequently a rule ordering strategy was proposed, based on the "best first" case satisfaction approach, which can be applied when classifying "unseen" data. The concept of selectors [19] was summarised in Sect. 3 together with some discussion of the randomised selectors. A novel rule mining approach was presented in Sect. 4 based on the concepts of selectors (both deterministic and randomised). In theory, the proposed rule mining approach identifies significant CARs in time $O(k^2n^2)$ in its deterministic fashion, and $O(kn)$ in its randomised fashion. This mining approach avoids computing CAR scores and finding significant CARs on a "one-by-one" basis, which will require an exponential time $O(2^n)$. In Sect. 5, two sets of evaluations were presented that evidence:

1. The proposed rule weighting and rule ordering approach's perform well with respect to the accuracy of classification.
2. The proposed randomised rule mining approach is comparable to the "one-by-one" rule mining approach in significant CAR identification with respect to both the accuracy of classification and the efficiency of computation.

From the experimental results, it can be seen that the accuracy of classification obtained by the proposed randomised rule mining approach can be better than the accuracy obtained by the "one-by-one" approach. Further research is suggested to identify improved rule weighting scheme to find more significant rules in $\mathcal{R}$. Other obvious direction for further research include: finding other rule ordering mechanisms that give a better classification accuracy; investigating other techniques to replace the proposed deterministic and/or randomised selectors to give a better performance; etc.

Acknowledgement

The authors would like to thank Professor Leszek Gąsieniec and Professor Paul Leng of the Department of Computer Science at the University of Liverpool for their support with respect to the work described here.

References

1. Agrawal R, Imielinski T, Swami A (1993) Mining association rules between sets of items in large databases. In: Buneman P, Jajodia S (eds): Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data (SIGMOD-93, ACM, New York, NY), Washington, DC, United States, May 1993. (pages 207–216)
2. Agrawal R, Srikant R (1994) Fast algorithm for mining association rules. In: Bocca JB, Jarke M, Zaniolo C (eds): Proceedings of the 20th International Conference on Very Large Data Bases (VLDB-94, Morgan Kaufmann, San Francisco, CA), Santiago de Chile, Chile, September 1994. (ISBN 1-55860-153-8, pages 487–499)
3. Ali K, Manganaris S, Srikant R (1997) Partial classification using association rules. In: Heckerman D, Mannila H, Pregibon D, Uthurusamy R (eds): Proceedings of the Third International Conference on Knowledge Discovery and Data Mining (KDD-97, AAAI, Menlo Park, CA), Newport Beach, California, United States, August 1997. (ISBN 1-57735-027-8, pages 115–118)
4. Blake CL, Merz CJ (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
5. Bong CH, Narayanan K (2004) An empirical study of feature selection for text categorization based on term weightage. In: Proceedings of the 2004 IEEE/WIC/ACM International Conference on Web Intelligence (WI-04, IEEE Computer Society), Beijing, China, September 2004. (ISBN 0-7695-2100-2, pages 599–602)
6. Boser BE, Guyon IM, Vapnik VN (1992) A training algorithm for optimal margin classifiers. In: Haussler D (ed): Proceedings of the fifth ACM Annual Workshop on Computational Learning Theory (COLT-92, ACM, New York, NY), Pittsburgh, Pennsylvania, United States, July 1992. (ISBN 0-89791-497-X, pages 144–152)
7. Brin S, Motwani R, Ullman JD, Tsur S (1997) Dynamic itemset counting and implication rules for market basket data. In: Peckham J (ed): Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data (SIGMOD-97, ACM, New York, NY), Tucson, Arizona, United States, May 1997. (SIGMOD Record 26(2), pages 255–264)
8. Burdick D, Calimlim M, Gehrke J (2001) MAFIA: A maximal frequent itemset algorithm for transactional databases. In: Proceedings of the 17th International Conference on Data Engineering (ICDE-01, IEEE Computer Society), Heidelberg, Germany, April 2001. (ISBN 0-7695-1001-9, pages 443–452)
9. Cheung W, Zaïane OR (2003) Incremental mining of frequent patterns without candidate generation or support constrain. In: Seventh International Database Engineering and Applications Symposium (IDEAS-03, IEEE Computer Society), Hong Kong, China, July 2003. (ISBN 0-7695-1981-4, pages 111–116)
10. Clark P, Boswell R (1991) Rule induction with CN2: Some recent improvements. In: Kodratoff Y (ed): Machine Learning – Proceedings of the Fifth European Working Session on Learning (EWSL-91, Springer, Berlin Heidelberg New York), Porto, Portugal, March 1991. (LNAI 482, ISBN 3-540-53816-X, pages 151–163)

11. Coenen F, Goulbourne G, Leng P (2001) Computing association rules using partial totals. In: Raedt LD, Siebes A (eds): Principles of Data Mining and Knowledge Discovery – Proceedings of the 5th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD-01, Springer, Berlin Heidelberg New York), Freiburg, Germany, September 2001. (LNAI 2168, ISBN 3-540-42534-9, pages 54–66)
12. Coenen F, Leng P (2001) Optimising association rule algorithms using itemset ordering. In: Bramer M, Coenen F, Preece A (eds): Research and Development in Intelligent Systems XVIII – Proceedings of the Twenty-first SGES International Conference on Knowledge Based Systems and Applied Artificial Intelligence (ES-01, Springer, Berlin Heidelberg New York), Cambridge, United Kingdom, December 2001. (ISBN 1852335351, pages 53–66)
13. Coenen F, Leng P (2002) Finding association rules with some very frequent attributes. In: Elomaa T, Mannila H, Toivonen H (eds): Principles of Data Mining and Knowledge Discovery – Proceedings of the 6th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD-02, Springer, Berlin Heidelberg New York), Helsinki, Finland, August 2002. (LNAI 2431, ISBN 3-540-44037-2, pages 99–111)
14. Coenen F (2003) The LUCS-KDD discretised/normalised ARM and CARM data library. http://www.csc.liv.ac.uk/ frans/KDD/Software/LUCS-KDD-DN/, Department of Computer Science, The University of Liverpool, UK
15. Coenen F, Leng P (2004) An evaluation of approaches to classification rule selection. In: Proceedings of the Fourth IEEE International Conference on Data Mining (ICDM-04, IEEE Computer Society), Brighton, UK, November 2004. (ISBN 0-7695-2142-8, pages 359–362)
16. Coenen F, Leng P, Ahmed S (2004) Data structure for association rule mining: T-trees and p-trees. IEEE Transactions on Knowledge and Data Engineering, Volume 16(6):774–778
17. Coenen F, Leng P, Goulbourne G (2004) Tree structures for mining association rules. Journal of Data Mining and Knowledge Discovery, Volume 8(1):25–51
18. Coenen F, Leng P, Zhang L (2005) Threshold tuning for improved classification association rule mining. In: Ho TB, Cheung D, Liu H (eds): Advances in Knowledge Discovery and Data Mining – Proceedings of the Ninth Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD-05, Springer, Berlin Heidelberg New York), Hanoi, Vietnam, May 2005. (LNAI 3518, ISBN 3-540-26076-5, pages 216–225)
19. De Bonis A, Gąsieniec L, Vaccaro U (2003) Generalized framework for selectors with applications in optimal group testing. In: Baeten JCM, Lenstra JK, Parrow J, Woeginger GJ (eds): Proceedings of the Thirtieth International Colloquium on Automata, Languages and Programming (ICALP-03, Springer, Berlin Heidelberg New York), Eindhoven, The Netherlands, June 30–July 4, 2003. (LNAI 2719, ISBN 3-540-40493-7, pages 81–96)
20. Domingos P, Pazzani M (1997) On the optimality of the simple bayesian classifier under zero-one loss. Machine Learning, 29(2/3):103–130.
21. Dong G, Li J (1999) Efficient mining of emerging patterns: Discovering trends and differences. In: Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-99, ACM, New York, NY), San Diego, CA, United States, August 1999. (pages 43–52)
22. Dong G, Zhang X, Wong L, Li J (1999) CAEP: Classification by aggregating emerging patterns. In: Arikawa S, Furukawa K (eds): Discovery Science –

Proceedings of the Second International Conference Discovery Science (DS-99, Springer, Berlin Heidelberg New York), Tokyo, Japan, December 1999. (LNAI 1721, ISBN 3-540-66713-X, pages 30–42)
23. Dunham MH (2002) Data mining: Introductory and advanced topics. Prentice-Hall, August 2002. (ISBN 0-13-088892-3)
24. El-Hajj M, Zaïane OR (2003) Inverted matrix: efficient discovery of frequent items in large datasets in the context of interactive mining. In: Getoor L, Senator TE, Domingos P, Faloutsos C (eds): Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-03, ACM, New York, NY), Washington, DC, United States, August 2003. (ISBN 1-58113-737-0, pages 109–118)
25. Freitas AA (2002) Data mining and knowledge discovery with evolutionary algorithms, Springer, Berlin Heidelberg New York, Germany, 2002. (ISBN 3-540-43331-7)
26. Gouda K, Zaki MJ (2001) Efficiently mining maximal frequent itemsets. In: Cercone N, Lin TY, Wu X (eds): Proceedings of the 2001 IEEE International Conference on Data Mining (ICDM-01, IEEE Computer Society), San Jose, CA, United Stated, 29 November–2 December 2001. (ISBN 0-7695-1119-8, pages 163–170)
27. Han J, Pei J, Yin Y (2000) Mining frequent patterns without candidate generation. In: Chen W, Naughton JF, Bernstein PA (eds): Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data (SIGMOD-00, ACM, New York, NY), Dallas, TX, United States, May 2000. (ISBN 1-58113-218-2, pages 1–12)
28. Han J, Kamber M (2001) Data mining: Concepts and techniques. Morgan Kaufmann, San Francisco, CA, United States, 2001. (ISBN 1-55860-489-8)
29. Han J, Kamber M (2006) Data mining: Concepts and techniques (Second Edition). Morgan Kaufmann, San Francisco, CA, United States, March 2006. (ISBN 1-55860-901-6)
30. Hand D, Mannila H, Smyth R (2001) Principles of data mining. MIT, Cambridge, MA, United States, August 2001. (ISBN 0-262-08290-X)
31. Hidber C (1999) Online association rule mining. In: Delis A, Faloutsos C, Ghandeharizadeh S (eds): Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data (SIGMOD-99, ACM, New York, NY), Philadelphia, Pennsylvania, United States, June 1999. (ISBN 1-58113-084-8, pages 145–156)
32. Holsheimer M, Kersten ML, Mannila H, Toivonen H (1995) A perspective on databases and data mining. In: Fayyad UM, Uthurusamy R (eds): Proceedings of the First International Conference on Knowledge Discovery and Data Mining (KDD-95, AAAI Press, Menlo Park, CA), Montreal, Canada, August 1995. (ISBN 0-929280-82-2, pages 150–155)
33. Houtsma M, Swami A (1995) Set-oriented mining of association rules in relational databases. In: Yu PS, Chen AL (eds): Proceedings of the Eleventh International Conference on Data Engineering (ICDE-95, IEEE Computer Society), Taipei, Taiwan, March 1995. (ISBN 0-8186-6910-1, pages 25–33)
34. James M (1985) Classification algorithms. Wiley, New York, NY, United States, 1985. (ISBN 0-471-84799-2)
35. Lavrač N, Flach P, Zupan B (1999) Rule evaluation measures: A unifying view. In: Dzeroski S, Flach PA (eds): Proceedings of the Ninth International Work-

shop on Inductive Logic Programming (ILP-99, Springer, Berlin Heidelberg), Bled, Slovenia, June 1999. (LNAI 1634, ISBN 3-540-66109-3, pages 174–185)
36. Li W, Han J, Pei J (2001) CMAR: Accurate and efficient classification based on multiple class-association rules. In: Cercone N, Lin TY, Wu X (eds): Proceedings of the 2001 IEEE International Conference on Data Mining (ICDM-01, IEEE Computer Society), San Jose, CA, United States, 29 November–2 December 2001. (ISBN 0-7695-1119-8, pages 369–376)
37. Lin D-I, Kedem ZM (1998) Pincer search: A new algorithm for discovering the maximum frequent set. In: Schek H-J, Saltor F, Ramos I, Alonso G (eds): Advances in Database Technology – Proceedings of the Sixth International Conference on Extending Database Technology (EDBT-98, Springer, Berlin Heidelberg New York), Valencia, Spain, March 1998. (LNAI 1377, ISBN 3-540-64264-1, pages 105–119)
38. Liu B, Hsu W, Ma Y (1998) Integrating classification and association rule mining. In: Agrawal R, Stolorz PE, Piatetsky-Shapiro G (eds): Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD-98, AAAI, Menlo Park, CA), New York City, New York, United States, August 1998. (ISBN 1-57735-070-7, pages 80–86)
39. Liu J, Pan Y, Wang K, Han J (2002) Mining frequent item sets by opportunistic projection. In: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-02, ACM, New York, NY), Edmonton, Alberta, Canada, July 2002. (ISBN 1-58113-567-X, pages 229–238)
40. Mannila H, Toivonen H, Verkamo AI (1994) Efficient algorithms for discovering association rules. In: Fayyad UM, Uthurusamy R (eds): Knowledge Discovery in Databases: Papers from the 1994 AAAI Workshop (KDD-94, AAAI, Menlo Park, CA), Seattle, Washington, United States, July 1994. (Technical Report WS-94-03, ISBN 0-929280-73-3, pages 181–192)
41. Michalski RS (1980) Pattern recognition as rule-guided inductive inference. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1980. (pages 774–778)
42. Mirkin B, Mirkin BG (2005) Clustering for data mining: A data recovery approach. Chapman & Hall/CRC Press, April 2005. (ISBN 1584885343)
43. Park JS, Chen M-S, Yu PS (1995) An effective hash based algorithm for mining association rules. In: Carey MJ, Schneider DA (eds): Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data (SIGMOD-95, ACM, New York, NY), San Jose, CA, United States, May 1995. (SIGMOD Record 24(2), pages 175–186)
44. Pei J, Han J, Mao R (2000) CLOSET: An efficient algorithm for mining frequent closed itemsets. In: Gunopulos D, Rastogi R (eds): 2000 ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery (SIGMOD-DMKD-01), Dallas, TX, United Stated, May 2000. (pages 21–30)
45. Quinlan JR (1993) C4.5: Programs for machine learning. Morgan Kaufmann Publishers, San Francisco, CA, United States, 1993. (ISBN 1-55860-238-0)
46. Quinlan JR, Cameron-Jones RM (1993) FOIL: A midterm report. In: Brazdil R (ed): Machine Learning – Proceedings of the 1993 European Conference on Machine Learning (ECML-93, Springer, Berlin Heidelberg New York), Vienna, Austria, April 1993. (LNAI 667, ISBN 3-540-56602-3, pages 3–20)

47. Roberto J, Bayardo Jr (1998) Efficiently mining long patterns from databases. In: Hass LM, Tiwary A (eds): Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data (SIGMOD-98, ACM, New York, NY), Seattle, Washington, United States, June 1998. (ISBN 0-89791-995-5, pages 85–93)
48. Rymon R (1992) Search through systematic set enumeration. In: Nebel B, Rich C, Swartout WR (eds): Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning (KR-92, Morgan Kaufmann, San Francisco, CA), Cambridge, MA, United States, October 1992. (ISBN 1-55860-262-3, pages 539–550)
49. Savasere A, Omiecinski E, Navathe S (1995) An efficient algorithm for mining association rules in large databases. In: Proceedings of the twenty-first International Conference on Very Large Data Bases (VLDB-95, Morgan Kaufmann, San Francisco, CA), Zurich, Switzerland, September 1995. (ISBN 1-55860-379-4, pages 432–444)
50. Toivonen H (1996) Sampling large databases for association rules. In: Vijayaraman TM, Buchmann AP, Mohan C, Sarda NL (eds): Proceedings of the twenty-second International Conference on Very Large Data Bases (VLDB-96, Morgan Kaufmann, San Francisco, CA), Mumbai (Bombay), India, September 1996. (ISBN 1-55860-382-4, pages 134–145)
51. Wang J, Han J, Pei J (2003) CLOSET+: Searching for the best strategies for mining frequent closed itemsets. In: In: Getoor L, Senator TE, Domingos P, Faloutsos C (eds): Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-03, ACM, New York, NY), Washington, DC, United States, August 2003. (ISBN 1-58113-737-0, pages 236–245)
52. Wang W, Yang J (2005) Mining sequential patterns from large data sets. Springer, Berlin Heidelberg New York, April 2005. (ISBN 0-387-24246-5)
53. Yin X, Han J (2003) CPAR: Classification based on predictive association rules. In: Barbará D, Kamath C (eds): Proceedings of the Third SIAM International Conference on Data Mining (SDM-03, SIAM, Philadelphia, PA), San Francisco, CA, United States, May 2003. (ISBN 0-89871-545-8, pages 331–335)
54. Zaki MJ, Parthasarathy S, Ogihara M, Li W (1997) New algorithms for fast discovery of association rules. In: Heckerman D, Mannila H, Pregibon D (eds): Proceedings of the Third International Conference on Knowledge Discovery and Data Mining (KDD-97, AAAI, Menlo Park, CA), Beach, CA, United States, August 1997. (ISBN 1-57735-027-8, pages 283–286)
55. Zaki MJ, Hsiao C-J (2002) CHARM: An efficient algorithm for closed itemset mining. In: Grossman RL, Han J, Kumar V, Mannila H, Motwani R (eds): Proceedings of the Second SIAM International Conference on Data Mining (SDM-02, SIAM, Philadelphia, PA), Arlington, VA, United States, April 2002. (ISBN 0-89871-517-2, Part IX No. 1)

Data Preprocessing and Data Mining as Generalization

Anita Wasilewska[1] and Ernestina Menasalvas[2]

[1] Department of Computer Science, State University of New York, Stony Brook, NY, USA
anita@cs.sunysb.edu

[2] Departamento de Lenguajes y Sistemas Informaticos Facultad de Informatica, U.P.M, Madrid, Spain
ernes@fi.upm.es

Summary. We present here an abstract model in which data preprocessing and data mining proper stages of the Data Mining process are are described as two different types of generalization. In the model the data mining and data preprocessing algorithms are defined as certain generalization operators. We use our framework to show that only three Data Mining operators: classification, clustering, and association operator are needed to express all Data Mining algorithms for classification, clustering, and association, respectively. We also are able to show formally that the generalization that occurs in the preprocessing stage is different from the generalization inherent to the data mining proper stage.

1 Introduction

We build models in order to be able to address formally intuitively expressed notions, or answer intuitively formulated questions. We say for example, that Data Mining generalizes the data by transforming them into a more general information. But what in fact is a generalization? When a transformation of data is, and when is not a generalization? How one kind of generalization differs from the other? The model presented here addresses and answers, even if partially these questions.

There are many data mining algorithms and thousands of implementations. A natural questions arise: why very different algorithms are all called, for example, the classification algorithms? What do they have in common? How do they differ from other algorithms?

We hence build our model in such a way as to be able to define characteristics common to one type of algorithms, and not to the other types.

We present here three models: generalization model (Definition 1) and its particular cases, data mining model (Definition 12), and preprocessing model (Definition 26).

A. Wasilewska and E. Menasalvas: *Data Preprocessing and Data Mining as Generalization*, Studies in Computational Intelligence (SCI) **118**, 469–484 (2008)
www.springerlink.com

In data mining model each class of data mining algorithms is represented by an operator. Theses operators are also generalization operators of the generalization model, i.e. they capture formally the intuitive notion of information generalization. Moreover, we show that (Theorem 4) all operators belonging to one category are distinctive with other categories.

The generalization model presented here is an extension of the model presented in [16], preliminary version of the data mining model and preprocessing models was presented in [13, 14], respectively.

We usually view Data Mining results and present them to the user in their descriptive form as it is the most natural form of communication. But the Data Mining process is deeply semantical in its nature. The algorithms process records (semantics) finding similarities which are then often presented in a descriptive i.e. syntactic form. Our model is a semantical one. Nevertheless it supports the extraction of syntactical information, at any level of generalization. We address the semantics-syntax duality in the case of classification model in [**?**], and we develop a more general framework in [15].

2 Generalization Model

Data Mining, as it is commonly said, is a process of generalization. In order to model this process we have to define what does it mean that one stage of data mining process is more general then the other. The main idea behind our definition of Generalization Model is that generalization consists in putting objects (records) in sets of objects.

From syntactical point of view generalization consists also of *building descriptions* of these sets of objects, with some extra parameters, if needed. We build descriptions in terms of attribute and values of the attribute pairs in particular, and they are expressions of some formal language in general.

Our Generalization Model is semantic in nature, but, as we mentioned before, it also incorporates the syntactic information to be extracted, when (and if) needed.

The model presented here generalizes many ideas developed during years of investigations. First they appeared as a part of development of Rough Sets Theory (to include only few recent publications) [3–5, 10, 11, 17–19]; then in building Rough Sets inspired foundations of information generalization in [1, 6, 7], and foundations of Data Mining in [?, 8, 13–16].

Definition 1. A Generalization Model *is a system*

$$\mathcal{GM} = (U,\ \mathcal{K},\ \mathcal{G},\ \preceq)$$

where

$U \neq \emptyset$ *is the* universe*;*
$\mathcal{K} \neq \emptyset$ *is the set of* knowledge generalization states*;*

$\preceq \subseteq \mathcal{K} \times \mathcal{K}$ *is a transitive relation, called* a generalization relation*;*
$\mathcal{G} \neq \emptyset$ *is the set of partial functions*

$$G: \mathcal{K} \longrightarrow \mathcal{K}.$$

Elements of $\mathcal{G}$ *are called* generalizations operators .

We define all components of the model in the following Sects. 2.1–2.4.

2.1 Knowledge Generalization System

The *knowledge generalization system* is an extension of the notion of an information system. The information system was introduced in [9] as a database model. The information system represents the relational table with key attribute acting as object attribute and is defined as follows.

Definition 2. *Pawlak's* Information System *is a system* $I = (U, A, V_A, f)$, *where* $U \neq \emptyset$ *is called a set of* objects, $A \neq \emptyset$, $V_A \neq \emptyset$ *are called the set of* attributes *and* values *of attributes, respectively,* f *is called an* information *function and* $f : U \times A \longrightarrow V_A$

Any Data Mining process starts with a certain initial set of data. The model of such a process depends on representation of this data, i.e. it starts with an initial information system

$$I_0 = (U_0, A_0, V_{A_0}, f_0)$$

and we adopt the set U_0 as the universe of the model, i.e.

$$\mathcal{GM} = (U_0, \ \mathcal{K}, \ \mathcal{G}, \ \preceq).$$

In preprocessing stage of data mining process we might perform the following standard operations:

1. Eliminate some attributes, apply concept hierarchy. etc.. obtaining as result the information system I with the set of attributes $A \subset A_0$
2. Perform some operations on values of attributes: normalization, clustering, etc ..., obtaining some set V_A of values of attributes that is similar, or equivalent to V_0. We denote it by

$$V_A \sim V_0$$

Given an attribute value $v_a \in V_A$ and a corresponding attribute $v_a^0 \in V_0$ (for example v_a being a normalized form of v_a^0 or v_a being a more general form as defined by concept hierarchy of v_a^0) we denote this correspondence by

$$v_a \sim v_a^0.$$

We call any information system I obtained by any of the above operation *a subsystem of* I_0. We put it formally in the following definition.

Definition 3. *Given two information systems* $I_0 = (U_0, A_0, V_{A_0}, f_0)$ *and* $I = (U, A, V_A, f)$*, we say that* I is a subsystem of I_0 *and denote it as*

$$I \subseteq I_0$$

if and only if the following conditions are satisfied

(i) $|U| = |U_0|$
(ii) $A \subseteq A_0$, $V_A \sim V_0$
(iii) The information functions f *and* f_0 *are such that*

$$\forall x \in U \forall a \in A(f(x,a) = v_a \ \Leftrightarrow \exists v_a^0 \in V_0(f_0(x,a) = v_a^0 \cap v_a^0 \sim v_a))$$

In the data analysis, preprocessing and data mining we start the process with the input data. We assume here that they are represented in a format of information system table. We hence define the lowest level of information generalization as the relational table. The meaning of the intermediate and final results are considered to be of a higher level of generalization. We represent those levels of generalization by a sets of objects of the given (data mining) universe U, as in [1, 7].

This approach follows the granular view of the data mining and is formalized within a notion of knowledge generalization system, defined as follows.

Definition 4. A knowledge generalization system *based on the information system* $I = (U, A, V_A, f)$ *is a system*

$$K_I = (\mathcal{P}(U), A, E, V_A, V_E, g)$$

where

E *is a finite set of* knowledge attributes *(k-attributes) such that* $A \cap E = \emptyset$.
V_E *is a finite set of* values of k- attributes.
g *is a partial function called* knowledge information function*(k-function)*

$$g : \mathcal{P}(U) \times (A \cup E) \longrightarrow (V_A \cup V_E)$$

such that
(i) $g \mid (\bigcup_{x \in U} \{x\} \times A) = f$
(ii) $\forall_{S \in \mathcal{P}(U)} \forall_{a \in A}((S,a) \in dom(g) \ \Rightarrow \ g(S,a) \in V_A)$
(iii) $\forall_{S \in \mathcal{P}(U)} \forall_{e \in E}((S,e) \in dom(g) \ \Rightarrow \ g(S,e) \in V_E)$

Any set $S \in \mathcal{P}(U)$ i.e. $S \subseteq U$ is often called a granule or a group of objects.

Definition 5. *The set*

$$Gr_K = \{S \in \mathcal{P}(U) : \ \exists b \in (E \cup A)((S,b) \in dom(g))\}$$

is called a granule universe *of* K_I.

Observe that g is a total function on Gr_K.

Definition 6. *We call the system* $K = (Gr_K, E, V_E, g)$ *a* granule knowledge generalization *system.*

The condition (i) of Definition 4 says that when $E = \emptyset$ the k-function g is total on the set $\{\{x\} : x \in U\} \times A$ and

$$\forall x \in U \forall a \in A(g(\{x\}, a) = f(x, a)).$$

Definition 7. *The set*

$$\mathcal{P}^{obj}(U) = \{\{x\} : x \in U\}$$

is called an object universe. The knowledge generalization system

$$K^{obj} = (\mathcal{P}^{obj}(U), A, \emptyset, V_A, \emptyset, g) = (\mathcal{P}^{obj}(U), A, V_A, g)$$

is called an object knowledge generalization *system.*

Theorem 1. *For any information system* $I = (U, A, V_A, f)$*, the object knowledge generalization system* $K_I^{obj} = (\mathcal{P}^{obj}(U), A, V_A, g)$ *is isomorphic with* I*. We denote it by*

$$I \simeq K_I^{obj}.$$

The function $F : U \longrightarrow \mathcal{P}^{obj}(U), \quad F(x) = \{x\}$ establishes (by condition (i) of Definition 4) the required isomorphism of K_I^{obj} and I.

2.2 Universe and Knowledge Generalization States

Any Data Mining process starts with a certain initial set of data. The model of such a process depends on representation of this data and we represent it in a form information system table.

We assume hence that the data mining process we model starts with an initial information system

$$I_0 = (U_0, A_0, V_{A_0}, f_0)$$

and we adopt the *universe* U_0 *as the universe of the model*, i.e.

$$\mathcal{GM} = (U_0,\ \mathcal{K},\ \mathcal{G},\ \preceq).$$

Data Mining process consists of transformations the initial I_0 into an initial knowledge generalizations systems K_0 that in turn is being transformed into some knowledge generalizations systems K_I, all of them based on some subsystems I of the input system I_0, what we denote by $I \subseteq I_0$. The formal definition of the notion of subsystem I of the input system I_0 is presented in [13]. These transformations of the initial input data (system I_0) in practice are defined by different Data Preprocessing and Data Mining algorithms, and in our model by appropriate generalization operators. We hence adopt the following definition.

Definition 8. *Let I_0 the initial input data. We adopt the set*

$$\mathcal{K} = \{K_I : \quad I \subseteq I_0\}$$

as the set of all knowledge generalization states

2.3 Generalization Relation

A generalization process starts with the input data I_0 i.e. with the initial knowledge generalization system K_{I_0} with its universe $U = \{\{x\} : x \in U_0\}$, called (Definition 7) an object knowledge generalizations system. It then produces systems which are more general, with universes $S \subseteq \mathcal{P}(U_0)$ with more then one element i.e. such that $|S| > 1$. We adopt hence the following definition of generalization relation.

Definition 9. *Given set $\mathcal{K}$ of knowledge states based on the input data I_0 and $K, K' \in \mathcal{K}$ i.e.*

$$K = (\mathcal{P}(U_0), A, E, V_A, V_E, g),$$

$$K' = (\mathcal{P}(U_0), A', E', V_{A'}, V_{E'}, g').$$

Let $G_K, G_{K'}$ be granule universes (Definition 5) of K, K' respectively. We define a generalization relation

$$\preceq \subseteq \mathcal{K} \times \mathcal{K}$$

as follows:
$K \preceq K'$ if and only if the following conditions are satisfied.

(i) $|G_{K'}| \leq |G_K|$
(ii) $A' \subseteq A$

If $K \preceq K'$ we say that the system K' is more or equally general as K.

Directly from the definitions we get the following theorem.

Theorem 2. *Let $\preceq$ be a relation defined in the Definition 9. The following conditions hold.*

(i) The relation $\preceq$ is transitive, and hence is a generalization relation in a sense of the definition 1 of the Generalization Model.
(ii) $\preceq$ is reflexive but is not antisymmetric, as systems K and K' such that $K \preceq K'$ may have different sets of knowledge attributes and knowledge functions.

Definition 10. *Let $\preceq \subseteq \mathcal{K} \times \mathcal{K}$ be relation defined in the Definition 9. A relation $\prec_{dm} \subseteq \preceq$ such that it satisfies additional conditions:*

(iii) $|G_{K'}| \neq |G_K|$
(iv) $\exists S \in G_{K'}(|S| > 1)$

is called a data mining generalization relation.

Directly from the above definition we get that the following theorem holds.

Theorem 3. *Let $\prec_{dm}$ be a relation defined in the Definition 10. The following conditions hold.*

(i) The relation $\prec_{dm}$ is transitive, and hence is a generalization relation in a sense of the definition 1 of the Generalization Model
(ii) $\prec_{dm}$ is not reflexive and is not antisymmetric

2.4 Generalization Operators

Generalization operators by Definition 1, operate on the knowledge states, preserving their generality, as defined by the generalization relation. I.e. a partial function $G: \mathcal{K} \longrightarrow \mathcal{K}$ is called a generalization operator if for any $K, K' \in domainG$

$$G(K) = K' \quad if\ and\ only\ if \quad K \preceq K'.$$

Generalization operators are designed to describe the action of different data mining algorithms.

3 Data Mining Model

Data Mining process consists of two phases: preprocessing and data mining proper. The Data Mining phase with its generalization operators is discussed in detail in Sect. 4 and in its preliminary version in [14]. The preprocessing operators and preprocessing phase as expressed within our Generalization Model and are presented in Sect. 5 and in [13].

Data Mining Model defined below is a special case of the Generalization Model, with generalization relation being data mining relation as defined in Definition 10 and in which the generalization operators are defined as follows.

Definition 11. *An operator $G \in \mathcal{G}$ is called a* data mining generalization operator *if and only if for any $K, K' \in domainG$*

$$G(K) = K' \quad if\ and\ only\ if \quad K \prec_{dm} K'$$

for some data mining generalization relation $\prec_{dm}$ (Definition 10)

Definition 12. *A* Data Mining Model *is a system*

$$\mathbf{DM} = (U,\ \mathcal{K},\ \mathcal{G}_{dm},\ \prec_{dm}),$$

where the set $\mathcal{G}_{dm}$ is the set of data mining generalization operators.

The above Definition 11 defines a class of data mining operators. They are discussed in detail in the next section.

4 Data Mining Operators

The main idea behind the concept of generalization operator is to capture not only the fact that data mining techniques generalize the data but also to categorize existing methods. We want to do it in as exlusive/inclusive sense as possible. We don't include in our analysis purely statistical methods like regression, etc... This gives us only three data mining generalization operators to consider: classification, clustering, and association.

In the following sections we define the appropriate sets of operators: $\mathcal{G}_{clf}, \mathcal{G}_{clr}$ and $\mathcal{G}_{assoc}$ (Definitions 17, 20, 21) and prove the following theorem.

Theorem 4 (Main Theorem). *Let $\mathcal{G}_{clf}, \mathcal{G}_{clr}$ and $\mathcal{G}_{assoc}$ be the sets of all classification, clustering, and association operators, respectively. The following conditions hold.*

(1) $\mathcal{G}_{clf} \neq \mathcal{G}_{clr} \neq \mathcal{G}_{assoc}$
(2) $\mathcal{G}_{assoc} \cap \mathcal{G}_{clf} = \emptyset$
(3) $\mathcal{G}_{assoc} \cap \mathcal{G}_{clr} = \emptyset$

4.1 Classification Operator

In the classification process we are given a data set (set of records) with a special attribute C, called a class attribute. The values $c_1, c_2, \ldots c_n$ of the class attribute C are called class labels. The classification process is both semantical (grouping objects in sets that would fit the classes) and syntactical (finding the descriptions of those sets in order to use them for testing and future classification). In fact all data mining techniques share the same characteristics of semantical-syntactical duality.

The formal definitions of classification data and classification operators are as follows.

Definition 13. *Any information system $I = (U, A \cup \{C\}, V_A \cup V_{\{C\}}, f)$ with a distinguished class attribute C and with the class attribute values $V_{\{C\}} = \{c_1, c_2, \ldots c_m\}$,$m \geq 2$ is called a classification information system, or shortly,* a classification system *if and only if the sets*

$$C_n = \{x \in U_0 : \quad f(x, C) = c_n\}$$

form a partition of U_0.

The classification information system is called in the Rough Set community and literature [3,10,11,18,19] a decision information system with the *decision attribute* C. We assume here, as it is the case in usual classification problems, that we have only one classification attribute. It is possible, as the Rough Set community does, to consider the decision information systems with any non empty set $C \subset A$ of decision attributes.

Definition 14. *Let* $I_0 = (U_0, A, V_A \cup V_{\{C\}}, f)$ *be the initial database with the class attribute* C*. The sets*

$$C_{n,0} = \{x \in U_0 : \quad f(x, C) = c_n\}$$

are called the initial classification classes.

Definition 15. *The corresponding set* $\mathcal{K}$ *of knowledge systems based on any subsystem* I *of the initial classification information system* I_0*, as defined in the Definition 8, is called the set of* classification knowledge systems *if and only if for any* $K \in \mathcal{K}$ *the following additional condition holds.*

$$\forall S \in \mathcal{P}(U)(\exists a \in A((S, a) \in dom(g)) \Rightarrow (S, C) \in dom(g)).$$

We denote,

$$\mathcal{K}^{clf}$$

the set of all classification knowledge systems based on a classification system I_0.

Definition 16. *For any* $K \in \mathcal{K}^{clf}$ *we the sets*

$$C_{n,K} = \{X \in \mathcal{P}(U_0) : \quad g(X, C) = c_n\}$$

are called group classes *of* K*.*

Let $\mathbf{DM} = (U_0,\ \mathcal{K}^{clf},\ \mathcal{G}_{dm},\ \prec_{dm})$ be a Data Mining Model based on a classification system I_0.

Definition 17. *A generalization operator* $G \in \mathcal{G}_{dm}$ *is called a* classification operator *if and only if* G *is a partial function*

$$G :\ \mathcal{K}^{clf} \quad \longrightarrow \quad \mathcal{K}^{clf},$$

such that for any $K' \in domG$*, any* $K \in \mathcal{K}^{clf}$ *such that* $K = G(K')$ *the following* classification condition *holds*

$$\forall X(X \in C_{n,K} \ \Rightarrow \ X \subseteq_K C_{n,0}),$$

where the sets $C_{n,0}, C_{n,K}$ *are the sets from Definitions 14 and 16, respectively and* $\subseteq_K$ *is an* approximate set inclusion *defined in terms of k-attributes of* K*.*

We denote the set of classification operators based on the initial system I_0 by

$$\mathcal{G}_{clf}.$$

Observe that our definition of the classification operators gives us freedom of choosing the level of generality (granularity) with which we want to describe our data mining technique, in this case the classification process.

4.2 Clustering Operator

In intuitive sense the term *clustering* refers to the process of grouping physical or abstract objets into classes of similar objects. It is also called *unsupervised* learning or unsupervised classification. We say that *a cluster* is a collection of data objects that are similar to one another within the collection and are dissimilar to the objects in other clusters [2]. Clustering analysis constructs hence meaningful partitioning of large sets of objects into smaller components. One of the basic property we consider while building clusters is the measure of similarity and dissimilarity. These measures must be present in our definition of the knowledge generalization system applied to clustering analysis. We define hence a notion of clustering knowledge system as follows.

Definition 18. *A knowledge generalization system* $K \in \mathcal{K}$,

$$K = (\mathcal{P}(U), A, E, V_A, V_E, g)$$

is called a clustering knowledge system *if and only if* $E \neq \emptyset$ *and there are two knowledge attributes* $s, ds \in E$ *such that for any* $S \in Gr_K$ *(definition 5)*

$g(S, sm)$ *is a measure of similarity of objects in* S
$g(S, ds)$ *is a measure of dissimilarity of objects in* S *with other* $X \in Gr_K$

We put

$$\mathcal{K}^{clr} = \{K \in \mathcal{K} : K \text{ is a clustering system}\}.$$

Definition 19. *We denote* $\mathcal{K}^{obj} \subset \mathcal{K}$ *the set of all* object knowledge generalization *systems.*

Definition 20. *An operator* $G \in \mathcal{G}_{dm}$ *is called a* clustering operator *if and only if* G *is a partial function*

$$G : \mathcal{K}^{obj} \longrightarrow \mathcal{K}^{clr}$$

and for any $K' \in domG$ *such that* $G(K') = K$ *the granule universe* Gr_K *(Definition 5) of* K *is a* partition *of* U *satisfying the following condition:*

$$\forall X, Y \in Gr_K (g(X, sm) = g(Y, sm) \cap g(X, ds) = g(Y, ds)).$$

Elements of the granule universe Gr_K of K are called *clusters* defined (generated) by the operator G and we denote the set of all clustering operators by $\mathcal{G}_{clr}$.

It is possible to define within our framework clustering methods that return not always disjoint clusters (with for example some measures of overlapping).

We can also allow the cluster knowledge system be a classification system. It allows the cluster operator to return not only clusters it has generated, their descriptions (with similarity measures) but their names, if needed.

Finally, our framework allows us to incorporate as well the notion of classification by clustering (for example k-nearest neighbor algorithm) by changing the domain of the cluster operator to allow the use of training examples.

4.3 Association Operator

The association analysis is yet another important subject and will be treated in a separate paper. We can define a special knowledge generalization system system AK, called an *association system*. All frequent $k-associations$ are represented in it (with the support count), as well as all information needed to compute association rules that follow from them.

We put

$$\mathcal{K}^{assoc} = \{K \in \mathcal{K} : K \text{ is an association system}\}.$$

Definition 21. *An operator $G \in \mathcal{G}_{dm}$ is called an* association operator *if and only if G is a partial function that maps the set of all* object *association systems $\mathcal{K}^{objassoc}$ into $\mathcal{K}^{assoc}$, i.e.*

$$G: \mathcal{K}^{objassoc} \longrightarrow \mathcal{K}^{assoc}$$

and some specific association conditions hold.

We denote the set of all association operators by $\mathcal{G}_{assoc}$.

5 Data Preprocessing Generalization

The preprocessing of data is the initial and often crucial step of the data mining process. We show here that the Generalization Model (Definition 1) presented here is strong enough to express not only the data mining stage of data mining process but the preprocessing stage as well. Moreover, we show that preprocessing stage and data mining stage generalize data in a different way and that in fact, the generalization proper, i.e. defined by the strong generalization relation (Definition 23) occurs only at the data mining stage. The preprocessing operations are expressed in the model as a weak generalization defined by a weak generalization relation (Definition 22). We show that they lead to the strong information generalization in the next, data mining proper stage and improve the quality (granularity) of the generalization process.

5.1 Strong and Weak Generalization Models

It is natural that when building a model of the data mining process one has to include data preprocessing methods and algorithms, i.e. one has to model within it preprocessing stage as well as the data mining proper stage. In order to achieve this task we introduce the preprocessing generalization relation (Definition 25) and examine its relationship with the data mining generalization relation (Definition 10). We show (Theorem 5)that they are particular cases of the information generalization relation as defined in our generalization model (Definition 1). We also prove (Theorem 8) that the preprocessing

relation is a special case of the weak information generalization relation and it is disjoint with our data mining generalization relation. This means that within the framework of our general model we are able to distinguish (as we should have) the preprocessing generalization from the generalization that occurs in the data mining proper stage.

Definition 22. A Weak Generalization Model *is the generalization model (Definition 1) in which the generalization relation is* reflexive. *We denote the generalization relation of the weak model by* $\preceq$ *and call it a* weak generalization relation.

Definition 23. A Strong Generalization Model *is the generalization model (Definition 1) in which the information generalization relation is not reflexive. We denote the generalization relation of the strong model by* $\prec$ *and call it a* strong generalization relation.

The relationship between weak and strong generalization relations, generalization relation (Definition 9) and data mining generalization relation (Definition 10) and hence between Data Mining Model, Generalization Model, and Strong and Weak Generalization Models is express by the following theorem.

Theorem 5. *Let* $\preceq$ *and* $\prec_{dm}$ *be the generalization relation as defined by the Definition 9 and the data mining generalization relation (Definition 10), respectively. The following properties hold.*

(i) $\prec_{dm} \subset \preceq$
(ii) $\prec$ *is a weak information generalization of the Definition 22*
(iii) $\preceq_{dm}$ *is a strong information generalization of the Definition 23*

The condition (i) is true by definition, condition (ii) follows from Definition 22 and Theorem 2, and the condition (iii) follows from Definition 23 and Theorem 3.

Given initial information system $I_0 = (U_0, A_0, V_{A_0}, f_0)$, the object knowledge generalization system (Definition 7)

$$K^{obj}_{I_0} = (\mathcal{P}^{obj}(U_0), A, V_A, g)$$

is isomorphic with I_0 i.e. $K^{obj}_{I_0} \simeq I_0$ by theorem 1 and is also called the *initial knowledge generalization* system.

Data preprocessing process in the preprocessing stage consists of transformations the initial knowledge generalization system $K^{obj}_{I_0} \simeq I_0$ into a certain $K^{obj}_{I} \simeq I$ for $I \subseteq I_0$. Any data mining stage of transformation starts, for unification purposes with corresponding initial knowledge generalization systems $K^{obj}_{I} \simeq I$ obtained by the preprocessing process.

Let $\mathcal{K}$ be the set of all *knowledge generalization states of* $\mathcal{GM}$ as defined in the Definition 8. We define its special subset $\mathcal{K}^{prep}$ corresponding to the preprocessing stage as follows.

Definition 24. *The set* $\mathcal{K}^{prep} \subseteq \mathcal{K}$ *such that*

$$\mathcal{K}^{prep} = \{K_I^{obj} : \ K_I^{obj} \simeq I \ \cap \ I \subseteq I_0 \cap |U_I| = |U_{I_0}|\}$$

is called a set of preprocessing knowledge states*, and any* $K \in \mathcal{K}^{prep}$ *is called a* preprocessing knowledge systems *of* $\mathcal{GM}$.

Observe that the condition $I \subseteq I_0 \cap |U_I| = |U_{I_0}|$ means that we model only the proper preprocessing stages, excluding data cleaning and records elimination.

Definition 25. *Let* $\mathcal{K}^{prep} \subseteq \mathcal{K}$ *be a the set of preprocessing states (Definition 24) and* $\preceq$ *be the generalization relation of the Definition 9. A relation* $\preceq_{prep} \subseteq \preceq$ *defined as follows:*

$$\preceq_{prep} = \preceq \cap (\mathcal{K}^{prep} \times \mathcal{K}^{prep})$$

is called a preprocessing generalization relation.

Observe that by Theorem 1 and Definition 7, any $K_I \in \mathcal{K}^{prep}$ is isomorphic with I. Let now consider any $K, K' \in \mathcal{K}^{prep}$. By Definition 24, $K = K_I^{obj} \simeq I$, for $I \subseteq I_0$, $K' = K_{I'}^{obj} \simeq I'$, for $I' \subseteq I_0$ and $|U_I| = |U_{I'}| = |U_{I_0}|$. By Definition 5 and condition **(i)** of Definition 4 we get that the cardinality of the granule universes of K and K' are equal. We hence proved the following theorem.

Theorem 6. *For any* $K, K' \in \mathcal{K}^{prep}$,

$$|Gr_K| = |Gr_{K'}| = |U_{I_0}|.$$

The Theorem 6 combined with the Definition 9, gives us the following theorem.

Theorem 7. *For any* $K, K' \in \mathcal{K}^{prep}$,

$$K \preceq_{prep} K' \ if \ and \ only \ if \ |Gr_K| = |Gr_{K'}|.$$

The above theorem says that within our framework the systems K, K' such that $K \preceq_{prep} K'$ are, in fact, equally general,

So why do we call some preprocessing operations a "generalization"? There are two reasons. One is that traditionally some preprocessing operations have been always called by this name. For example we usually state that we "generalize" attributes by clustering, by introducing attributes hierarchy, by aggregation, etc. as stated [2].

> ...."Data transformation (preprocessing stage) can involve the following ..... **Generalization** of the data , where low-level or "primitive" (raw) data are replaced by higher-level concepts through the use of concept hierarchies. For example, categorical attributes ..... can be generalized to higher level concepts. Similarly, values for numeric attributes, like ... may be mapped to higher level concepts." ....

The second, more important reason follows directly from Theorems 6 and 7 and is expressed in the following.

Theorem 8. *The preprocessing generalization relation* $\preceq_{prep}$*is a weak generalization relation and is not a data mining generalization relation (Definition 10) i.e.*

$$\preceq_{prep} \cap \prec_{dm} \ = \ \emptyset.$$

This theorem states that preprocessing operations are a weak generalization, disjoint with a strong, data mining generalization. We nevertheless perform the preprocessing weak generalization them because it leads to the strong generalization in the next, data mining proper stage. We need the preprocessing stage to improve the quality, i.e. the granularity of the data mining proper generalization. This is the reason why we routinely call the preprocessing transformations a generalization.

The Theorem 8 also says that within the framework of our generalization model we are able to distinguish (as we should have) the generalization that occurs during the preprocessing stage of the data mining process from the generalization of the data mining proper stage.

6 Data Preprocessing Model

It is natural that when building a model of the data mining process one has to include data preprocessing methods and algorithms, i.e. one has to model within it preprocessing stage as well as the data mining proper stage. In order to achieve this task we introduced the notion of weak information generalization relation as a component of our weak generalization model (Definition 22). We have then introduced the preprocessing and the data mining generalization relations (Definitions 10 and 25, respectively) and proved (Theorem 8) that the preprocessing relation is a special case of the weak information generalization relation and it is disjoint with our data mining generalization relation.

Consequently we define here a semantic model of data preprocessing, as a particular cases of our generalization model (Definition 1) as follows.

Definition 26. *When we adopt the preprocessing generalization relation* $\preceq_{prep}$ *(Definition 25) as the information generalization relation of the generalization model* $\mathcal{GM}$ *(Definition 1) we call the model thus obtained a* Preprocessing Model *and denote it* **PM***, i.e.*

$$\mathbf{PM} = (U, \ \mathcal{K}^{prep}, \ \mathcal{G}_{prep}, \ \prec_{prep})$$

where

$\mathcal{K}^{prep}$ *is the set of preprocessing knowledge states (Definition 24),*
$\mathcal{G}_{prep} \subseteq \mathcal{G}$ *called a set of preprocessing generalization operators defined on* $\mathcal{K}^{prep}$*. We assume that* $\mathcal{G}_{prep} \cap \mathcal{G}_{dm} = \emptyset$*, where* $\mathcal{G}_{dm}$ *is the set of data mining operators (Definition 12).*

The data mining proper stage is determined by the data mining generalization relation $\prec_{dm}$ (Definition 10).

We express the whole data mining process within our generalization model as follows.

Definition 27. Data Mining Process Model *is a system*

$$\mathbf{DMP} = (U,\ \mathcal{K},\ \mathcal{G}_p,\ \preceq_{process}),$$

where

(i) $\preceq_{process} = \preceq_{prep} \cup \prec_{dm}$
(ii) $\mathcal{G}_{process} = \mathcal{G}_{prep} \cup \mathcal{G}_{dm}$

The set $\mathcal{G}_{dm}$ of data mining is defined in detail in Sect. 4, the detailed definition of the set $\mathcal{G}_{prep}$ of data preprocessing operators will be a subject of a separate paper.

References

1. M. Hadjimichael, A. Wasilewska. *A Hierarchical Model for Information Generalization.* Proceedings of the Fourth Joint Conference on Information Sciences, Rough Sets, Data Mining and Granual Computing (RSDMGrC'98), NC, USA, vol. II, pp. 306–309
2. J. Han, M. Kamber. *Data Mining: Concepts and Techniques.* Morgan Kauffman, Los Altos, CA, 2000
3. M. Inuiguchi, T. Tanino. *Classification Versus Approximation Oriented Generalization of Rough Sets.* Bulletin of International Rough Set Society, 7:1/2, 2003
4. J. Komorowski. *Modelling Biological Phenomena with Rough Sets.* Proceedings of Third International Conference RSCTC'02, Malvern, PA, October 2002, p. 13. Springer Lecture Notes in Artificial Intelligence
5. T.Y. Lin. *Database Mining on Derived Attributes.* Proceedings of Third International Conference RSCTC'02, Malvern, PA, October 2002, pp. 14–32. Springer Lecture Notes in Artificial Intelligence
6. J.F. Martinez, E. Menasalvas, A. Wasilewska, C. Fernández, M. Hadjimichael. *Extension of Relational Management System with Data Mining Capabilities.* Proceedings of Third International Conference RSCTC'02, Malvern, PA, October 2002, pp. 421–428. Springer Lecture Notes in Artificial Intelligence
7. E. Menasalvas, A. Wasilewska, C. Fernández. *The Lattice Structure of the KDD Process: Mathematical Expression of the Model and its Operators.* International Journal of Information Systems and Fundamenta Informaticae, 48–62, special issues, 2001
8. E. Menasalvas, A. Wasilewska, C. Fernández, J.F. Martinez. *Data Mining – A Semantical Model.* Proceedings of 2002 World Congress on Computational Intelligence, Honolulu, Hawai, May 11–17, 2002, pp. 435–441
9. Z. Pawlak, *Information Systems – Theoretical Foundations.* Information Systems, 6:205–218, 1981

10. Z. Pawlak, *Rough Sets – Theoretical Aspects Reasoning About Data*. Kluwer, Dordecht, 1991
11. A. Skowron, *Data Filtration: A Rough Set Approach*. Proceedings de Rough Sets, Fuzzy Sets and Knowledge Discovery. 1993, pp. 108–118
12. A. Wasilewska, E.M. Ruiz, M.C. Fernández-Baizan. *Modelization of Rough Set Functions in the KDD Frame*. First International Conference on Rough Sets and Current Trends in Computing (RSCTC'98), Warsaw, Poland, June 22–26 1998
13. A. Wasilewska, E. Menasalvas. *Data Preprocessing and Data Mining as Generalization Process*. Proceedings of ICDM'04, the Fourth IEEE International Conference on Data Mining, Brighton, UK, November 1–4, 2004, pp. 25–29
14. A. Wasilewska, E. Menasalvas. *Data Mining Operators*. Proceedings of ICDM'04, the Fourth IEEE International Conference on Data Mining, Brighton, UK, November 1–4, 2004, pp. 43–52
15. A. Wasilewska, E. Menasalvas, C. Scharff. *Uniform Model for Data Mining*. Proceedings of FDM05 (Foundations of Data Mining), in ICDM2005, Fifth IEEE International Conference on Data Mining, Austin, Texas, November 27–29, 2005, pp. 19–27
16. A. Wasilewska, E.M. Ruiz. *Data Mining as Generalization: A Formal Model*. Foundation and Advances in Data Mining, T.Y. Lin, W. Chu, editors. Springer Lecture Notes in Artificial Intelligence, 2005
17. W. Ziarko, X. Fei. *VPRSM Approach to WEB Searching*. Proceedings of Third International RSCTC'02 Conference, Malvern, PA, October 2002, pp. 514–522. Springer Lecture Notes in Artificial Intelligence
18. W. Ziarko. *Variable Precision Rough Set Model*. Journal of Computer and Systen Sciences, 46(1):39–59, 1993
19. J.T. Yao, Y.Y. Yao. *Induction of Classification Rules by Granular Computing*. Proceedings of Third International RSCTC'02 Conference, Malvern, PA, October 2002, pp. 331–338. Springer Lecture Notes in Artificial Intelligence

Capturing Concepts and Detecting Concept-Drift from Potential Unbounded, Ever-Evolving and High-Dimensional Data Streams

Ying Xie[1], Ajay Ravichandran[2], Hisham Haddad[1], and Katukuri Jayasimha[2]

[1] The CSIS Department, Kennesaw State University, Kennesaw, GA 30144, USA
yxie2@kennesaw.edu, hhaddad@kennesaw.edu
[2] CACS, University of Louisiana at Lafayette, Lafayette, LA 70504, USA
ajay@louisiana.edu,jrk@louisiana.edu

Summary. Envisioning the needs of a new platform that supports comprehensive knowledge discovery and retrieval from potential unbounded, multi-dimensional and ever evolving data streams, this paper proposes a novel integrated architecture that encapsulates a suit of interrelated data structures and algorithms which support (1) real-time capturing and compressing dynamics of stream data into space-efficient synopses, (2) online mining and visualizing both dynamics and historical snapshots of multiple types of patterns from stored synopses. Preliminary experimental results are provided to illustrate the effectiveness of the provided architecture in capturing concepts and detecting concept-drift from streaming data.

1 Introduction

Unbounded, ever-evolving and high-dimensional data streams, which are generated by various sources such as scientific experiments, real-time production systems, e-transactions, sensor networks and online equipments, add further layers of complexity to the already challenging drown in data, starving for knowledge problem. While multiple stream mining algorithms have been proposed, each of which is dedicated to extract particular predefined type of patterns, there is an urgent need to investigate new integrated architectures that supports comprehensive knowledge discovery from huge volume of transient data. The principle goal of this work is to develop a novel integrated architecture that encapsulates a suit of interrelated data structures and algorithms which support (1) real-time capturing and compressing dynamics of stream data into space-efficient synopses, (2) online mining and visualizing both dynamics and historical snapshots of multiple types of patterns from stored synopses. This is definitely a challenging task given that such

Y. Xie et al.: *Capturing Concepts and Detecting Concept-Drift from Potential Unbounded, Ever-Evolving and High-Dimensional Data Streams*, Studies in Computational Intelligence (SCI) **118**, 485–499 (2008)
www.springerlink.com

comprehensive knowledge discovery architecture should be built solely upon one-pass scan of potentially unbounded, ever evolving and high dimensional data stream. In order to tackle this challenge, in this paper, we (1) proposes a unique cube structure, where the time dimension is dynamically segmented by concept drifts into concept cycles, the sub-cube at each concept cycle only consists of cells that are active at that cycle, and the facts of each cell are compressed time series of statistics summaries of the events occurred in that cell over a concept cycle; (2) explores an efficient in-memory data structure to organize the sub-cube for the current concept cycle, capture and compress statistics of events occurred in each active cell, and automatically detect concept drifts; (3) investigates effectively compressed and efficiently indexed storage structures to warehouse historical concepts; (4) develops online algorithms to query and visualize the current or a historical concept from different perspectives solely based on synopses stored in the disk. The rest of this paper is organized as follows. A brief review of related works is presented in Sect. 2. In Sect. 3, we define concept and concept drift, and describe the data structure used to compress and store concepts. Then, in Sect. 4, we present the data structures and algorithms for real-time capturing concept and detecting concept drift. In Sect. 5, we discuses the learning algorithm for pre-determining a couple of parameters. The online algorithms to query and visualize the current or a historical concept from different perspectives are presented in Sect. 6. In Sect. 7, preliminary experimental results are presented. Finally, we will reach our conclusion in Sect. 8.

2 Related Work

A large number of works focus on building adaptive classification models to deal with concept drifts occurred in data stream [1–8]. Two major types of methodologies are adopted to serve this purpose. The first type is to build incremental classification model that continuously incorporating new data and fading the effects of old examples at certain rates [2,4,7,8]. The other type of methods is to take advantage of classifier ensembles [3,5,6], where the weights of classifiers are continuously modified in order to adapt to concept drift. Obviously, this group of works process data streams that are mixed with labeled and unlabeled data. But in many cases, class labels are not available in data streams. Another problem is that these works can not detect evolving pattern itself. The unsupervised learning (clustering algorithms) are thought to be a challenging problem to data streams [9]. Some one-pass versions of clustering algorithms have been proposed to tackle the scalability problem brought by unbounded data streams [10–12]. Arguing that one-pass clustering algorithms over an entire data stream suffer from the heavy effects of the outdated historical data, Aggarwal et al. proposed an interesting method to explore clustering patterns over different time windows [9]. This work stores statistical snapshots of the data stream at different level of granularity depending upon

the recency. Based on stored snapshots, the user can specify the time horizon for which the clustering patterns can be obtained by running an offline clustering component. This work can effectively get the clustering pattern formed in the time horizon specified by the user, which is useful in applications where the user know exactly the time period he or she is interested in. However, this work is still unable to automatically detect the concept drift in a stream and discover the evolving patterns of the stream. There are few works dedicated to mining changes from a data stream. Kifer et al. provides a statistical definition of changes and proposes a change-detection algorithm by comparing the data in some "reference window" to the data in the current window [13]. While Aggarwal proposed a way to diagnose changes in evolving data streams based on velocity density estimation [14]. However, this type of works does not describe the pattern presented by the data stream in a stable period or at a historical snapshot. The work closest to ours is the multidimensional stream analysis [15]. This work applies a cube structure to organize the streaming data. The granularity of the time dimension of the cube can be second, minute, quarter, hour, and so on. Then each base cell of the cube stores a compressed time series of data points arriving in the corresponding time period. This work uses linear regression to compress the time series and demonstrates how to aggregate the linear regression function along each dimension. Although the proposed architecture can facilitate OLAP queries over stream data, it is not a suitable platform to automatically discover patterns and pattern drifts across the section boundaries manually imposed on each dimension. Unlike this work which differentiates numerical facts from other dimensions, the approach proposed in this paper views both numerical and categorical attributes as dimensions. The segmentations on the numerical attributes are pre-learned from sample data extracted from the stream. More importantly, rather than manually separating the time dimension into even length units, our work automatically divide the time dimension into a series of concept cycles by detected concept drift, so that each concept cycle reflects a relatively stable concept. A base cell of our cube structure maintains the compressed time series of statistics of the data points falling into that cell, instead of the time series of numerical attribute values. In the following sections we will show that our cube structure can facilitate not only the generation of different types of patterns at any snapshot or within any concept cycle, but also detecting the concept drift across multiple concept cycles, as well as the micro-shift within a concept cycle. Furthermore, one can easily impose manually determined hierarchical levels onto the natural segmentations of each dimension in order to facilitate OLAP queries.

3 Concept and Concept Drift

Let's assume we capture the n-dimensional data within only one concept cycle marked as c1 from a data stream and plot the data in an n-dimensional space.

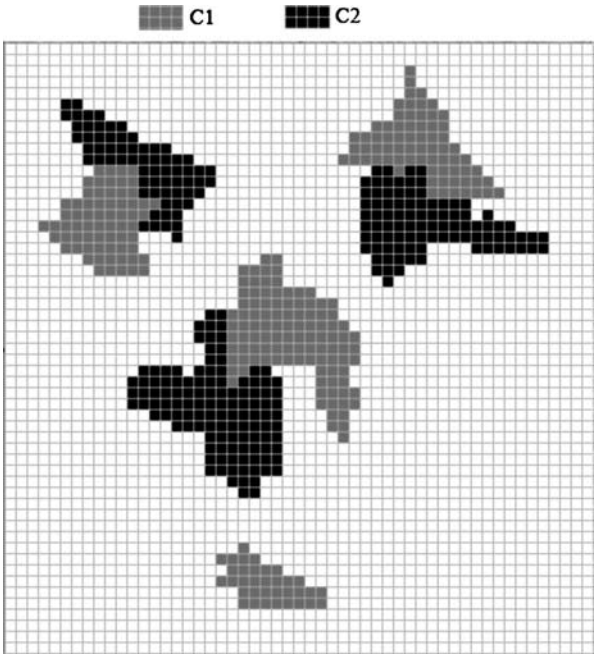


Fig. 1. Descriptions of concept C1 and C2

All the dense areas in the n-dimension space can be viewed as the approximate description for c1. When the data captured is streaming, it surely takes time for the description of the concept to be identifiable. During this time period, some dense areas may appear earlier, some may appear later. Let's assume we also capture the data within the concept cycle immediately following the concept cycle $c1$ and mark it as $c2$. Again we can approximately describe $c2$ by using its dense areas (see Fig. 1). Now, the question that needs to be addressed is how to identify the boundary between $c1$ and $c2$? In other words, when the data is streaming, how can we know whether it is still in the forming period of cycle $c1$ or it has already entered into cycle $c2$? In order to address this problem, we first view the dense area in a concept cycle as composed of a group of adjacent dense cells. The size of the dense cell is learned from static training data extracted from the corresponding data stream, which is discussed in Sect. 4. When the data is streaming, data points keep falling into the corresponding cells, making some of the cells hold the number of points exceeding certain threshold θ_n (θ_n is learned from static training data as well) and become dense. We stamp each time point when a cell becomes dense. In this way, we maintain a time series of timestamps that mark the occurrences of new dense cells. Whenever a new timestamp is added to this time series, a linear regression is conducted to predict the next timestamp t_{pred}. When the real next timestamp tnext is marked, we calculate the difference between tnext and tpred. If $(t_{next} - t_{pred}) > \theta_t$, we view the new dense cell formed at

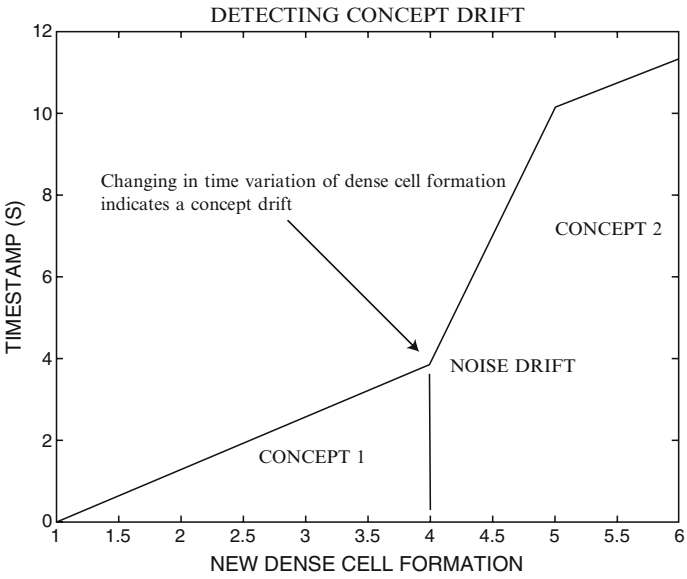


Fig. 2. Descriptions of concept C1 and C2

tnext as the first dense cell of the next concept cycle. Concept drift of a date stream is illustrated in Fig. 2.

After solving the concept drift problem, the next question that needs to be addressed is how to design a space-efficient synopsis for each concept cycle, such that the user can easily query different patterns from this concept cycle, or in other words, view this concept from multiple perspectives. We define *active cells* of a concept cycle c_i to be those cells that become dense within the cycle. For the current concept cycle, each active cell is associated with three time series. The first time series of a cell records timestamps at which the number of points falling into that cell is dividable by the dense threshold θ_n (i.e., time stamps at which the number of points falling into that cell becomes $1 \times \theta_n, 2 \times \theta_n, 3 \times \theta_n$, and so on), while the second and third time series of that cell record the cell's mean and deviation at each timestamp captured by the first time series. For example, the three time series maintained for active cell $cell_j$ of the current concept cycle are shown as follows:

- Time series of timestamp: $Cell - t_j : \{t1, t2, t3, t4, t5, \cdots\}$.
- Time series of means: $Cell - m_j : \{m1, m2, m3, m4, m5, \cdots\}$.
- Time series of means: $Cell - d_j : \{d1, d2, d3, d4, d5, \cdots\}$.

From the first time series of timestamps, we can easily calculate the number of data points falling into this cell at each time point. For example, at t5, the number of points in $cell_j$, denoted as n_{t5}, is $5\theta_n$. Therefore, the time series of timestamps for $cell_j$ can be converted into the time series of number of points:

$cell_j : \{(t_1, \theta_n), (t_2, 2\theta_n), (t_3, 3\theta_n), (t_4, 4\theta_n), (t_5, 5\theta_n), \cdots\}$

Then, by applying linear regression, the number of points in $cell_j$ at time t can be described as a linear function $n_t = a_n t + b_n$. In the same way, the means of $cell_j$ at time t can be described as $m_t = a_m t + b_m$, and the deviation of $cell_j$ at time t can be described as $d_t = a_d t + b_d$. Therefore, $cell_j$ can be approximately represented by a triple $((a_n, b_n), (a_m, b_m), (a_d, b_d))$. In this way, the space size for storing each concept cycle is approximately the number of the active cells in that cycle.

4 Capturing Concept and Detecting Concept Drift

The previous section deals with the definitions of concept and concept-drift, as well as the description and storage of a concept. This section presents the data structure called time-stamped partricia trie (TSP trie) for real-time capturing concepts and detecting concept-drift. Like a general patricia trie, the TSP trie stores all its data at the external nodes (or leaf node) and keeps one integer, the bit-index, in each internal nodes as an indication of which bit of a query is to be used for branching. This avoids empty sub-trie and guarantees that every internal node will have non-null descendants. Each external node of the TSP trie is a string that represents the coordinates of a dense cell. The root of the TSP trie maintains a time series of timestamps. Whenever there is a dense cell occur (or re-occur), that time point will be stamped at the root of the trie. When the data begin streaming, each data point $a_i = (a_{i1}, a_{i2}, \cdots, a_{ij}, \cdots, a_{in})$ is mapped to the corresponding cell in the n-dimensional space by a hashing function. For each non empty cell, a triple (n, m, d), where n is the number of points in the cell, m is the mean, and d is the standard deviation, is organized in a hash table. When a cell becomes dense at the first time, it is inserted into the TSP trie. The root of the TSP trie stamps the time point when the new dense cell is inserted. The new node sets up three time series $Cell{-}t_j, Cell{-}m_j, Cell{-}d_j$ as described in the previous section. Simultaneously, the corresponding entry (triple (n, m, d)) in the hash table is flushed to empty. When a cell becomes dense again, the root of the TSP trie stamps the time point again, and the corresponding node update its three time series. When each time a new time point t_k is added to the time series associated with the root, a linear predication algorithm is started to predict the time point t_{k+1} at which another cell would become dense. Then, the predict value $\hat{t}_{k+1}$ will be compared with the real value t_{k+1}. If $t_{k+1} - \hat{t}_{k+1} > \theta_t$, then t_k becomes the last time point of the current concept cycle. At the same time, the current TSP trie is stored into disk. A new TSP trie which represents the new concept cycle is built in memory with t_{k+1} as the timestamp for the first dense cell. A TSP trie that represents the previous concept cycle is stored in disk in a compressed form, where each active cell is represented by a triple $((a_n, b_n), (a_m, b_m), (a_d, b_d))$ as described in the previous section. A compressed TSP trie is indexed by a couple (t_{start}, t_{end}), which represents the first and last timestamp maintained by the root. As shown in Sect. 6, we can easily retrieve multiple patterns from stored TSP trie.

5 Pre-Learning Parameters

Obviously, the performance of our work largely depends on the resolution of the grid on the n-dimensional space, as well as the dense threshold θ_n. In order to set these parameters reasonably, one has to gain insights into the particular streaming data. In this section, we present a learning mechanism based on genetic algorithm to enable the system to learn these parameters from a set of static sample data obtained from the data stream. Assume we know the clustering pattern of the sample data, which can be obtained by running clustering algorithms or by visualizing the data or by being assigned by the experts. We encode this clustering pattern into a fitness function. The chromosome encodes the resolution for each dimension and the dense threshold θ_n. Once the fitness function and chromosome structure have been designed, individuals are randomly generated to form the first population. Based on each individual's chromosome setting (resolution for each dimension and starting dense threshold θ_n), cell-based clustering algorithm is conducted to create the dense-based clusters, from which the fitness function is evaluated. Then, the better genes are moved forth to the next generation. This process is repeated for a certain number of iterations to get the final parameter values.

6 Retrieving Patterns from The Stored Concepts

The compressed TSP tries stored in disk support online mining and visualizing both dynamics and historical snapshots of multiple types of patterns.

6.1 Retrieving and Visualizing a Historical Snapshot

Given any past time point t_x, Algorithm 1 shows the steps of retrieving and visualizing the historical snapshot of data at t_x.

Algorithm 1 Retrieving and visualizing the historical snapshot at any time point t_x

1: Retrieve the compressed TSP trie indexed by (t_{start}, t_{end}), where $t_{start} \leq t_x \leq t_{end}$.
2: Calculate the number of points, mean, and standard deviation for each active cell at t_x by using the following formulas

- $n_t = a_n t_x + b_n$
- $m_t = a_m t_x + b_m$
- $d_t = a_d t_x + b_d$

3: Use n_i, m_i, d_i to deploy data in each active cell at the n-dimensional space.

6.2 Retrieving and Visualizing a Historical Concept

Given any past time point t_x, Algorithm 2 shows the steps of retrieving and visualizing the historical concept presented at t_x. An effective way to present a historical concept is to visualize the clustering pattern of this concept. In this paper, a cluster is a continuous dense area with arbitrary shape. We can easily use TSP trie to identify a continuous dense area composed by active cells that are immediate neighbor of at least one another active cell in this area.

Algorithm 2 Retrieve the compressed TSP trie indexed by (t_{start}, t_{end}), where $t_{start} \leq t_x \leq t_{end}$

```
1: Retrieve the compressed TSP trie indexed by (t_start, t_end), where t_start ≤ t_x ≤ t_end.
2: numOfCluster = 0
3: while existing a cell in the TSP trie which is unmarked do
4:    numOfCluster++
5:    Select an unmarked cell c_i
6:    Mark c_i
7:    Insert c_i into list l
8:    while l is not empty do
9:       c = l.pop_front()
10:      cluster[numOfCluster].insert(c)
11:      List neighbor = findImmediateUnmarkedNeighbour(c)
12:      while neighbor is not empty do
13:         d = neighbor.pop_front()
14:         Mark d
15:         Cluster[numOfCluster].insert(d)
16:         l.push_back(d)
17:      end while
18:   end while
19: end while
20: for int i = 1 to numOfCluster do
21:    visualize cluster[i]
22: end for
```

6.3 Retrieving and Visualizing Concept Drifts

Since the dense-based clustering pattern can be used to approximately represent the concept in a concept cycle, by visualizing the shift of clustering patterns, we can detect the concept drift over a time period. Algorithm 3 shows the steps of retrieving and visualizing all the concepts occurring between two past time points.

Algorithm 3 Visualizing concept drift from t_m to t_n

1: Retrieve the compressed TSP trie TSP_i indexed by (t_{istart}, t_{iend}), where $t_{istart} \leq t_m \leq t_{iend}$
2: $Listl.push\_back(TSPi)$
3: $t = t_i end$
4: **while** existing a compressed TSP trie TSP_k indexed by (t_{kstart}, t_{kend}), where $t \leq t_{kstart} \leq t_n$ **do**
5: $Listl.push\_back(TSP_k)$
6: $t = t_{kend}$
7: **end while**
8: **while** l is not empty **do**
9: $TSP\ tsp = l.pop\_front()$
10: Retrieve and visualize clustering pattern from tsp
11: time-delay()
12: **end while**

6.4 Visualizing Micro-shift with a Concept Cycle

We can not only retrieval and visualize cluster pattern for each concept cycle, but also zoom into each active cell to detect the micro-shift of that cell over the concept cycle. Algorithm 4 shows the steps for this purpose.

Besides the above patterns, we can also retrieve attribute correlation patterns, and clustering patterns in certain subspace for a concept cycle, as well as the drifts of the above patterns over multiple cycles based on compressed TSP tries. We will report these algorithms in a separate paper.

Algorithm 4 Visualizing the micro-shift of an active cell $cell_i$ within a concept cycle around t_x

1: Retrieve the compressed TSP trie indexed by (t_{start}, t_{end}), where $t_{start} \leq t_x \leq t_{end}$.
2: Specify the visualization resolution n, based on which generate a list of timestamps $l_t = (t_{start}, t_2, \cdots, t_{n-2}, t_{end})$
3: **while** l_t is not empty **do**
4: $t = l_t.pop\_front()$
5: Calculate the number of points, mean, and standard deviation of $cell_i$ at t by using the following formulas

- $n_t = a_n t_x + b_n$
- $m_t = a_m t_x + b_m$
- $d_t = a_d t_x + b_d$

6: Then use n_i, m_i, d_i to deploy data in $cell_i$ at the n-dimensional space
7: time-delay()
8: **end while**

7 Prelimilary Experimental Results

We conducted experiments on several synthetic data streams. The main goals of our experimentation include, (1) testing the effectiveness and efficiency of our approach to capture concepts and detect concept drifts for continuously multi-dimensional streaming data, and (2) testing the space-efficiency of our storage structure. In order to simulate the functionality of the data stream, we designed a data stream server that will constantly stream the chosen data set to a receiving client.

7.1 Synthetic Datastreams

We created the streaming data in the way that it is steaming concepts one by one along the time. Each concept is composed of several dense areas and mixed with noise data. When streaming a concept, points involved in this concept are randomly picked and send out one by one. Some characteristics of the synthetic data we use are presented in Table 1.

7.2 Pre-Learning Parameters

For each synthetic data stream, the first concept is picked as the training data based on which the dense threshold and the resolution of the grid are learned through applying genetic algorithm. In Table 2, we list the learned parameter values for each data stream.

7.3 Detecting Concepts and Concept Drift

For a data stream, at any given time, a TSP trie for the current concept cycle is maintained in the memory. As described earlier, the root of the TSP

Table 1. Characteristics of the synthetic data streams

Stream no.	Ave. number of clusters per concept	Dimensions	Average data size per concept
1	3.33	8	0.43 MB
2	3.6	8	0.45 MB
3	3.42	8	0.42 MB
4	3.22	8	0.45 MB

Table 2. Pre-learned parameter values

No.	Cell Resolution(Dimension 1-8)								θ_n
	1	2	3	4	5	6	7	8	
1	0.06	0.28	0.73	0.35	0.39	0.76	0.59	0.58	66
2	0.93	0.02	0.11	0.4	0.91	0.80	0.83	0.62	77
3	0.65	0.60	0.81	0.24	0.24	0.06	0.34	0.38	90
4	0.01	0.48	0.129	0.08	0.15	0.25	0.14	0.13	98

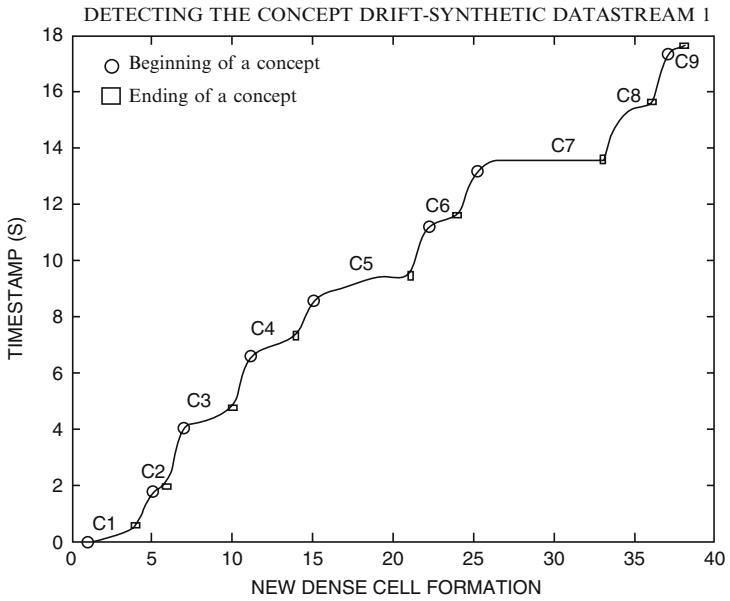


Fig. 3. Detected concepts and concept drifts for stream 1

trie maintains a time series of timestamps at each of which a new cell become active (dense) within this concept cycle. A linear regression algorithm is used to predict the time point when another new active cell is supposed to appear for this concept. A concept drift is reported if there appears a sharp jump of the timestamp. For each data stream, we deploy each recorded time stamp vs. the corresponding total number of dense cells formed so far (see Figs. 3–6 for the deployment results). Each s-circle-marked point in those figures represents the first active cell in a newly appearing concept cycle; and each square-marked point represents the last formed active cell in the current concept cycle. Those marked points are automatically detected by our program. For each concept cycle detected by our program, we retrieve the corresponding TSP trie stored in the disk and based upon the retrieved TSP trie generates the dense-based clusters by applying Algorithm 2. The dense-based clusters we obtained for each concept cycle exactly match the dense areas we designed for the corresponding concept. Therefore, the experimental results show that our algorithm can correctly detect concepts and concept drifts for those synthetic data streams.

7.4 Storage Efficiency of TSP Trie

As described earlier, each historical concept is stored in the disk as a TSP trie indexed by the pair of beginning and ending timestamps. In Table 3, we compare the size of the stored TSP trie and the size of the stream data

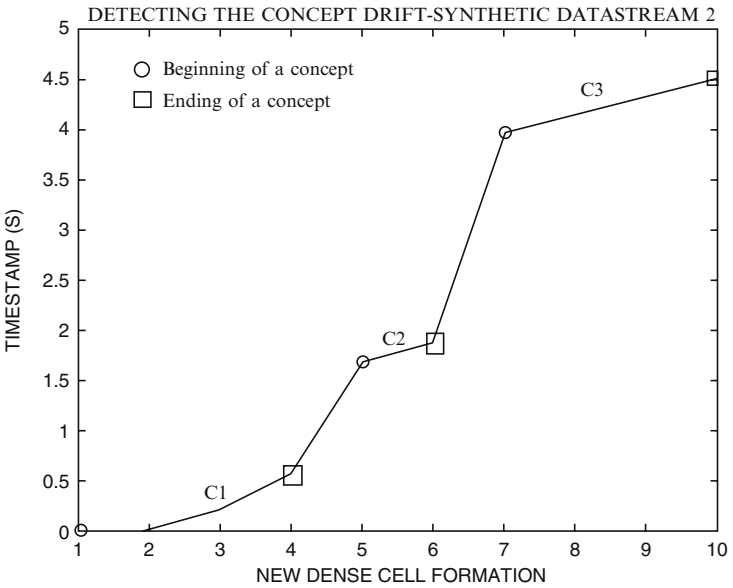


Fig. 4. Detected concepts and concept drifts for stream 2

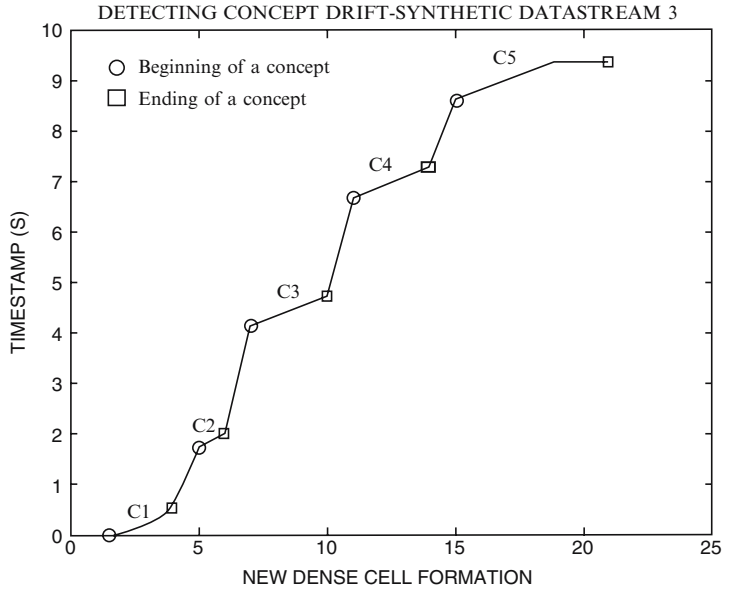


Fig. 5. Detected concepts and concept drifts for stream 3

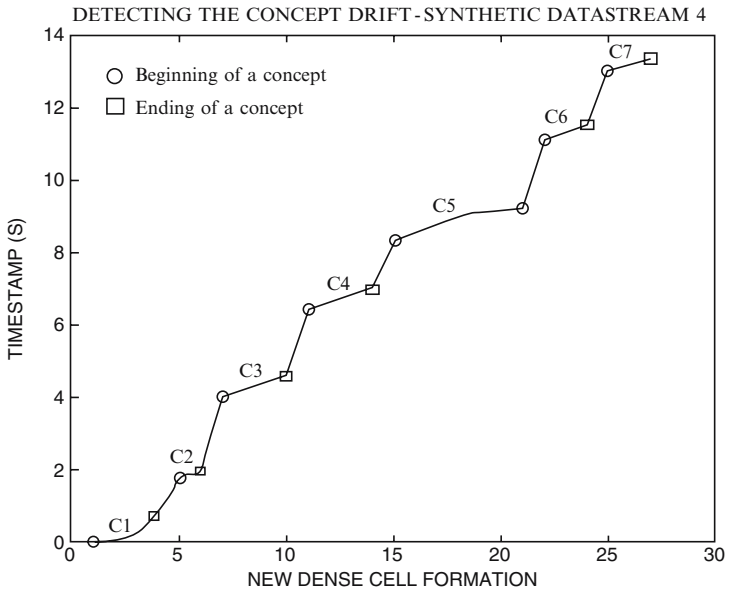


Fig. 6. Detected concepts and concept drifts for stream 4

Table 3. Average TSP trie size vs. average data size for each concept

Stream no.	Ave. TSP trie size	Ave. data size per concept	Compression ratio
1	160 Bytes	0.43 MB	1:2687
2	370 Bytes	0.45 MB	1:1216
3	335 Bytes	0.42 MB	1:1253
4	298 Bytes	0.45 MB	1:1510

involved in the concept represented by that TSP trie, in order to illustrate the storage efficiency of the TSP trie as the synopsis structure for the data stream.

8 Conclusion and Future Works

In this paper, we explored a novel integrated architecture that encapsulates a suit of interrelated data structures and algorithms which support (1) real-time capturing and compressing dynamics of stream data into space-efficient synopses, (2) online mining and visualizing both dynamics and historical snapshots of multiple types of patterns from stored synopses. We also provide preliminary experimental results to illustrate the effectiveness of this proposed approach in capturing concepts and detecting concept drifts from several synthetic data streams. Based on this work, we are aiming at designing

a datastream warehousing system as a comprehensive platform for discovering and retrieving knowledge from potential unbounded, ever-evolving, and multi-dimensional data streams. In order to achieve this goal, we plan to conduct the following tasks: (1) Testing our architecture on various real-life data streams. (2) Studying cross-cycle compression strategy. (3) Formalizing roll-up and drill-down operations on time and other dimensions. (4) Exploring ways to dynamically learning related parameters.

References

1. C. Aggarwal, J. Han, J. Wang, and P.S. Yu, On Demand Classification of Data Streams, *Proceedings of 2004 International Conference on Knowledge Discovery and Data Mining* (Seattle, USA, 2004).
2. P. Domingos, and G. Hulten, Mining High Speed Data Streams, *Proceedings of 2000 ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)* (Boston, USA, 2000).
3. H. Wang, W. Fan, P.S. Yu, and J. Han, Mining Concept Drifting Data Streams using Ensemble Classifiers, *Proceedings of 9th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)* (Washington DC, USA 2003).
4. J. Gehrke, V. Ganti, R. Ramakrishnan, and W. Loh, BOAT-optimistic decision tree construction, *Proceedings of 1999 ACM International Conference on Management of Data (SIGMOD)* (Philadelphia, USA, 1999).
5. J.Z. Kolter, and M.A. Maloof, Dynamic Weighted Majority: A New Ensemble Method for Tracking Concept Drift, *Proceedings of 3rd IEEE International Conf. Data Mining (ICDM)* (Melbourne, Florida, USA, 2003).
6. W.N. Street, and Y. Kim, A Streaming Ensemble Algorithm (SEA) for Large-Scale Classification, *Proceedings of 7th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)* (San Jose, USA, 2001).
7. G. Hulten, L. Spencer, and P. Domingos, Mining Time-Changing Data Streams, *Proceedings of 7th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)* (San Jose, USA, 2001).
8. J. Gama, R. Rocha, and P. Medas, Accurate Decision Trees for Mining High-speed Data Streams, *Proceedings of 9th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)* (Washington DC, USA 2003).
9. C.C. Aggarwal, J. Han, J. Wang, and P.S. Yu, A Framework for Clustering Evolving Data Streams, *Proceedings of the 29th International Conference on Very Large Data Bases (VLDB)* (Berlin, Germany, 2003).
10. L. O'Callaghan, N. Mishra, A. Meyerson, S. Guha, and R. Motawani, Streaming-Data Algorithms for High-Quality Clustering, *Proceedings of 17th IEEE International Conference Data Engineering (ICDE)* (Heidelberg, Germany, 2001).
11. C. Ordonez, Clustering Binary Data Streams with Kmeans, *Proceedings of 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery (DMKD)* (San Diego, USA, 2003).
12. S. Guha, N. Mishra, R. Motwani, and L. O'Callaghan, Clustering data streams, *Proceedings of 41st IEEE Symposium on Foundations of Computer Science* (Redondo Beach, USA, 2000).

13. D. Kifer, A.B. David, and J. Gehrke, Detecting Change in Data Streams, *Proceedings of 13th International Conference on Very Large Data Bases (VLDB)* (Toronto, Canada, 2004).
14. C.C. Aggarwal, A Framework for Diagnosing Changes in Evolving Data Streams, *Proceedings of 2003 ACM International Conference on Management of Data (SIGMOD)* (San Diego, USA, 2003).
15. Y. Chen, G. Dong, J. Han, J. Pei, B. Wah, and J. Wang, On-line Analytical Processing of Data Streams: Is it Feasible, *Proceedings of 2002 Workshop on Research Issues on Data Mining and Knowledge Discovery (DMKD)* (Madison, USA, 2002).

A Conceptual Framework of Data Mining

Yiyu Yao[1], Ning Zhong[2], and Yan Zhao[1]

[1] Department of Computer Science, University of Regina
Regina, Saskatchewan, Canada S4S 0A2
yyao@cs.uregina.ca, yanzhao@cs.uregina.ca
[2] Department of Information Engineering, Maebashi Institute of Technology
460-1, Kamisadori-Cho, Maebashi 371, Japan
zhong@maebashi-it.ac.jp

Summary. The study of foundations of data mining may be viewed as a scientific inquiry into the nature of data mining and the scope of data mining methods. There is not enough attention paid to the study of the nature of data mining, or its philosophical foundations. It is evident that the conceptual studies of data mining as a scientific research field, instead of a collection of isolated algorithms, are needed for a further development of the field. A three-layered conceptual framework is thus proposed, consisting of the philosophy layer, the technique layer and the application layer. Each layer focuses on different types of fundamental questions regarding data mining, and they jointly form a complete characterization of the field. The layered framework is demonstrated by applying it to three sub-fields of data mining, classification, measurements, and explanation-oriented data mining.

1 Introduction

With the development and success of data mining, many researchers became interested in the fundamental issues, namely, the foundations of data mining [2, 7, 8, 22]. The study of foundations of data mining should be viewed as a scientific inquiry into the *nature* of data mining and the scope of data mining *methods*. This simple view separates two important issues. The study of the nature of data mining concerns the philosophical, theoretical and mathematical foundations of data mining; while the study of data mining methods concerns its technological foundations by focusing on the algorithms and tools.

A review of the existing studies show that not enough attention has been paid to the study of the nature of data mining, more specifically, to the philosophical foundations of data mining [22]. Although three dedicated international workshops have been held [6–8], there still do not exist well-accepted and non-controversial answers to many basic questions, such as, what is data mining? what makes data mining a distinctive study? what are the foundations of data mining? What is the scope of the foundations of data mining?

Y. Yao et al.: *A Conceptual Framework of Data Mining*, Studies in Computational Intelligence (SCI) **118**, 501–515 (2008)
www.springerlink.com

What are the differences, if any, between the existing researches and the research on the foundations of data mining? The study of foundations of data mining attempts to answer these questions.

The foundational study is sometimes ignored or underestimated. In the context of data mining, one is more interested in algorithms for finding knowledge, but not what is knowledge, and what is the knowledge structure. One is often more interested in a more implementation-oriented view or framework of data mining, rather than a conceptual framework for the understanding of the nature of data mining. The following quote from Salthe [16] about studies of ecosystem is equally applicable to the studies of data mining:

> The question typically is not what is an ecosystem, but how do we measure certain relationships between populations, how do some variables correlate with other variables, and how can we use this knowledge to extend our domain. The question is not what is mitochondrion, but what processes tend to be restricted to certain region of a cell [page 3].

A lack of the study of its foundation may affect the future development of the field.

There are many reasons accounting for such unbalanced research efforts. The problems of data mining are first raised by very practical needs for finding useful knowledge. One inevitably focuses on the detailed algorithms and tools, without carefully considering the problem itself. A workable program or software system is more easily acceptable by, and at the same time is more concrete and more easily achievable by, many computer scientists than an in-depth understanding of the problem itself. Furthermore, the fundamental questions regarding the nature of the field, the inherent structure of the field and its related fields, are normally not asked at its formation stage. This is especially true when the initial studies produce useful results [16].

The study of foundations of data mining therefore needs to adjust the current unbalanced research efforts. We need to focus more on the understanding of the nature of data mining as a field instead of a collection of algorithms. We need to define precisely the basic notions, concepts, principles, and their interactions in an integrated whole. Many existing studies can contribute to the foundational study of data mining. Here are two examples: (a) Results from the studies of cognitive science and education are relevant to such a purpose. Posner suggested that, according to the cognitive science approach, to learn a new field is to build appropriate cognitive structures and to learn to perform computations that will transform what is known into what is not yet known [14]. (b) Reif and Heller showed that knowledge structure of a domain is very relevant to problem solving [15]. In particular, knowledge about a domain, such as mechanics, specifies descriptive concepts and relations described at various levels of abstraction, is organized hierarchically, and is accompanied by explicit guidelines that specify when and how knowledge is to be applied [15]. The knowledge hierarchy is used by Simpson for the study

of foundations of mathematics [19]. It follows that the study of foundations of data mining should focus on the basic concepts and knowledge of data mining, as well as their inherent connections, at multi-level of abstractions. Without such kind of understanding of data mining, one may fail to make further progress.

In summary, in order to study the foundations of data mining, we need to move beyond the existing studies. More specifically, we need to introduce a conceptual framework, to be complementary to the existing implementation and process-oriented views. The main objective of this chapter is therefore to introduce such a framework.

The rest of the chapter is organized as follows. In Sect. 2, we re-examine the existing studies of data mining. Based on the examination, we can observe several problems and see the needs for the study of foundations of data mining. More specifically, there is a need for a framework, within which to study the basic concepts and principles of data mining, and the conceptual structures and characterization of data mining. For this purpose, in Sect. 3, a three-layered conceptual framework of data mining is discussed, consisting of the philosophy layer, the technique layer, and the application layer [22]. The relationships among the three layers are discussed. The layered framework is demonstrated in Sect. 4 by applying it to an example of function-oriented view – classification, an example of theory-oriented view – measurement theory, and an example of procedure-oriented view – explanation-oriented data mining. Section 5 draws the conclusion.

2 Overview of the Existing Studies and the Problems

Data mining, as a relatively new branch of computer science, has received much attention. It is motivated by our desire of obtaining knowledge from huge databases. Many data mining methods, based on the extensions, combinations, and adaptation of machine learning algorithms, statistical methods, relational database concepts, and the other data analysis techniques, have been proposed and studied for knowledge extraction and abstraction.

2.1 Three Views of Data Mining

The existing studies of data mining can be classified broadly under three views.

The function-oriented view

The function-oriented view focuses on the goal or functionality of a data mining system, namely, the discovery of knowledge from data. In a well-accepted definition, data mining is defined as "the non-trivial process of identifying

valid, novel, potentially useful, and ultimately understandable patterns from data" [3]. The function-oriented approaches put forth efforts on searching, mining and utilizing different patterns embedded in various databases. A pattern is an expression in a language that describes data, and has a representation simpler than the data. For example, frequent itemsets, association rules and correlations, as well as clusters of data points, are common classes of patterns. Such goal-driven approaches establish a close link between data mining research and real world applications.

Depending on the data and their properties, one may consider different data mining systems with different functionalities and for different purposes, such as text mining, Web mining, sequential mining, and temporal data mining. Under the function-oriented view, the objectives of data mining can be divided into prediction and description. Prediction involves the use of some variables to predict the values of some other variables, and description focuses on patterns that describe the data [3].

The Theory-Oriented View

The theory-oriented approaches concentrate on the theoretical studies of data mining, and its relationship to the other disciplines. Many models of data mining have been proposed, critically investigated and examined from the theory-oriented point of view [3, 11, 21, 26].

Conceptually, one can draw a correspondence between scientific research by scientists and data mining by computers [25, 26]. More specifically, they share the same goals and processes. It follows that any theory discovered and used by scientists can be used by data mining systems. Thus, many fields contribute to the theoretical study of data mining. They include statistics, machine learning, databases, pattern recognition, visualization, and many other. There is also a need for the combination of existing theories. For example, some efforts have been made to bring the rough sets theory, fuzzy logic, utility and measurement theory, concept lattice and knowledge structure, and other mathematical and logical models into the data mining models.

The Procedure/Process-Oriented View

From the procedure/process-oriented view, data mining deals with a "nontrivial" process consisting of many steps, such as data selection, data preprocessing, data transformation, pattern discovery, pattern evaluation, and result explanations [3, 10, 26, 30]. Furthermore, it should be a dynamically organized process.

Under the process-oriented view, data mining studies have been focused on algorithms and methodologies for different processes, speeding up existing algorithms, and evaluation of discovered knowledge.

2.2 Problems and Potential Solutions

The three views jointly provide a good description of data mining research. The function-oriented view states the goals of data mining, the theory-oriented view studies the means with which one can carry out the desired tasks, and the process-oriented view deals with how to achieve the goals based on the theoretical means. However, the general conceptual framework is still not proposed and examined.

Intuitively, the terms of technology and science have different meanings. Science studies the nature of the world. On the other hand, technology studies the ways that people develop to control or manipulate the world. Science deals with "understanding" while technology deals with "doing". The scientific study requires the study of foundations of data mining, so that the fundamental questions of the field itself are asked, examined, explained and formalized.

The foundations of data mining may not be solely mathematics or logic, or any other individual fundamental disciplines. Considering the different types of databases, the diversity of patterns, the ever changing techniques and algorithms, and the different views, we require a multilevel (or multi-layer) understanding of data mining. By viewing data mining in many layers, one can identify the inherent structure of the fields, and put fundamental questions into their proper perspectives in the conceptual map of data mining.

3 A Three-Layered Conceptual Framework

A three-layered conceptual framework is recently proposed by Yao [22], which consists of the philosophy layer, the technique layer, and the application layer. The layered framework represents the understanding, discovery, and utilization of knowledge, and is illustrated in Fig. 1.

3.1 The Philosophy Layer

The philosophy layer investigates the basic issues of knowledge. One attempts to answer a fundamental question, namely, what is knowledge? There are many related issues to this question, such as the representation of knowledge, the expression and communication of knowledge in languages, the relationship between knowledge in mind and in the external real world, and the classification and organization of knowledge [20]. Philosophical study of data mining serves as a precursor to technology and application, it generates knowledge and the understanding of our world, with or without establishing the operational boundaries of knowledge. The philosophy layer study is primarily driven by curiosity, and responds to a certain hypothesis.

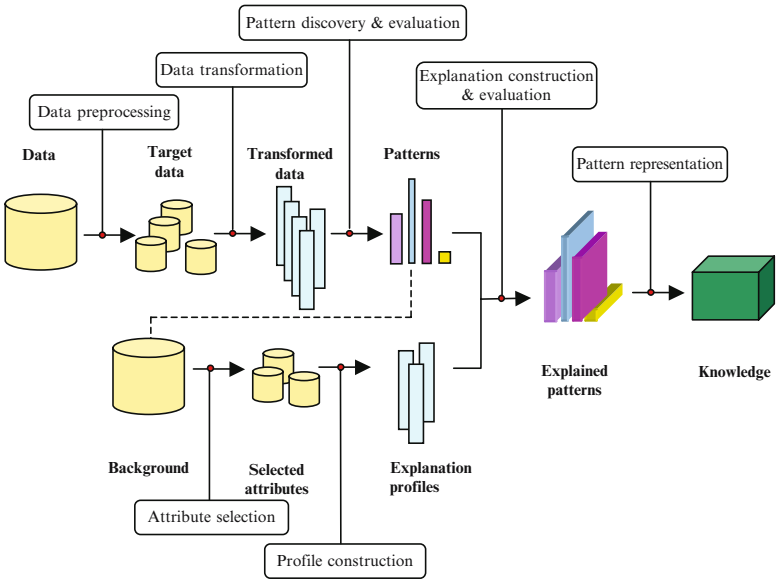


Fig. 1. The three-layered conceptual framework of data mining

3.2 The Technique Layer

The technique layer is the study of knowledge discovery in machine. One attempts to answer the question, how to discover knowledge? In the context of computer science, there are many issues related to this question, such as the implementation of human knowledge discovery methods by programming languages, which involves coding, storage and retrieval issues in a computer, and the innovation and evolution of techniques and algorithms in intelligent systems. The main streams of research in machine learning, data mining, and knowledge discovery have concentrated on the technique layer. Logical analysis and mathematical modeling are considered to be the foundations of technique layer study of data mining.

3.3 The Application Layer

The ultimate goal of knowledge discovery is to effectively use discovered knowledge. The question that needs to be answered is how to utilize the discovered knowledge. The application layer therefore should focus on the notions of "usefulness" and "meaningfulness" of discovered knowledge for the specific domain, and aim at many attributes, such as efficiency, optimization, reliability, cost-effectiveness, and appropriateness. These notions cannot be discussed in total isolation with applications, as knowledge in general is domain specific. The application layer therefore involves the design and development of a solution to a target problem that serves a real need.

3.4 The Relationships among the Three Layers

Two points need to be emphasized about the three-layered conceptual framework.

First, the three layers are different, relatively independent, and self-contained.

(1) The philosophical study does not depend on the availability of specific techniques and applications. In other words, it does not matter whether knowledge is discovered or not, utilized or not, and if the knowledge structure and expression are recognized or not. Furthermore, all human knowledge is conceptual and forms an integrated whole [13]. The output of the philosophical study can be expressed as theories, principles, concepts or other knowledge structures. Knowledge structure is built by connecting new bits of information to the old. The study of knowledge at the philosophy layer has important implications for the human society, even if it is not discovered or utilized yet, or it simply provides a general understanding of the real world.
(2) The technical study can carry out part of the philosophic study results but not all, and it is not constrained by applications. The philosophy layer describes a very general conceptual scheme. The current techniques, including hardware and software, may still be insufficient to bring all of it into reality. On the other hand, the existence of a technique/algorithm does not necessarily imply that discovered knowledge is meaningful and useful. The output of the technical study can be expressed by algorithms, mathematical models, and intelligent systems. The technology can be commercialized. The benefits of technological implementation and innovation tend to move the study of technical layer to be more and more profit-driven.
(3) The applications of data mining is the utilization of knowledge in specific domains. They are related to the evaluation of discovered knowledge, the explanation and interpretation of discovered knowledge in a particular domain. The discovered knowledge can be used in many ways. For example, knowledge from transaction databases can be used for designing new products, distributing, and marketing. Comparing to the philosophical and technological studies, the applications have more explicit targets and schedules.

Second, the three layers mutually support each other and jointly form an integrated whole.

(1) It is expected that the results from philosophy layer will provide guidelines and set the stage for the technique and application layers. The technology development and innovation can not go far without the conceptual guidance.
(2) The philosophical study cannot be developed without the consideration of reality. Technology development may raise new philosophical questions

and promote the philosophical study. Technique layer is the bridge between philosophical view of knowledge and the application of knowledge.

(3) The applications of philosophical and technical outcomes give an impetus for the re-examination of philosophical and technical studies too. The feedbacks from applications provide evidence for the confirmation, re-examination, and modification of philosophical and technical results.

Three layers of the conceptual framework are tightly integrated, namely, they are mutually connected, supported, promoted, facilitated, conditioned and restricted. The division between the three layers is not a clear cut, and may overlap and interweave with each other. Any of them is indispensable in the study of intelligence and intelligent systems. They must be considered together in a common framework through multi-disciplinary studies, rather than in isolation.

4 Enriched Views on the Three-Layered Framework

We believe the three-layered conceptual framework establishes a proper foundation of data mining. It can promote the progress of data mining; to advance the science and technology related to data mining. By putting the three existing views that we have discussed in Sect. 2 into the conceptual framework, we obtain a 3(view)-by-3(layer) Fig. 2 and more insights.

We study, from the function-oriented view, not only what data mining programs can and cannot do in the technology and application layer, but also in the abstract, what knowledge is possibly stored in the information system, how programs should store and retrieve specific kinds of information in their specific structure and representation. From the theory-oriented view, the key thing is not only to pile up various theories, algorithms, to beat up the efficiency, or even worse, to reinvent theories and methodologies that have been well-studied in the other domains. It emphasizes the understanding and human–computer interaction. From the procedure-oriented view, it should be guided by the procedure of general scientific research too.

Layer / View	Philosophy layer	Technique layer	Application layer
Function-oriented view	What knowledge can be discovered? What is the nature of the knowledge to be discovered?	How to discover this type of knowledge?	How to use this type of knowledge for the real life?
Theory-oriented view	Which theory is related to data mining? How related is it? What is the benefit? What is the expense?	How to implement this theory for data mining?	How to implement this theory for data mining in real life?
Procedure-oriented view	What is the connection between the procedure of general scientific research and the specific procedure of data mining?	How to implement each process, or a particular process, of data mining?	In real applications, some processes are focused on.

Fig. 2. Three views and three layers

In this section, we explain three examples with respect to the three-layered framework. The classification is corresponding to the function-oriented view, evaluation and measurement theory is corresponding to the theory-oriented view, and the explanation-oriented data mining extends the procedure-oriented view. We demonstrate how these three views are enriched by the conceptual framework, especially the philosophy layer study.

4.1 Classification on the Three-Layered Framework

Classification is considered as one of the most important tasks of data mining. It is therefore associated with the function-oriented view that we discussed in Sect. 2. Partitions and coverings are two simple and commonly used knowledge classifications of the universe. A partition of a finite universe is a collection of non-empty, and pairwise disjoint subsets whose union is the universe. A covering of a finite universe is a collection of non-empty and possibly overlapped subsets whose union is the universe. Partitions are a special case of coverings.

Knowledge is organized in a tower (hierarchy) or a partial ordering. Based on the above discussion, we have partition-based hierarchy and covering-based hierarchy. Hierarchy means that the base or minimal elements of the ordering are the most fundamental concepts and higher-level concepts depend on lower-level concepts [19]. Partial ordering means that the concepts in the hierarchy are reflexive, anti-symmetric and transitive. The first-level concept is formed directly from the perceptual data [13]. The higher-level concepts, representing a relatively advanced state of knowledge, are formed by a process of abstracting from abstractions [13]. On the other hand, the series of lower-level concepts, on whom the higher-level concept is formed, are not necessarily unique in content. Within the requisite overall structure, there may be many alternatives in detail [13].

The natural process of knowledge cognition follows the hierarchy from lower-level concepts to higher-level according to the intellectual dependency. The revise process does exist because of impatience, anti-effort, or simple error. Peikoff analyzes that the attempt to function on the higher levels of complex structure without having established the requisite base will build confusion on confusion, instead of knowledge on knowledge. In such minds, the chain relating higher-level content to perceptual reality is broken [13].

A virtual space that can hold knowledge as concepts is called a concept space, namely, it refers to the set or class of the concepts. If we consider the data mining process as searching for concepts in a particular concept space, we need to study different kinds of concept spaces first. Inside the concept space, the concept can be represented and discovered. Generally, a concept space S can hold all the concepts, including the ones that can be defined as a formula, and the ones that cannot. A definable concept space DS is a sub-space of the concept space S. There are many definable concept spaces in different forms. In most situations, one is only interested in the concepts in a certain form. Consider the class of conjunctive concepts, that formula

constructed from atomic formula by only logic connective $\wedge$. A concept space CDS is then referred to as the conjunctively definable space, which is a subspace of the definable space DS. Similarly, a concept space is referred to as a disjunctively definable space if the atomic formulas are connected by logic disjunctive $\vee$.

In [29], we discuss the complete concept space for classification tasks using granular network concepts. The immediate result is that a classification task can be understood as a search of the distribution of classes in a granule network defined by the descriptive attribute set. The analysis shows that the complexity of the search space of a consistent classification task is not polynomially bounded. This can be extremely complex especially when the number of possible values of attributes are large, let alone continuous. This forces us to use heuristic algorithms to quickly find solutions in a constrained space. Indeed, the existing heuristic algorithms perform very well. Each of them can be understood as a particular heuristic search within the granule network.

4.2 Rule Interestingness Evaluation on the Three-Layered Framework

Traditionally, when we talk about evaluating the usefulness and interestingness of discovered rules and patterns, we talk about many measures based on, for example, information theory and measurement theory. Thus, the study of interestingness evaluation is a theory-oriented study referring to the categories in Sect. 2.

With respect to the framework, in the philosophical layer, quantitative measures can be used to characterize and classify different types of rules. In the technique layer, measures can be used to reduce search space. In the application layer, measures can be used to quantify the utility, profit, effectiveness, or actionability of discovered rules.

From the existing studies, one can observe that rule evaluation plays at least three different types of roles:

i. In the data mining phase, quantitative measures can be used to reduce the size of search space. An example is the use of well-known support measure, which reduces the number of itemsets that need to be examined [1].
ii. In the phase of interpreting mined patterns, rule evaluation plays a role in selecting the useful or interesting rules from the set of discovered rules [17, 18]. For example, the confidence measure of association rules is used to select only strongly associated itemsets [1, 9].
iii. In the phase of consolidating and acting on discovered knowledge, rule evaluation can be used to quantify the usefulness and effectiveness of discovered rules. Many measures such as cost, classification error, and classification accuracy play such a role [4].

To carry out the above three roles, many measures have been proposed and studied. We need to understand that measures can be classified into two

categories consisting of objective measures and subjective measures [17]. Objective measures depend only on the structure of rules and the underlying data used in the discovery process. Subjective measures also depend on the user who examines the rules [17]. In comparison, there are limited studies on subjective measures. For example, Silberschatz and Tuzhilin proposed a subjective measure of rule interestingness based on the notion of unexpectedness and in terms of a user belief system [17, 18].

Yao et al. [23] suggest that, the rule interestingness measures have three forms: statistical, structural and semantic. Many measures, such as support, confidence, independence, classification error, etc., are defined based on statistical characteristics of rules. A systematic analysis of such measures is given by Yao et al. using a 2×2 contingency table induced by a rule [24, 27]. The structural characteristics of rules have been considered in many measures. For example, information, such as the disjunct size, attribute interestingness, the asymmetry of classification rules, etc., can be used [4]. These measures reflect the simplicity, easiness of understanding, or applicability of rules. Although statistical and structural information provides an effective indicator of the potential effectiveness of a rule, its usefulness is limited. One needs to consider the semantics aspects of rules or explanations of rules [25]. Semantics centered approaches are application and user dependent. In addition to statistical information, one incorporates other domain specific knowledge such as user interest, utility, value, profit, actionability, and so on.

Measures defined by statistical and structural information may be viewed as objective measures. They are user, application and domain independent. For example, a pattern is deemed interesting if it has certain statistical properties. These measures may be useful in philosophy layer of the three-layered framework. Different classes of rules can be identified based on statistical characteristics, such as peculiarity rules (low support and high confidence), exception rules (low support and high confidence, but complement to other high support and high confidence rules), and outlier patterns (far away from the statistical mean) [28]. Semantic based measures involve the user interpretation of domain specific notions such as profit and actionability. They may be viewed as subjective measures. Such measures are useful in the application layer of the three-layered framework. The usefulness of rules are measured and interpreted based on domain specific notions.

4.3 Explanation-Oriented Data Mining on the Three-Layered Framework

To complement the extensive studies of various tasks of data mining, the explanation task of data mining, more specifically, the concept of explanation-oriented data mining, was first proposed in [26]. Some technologies of data mining cannot immediately create *knowledge* or guarantee knowledge generation, but only retrieve, sort, quantify, organize and report information out of

data. Information can turn into knowledge if it can be rationalized, explained and validated. Similarly, explanation-oriented data mining can be explored with respect to the three-layered framework.

To add the explanation task into the existing data mining process is based on an important observation, that scientific research and data mining have much in common in terms of their goals, tasks, processes and methodologies. Scientific research is affected by the perceptions and the purposes of science. Martella et al. summarized the main purposes of science, namely, to describe and predict, to improve or manipulate the world around us, and to explain our world [12]. The results of the scientific research process provide a description of an event or a phenomenon. The knowledge obtained from research helps us to make predictions about what will happen in the future. Research findings are a useful tool for making an improvement in the subject matter. Research findings also can be used to determine the best or the most effective ways of bringing about desirable changes. Finally, scientists develop models and theories to explain why a phenomenon occurs.

Goals similar to those of scientific research have been discussed by many researchers in data mining. Guergachi stated that the goal of data mining is what science is and has been all about: discovering and identifying relationships among the observations we gather, making sense out of these observations, developing scientific principles, building universal laws from observations and empirical data [5]. Fayyad et al. identified two high-level goals of data mining as prediction and description [3]. Ling et al. studied the issue of manipulation and action based on the discovered knowledge [8]. Yao et al. compared the research process and data mining process [25, 26]. The comparison led to the introduction of the notion of the explanation-oriented data mining, which focuses on constructing models for the explanation of data mining results [26].

In the philosophy level, we need to understand what kind of pattern need to be explained. A target pattern that arouses user's interest, hooks up user's attention and needs to be deep explained resides in the set of discovered patterns. In fact, the targets may differ among the views of individuals. One may question the same target at different times based on different considerations, at different cognitive levels. We also need to understand what knowledge can be applied to explain the target pattern. Explanation-oriented data mining needs background knowledge to infer features that can possibly explain a discovered pattern. We need to note two facts: first, the explanation profiles may not situate in the original dataset. One needs to collect additional information that is characteristically associated with the target pattern, and can be practically transformed to be the explanation context. On the other hand, how one explains determines what one may learn. In general, the better one's background knowledge is, the more accurate the inferred explanations are likely to be.

The task of explanation construction and evaluation includes five separate phases that need to be undertaken: explanation subject selection, explanation profiles proposition, explanation construction, explanation evaluation, and explanation refinement. That is, given a target discovered pattern to be

explained, in order to result an explanatory account one needs to do the following: First, propose the heuristic for some explanation profiles, transform and associate them with the environment where the target pattern is located. Then, construct some rules by a particular method in the explanation context. After these, the learned results need to be evaluated. According to the evaluation results and the user feedback, the explanation profiles can be sharpened and refined, the same or different methods can be applied to the refined context for another plausible explanation construction until it is satisfied.

5 Conclusion

A three-layered conceptual framework of data mining is discussed in this chapter, consisting of the philosophy layer, the technique layer and the application layer. The philosophy layer deals with the formation, representation, evaluation, classification and organization, and explanation of knowledge; the technique layer deals with the technique development and innovation; and the application layer emphasizes on the application, utility and explanation of mined knowledge. The layered framework focuses on the data mining questions and issues in different abstract levels, and thus, offers us opportunities and challenges to reconsider the existing three views of data mining. The framework is aimed at the understanding of the data mining as a field of study, rather than a collection of theories, algorithms and tools.

References

1. Agrawal, R., Imielinski, T. and Swami, A., Mining association rules between sets of items in large databases, *Proceedings of ACM SIGMOD*, 207–216, 1993.
2. Chen, Z., The three dimensions of data mining foundation, *Proceedings of IEEE ICDM'02 Workshop on Foundation of Data Mining and Knowledge Discovery*, 119–124, 2002.
3. Fayyad, U.M., Piatetsky-Shapiro, G. and Smyth, P., From data mining to knowledge discovery: an overview, in: *Advances in Knowledge Discovery and Data Mining*, Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P. and Uthurusamy, R. (eds.), AAAI/MIT, Menlo Park, CA, 1–34, 1996.
4. Freitas, A.A., On rule interestingness measures, *Knowledge-Based Systems*, **12**, 309–315, 1999.
5. Guergachi, A.A., Connecting traditional sciences with the OLAP and data mining paradigms, *Proceedings of SPIE: Data Mining and Knowledge Discovery: Theory, Tools, and Technology*, 5098, 226–234, 2003.
6. Lin, T.Y., Hu, X.H., Ohsuga, S. and Liau, C.J. (eds.), *Proceedings of IEEE ICDM'03 Workshop on Foundation of New Directions in Data Mining*, 2003.
7. Lin, T.Y. and Liau, C.J. (eds.), *Proceedings of the PAKDD'02 Workshop on Fundation of Data Mining, Communications of Institute of Information and Computing Machinery*, **5**, 101–106, 2002.

8. Lin, T.Y. and Ohsuga, S. (eds.), *Proceedings of IEEE ICDM'02 Workshop on Foundation of Data Mining and Knowledge Discovery*, 2002.
9. Lin, T.Y., Yao, Y.Y. and Louie, E., Value added association rules, *Proceedings of PAKDD*, 328–333, 2002.
10. Mannila, H., Methods and problems in data mining, *Proceedings of International Conference on Database Theory*, 41–55, 1997.
11. Mannila, H., Theoretical frameworks for data mining, *SIGKDD Explorations*, **1**, 30–32, 2000.
12. Martella, R.C., Nelson, R. and Marchand-Martella, N.E., *Research Methods: Learning to Become a Critical Research Consumer*, Allyn & Bacon, Bosten, 1999.
13. Peikoff, L., *Objectivism: The Philosophy of Ayn Rand*, Dutton, 1991.
14. Posner, M.I. (ed.), *Foundations of Cognitive Science*, Preface: learning cognitive science, MIT, Cambridge, Massachusetts, 1989.
15. Reif, F. and Heller, J.I., Knowledge structure and problem solving in physics, *Educational Psychologist*, **17**, 102–127, 1982.
16. Salthe, S.N., *Evolving Hierarchical Systems, their Structure and Representation*, Columbia University Press, New York, 1985.
17. Silberschatz, A. and Tuzhilin, A., On subjective measures of interestingness in knowledge discovery, *Proceedings of KDD*, 275–281, 1995.
18. Silberschatz, A. and Tuzhilin, A., What makes patterns interesting in knowledge discovery systems? *IEEE Transactions on Knowledge and Data Engineering*, **8**, 970–974, 1996.
19. Simpson, S.G., What is foundations of mathematics? 1996. http://www.math.psu.edu/simpson/hierarchy.html, retrieved November 21, 2003.
20. Sowa, J.F., *Conceptual Structures, Information Processing in Mind and Machine*, Addison-Wesley, Reading, Massachusetts, 1984.
21. Yao, Y.Y., Modeling data mining with granular computing, *Proceedings of the 25th Annual International Computer Software and Applications Conference (COMPSAC 2001)*, 638–643, 2001.
22. Yao, Y.Y., A step towards the foundations of data mining, *Proceedings of the SPIE: Data Mining and Knowledge Discovery: Theory, Tools, and Technology*, 5098, 254–263, 2003.
23. Yao, Y.Y., Chen, Y.H. and Yang X.D., Measurement-theoretic foundation for rules interestingness, *ICDM 2003 Workshop on Foundations of Data Mining*, 221–227, 2003.
24. Yao, Y.Y. and Liau, C.J., A generalized decision logic language for granular computing, *FUZZ-IEEE on Computational Intelligence*, 1092–1097, 2002.
25. Yao, Y.Y. and Zhao, Y., Explanation-oriented data mining, in: Wang, J. (ed.), *Encyclopedia of Data Warehousing and Mining*, 492–297, Idea Group Inc., 2005.
26. Yao, Y.Y., Zhao, Y. and Maguire, R.B., Explanation-oriented association mining using rough set theory, *Proceedings of the 9th International Conference of Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing*, 165–172, 2003.
27. Yao, Y.Y. and Zhong, N., An analysis of quantitative measures associated with rules, *Proceedings of PAKDD'99*, 479–488, 1999.
28. Yao, Y.Y., Zhong, N. and Ohshima, M., An analysis of peculiarity oriented multi-database mining, *IEEE Transactions on Knowledge and Data Engineering*, **15**, 952–960, 2003.

29. Zhao, Y. and Yao, Y.Y., Interactive classification using a granule network, *Proceedings of ICCI'05*, 250–259, 2005.
30. Zhong, N., Liu, C. and Ohsuga, S., Dynamically organizing KDD processes, *International Journal of Pattern Recognition and Artificial Intelligence*, **15**, 451–473.

How to Prevent Private Data from being Disclosed to a Malicious Attacker

Justin Zhan[1], LiWu Chang[2], and Stan Matwin[3]

[1] Carnegie Mellon University
justinzh@andrew.cmu.edu
[2] Naval Research Laboratory
lchang@itd.nrl.navy.mil
[3] University of Ottawa
stan@site.uottawa.ca

1 Introduction

In this paper,[1] we address the following problem: multiple parties are cooperating on a data-rich task. Each of the parties owns data pertinent to the aspect of the task addressed by this party. More specifically, the data consists of instances, each party owns her instances but all parties have the same attributes. The overall performance, or even solvability, of this task depends on the ability of performing data mining using all the instances of all the parties. The parties, however, may be unwilling to release their instances to other parties, due to privacy or confidentiality of the data. How can we structure information sharing between the parties so that the data will be shared for the purpose of data mining, while at the same time specific instance values will be kept confidential by the parties to whom they belong? This is the task addressed in this paper. In the privacy-oriented data mining this task is known as data mining with horizontally partitioned data (also known as homogeneous collaboration [15]). Examples of such tasks abound in business, homeland security, coalition building, medical research, etc.

The following scenarios illustrate situations in which this type of collaboration is interesting: (i) Multiple competing supermarkets, each having an extra large set of data records of its customers' buying behaviors, want to conduct data mining on their joint data set for mutual benefit. Since these companies are competitors in the market, they do not want to disclose their customers' information to each other, but they know the results obtained from this collaboration could bring them an advantage over other competitors. (ii) Success of homeland security aiming to counter terrorism depends on combination of strength across different mission areas, effective international collaboration

[1] The preliminary version of this paper has been published [32].

J. Zhan et al.: *How to Prevent Private Data from being Disclosed to a Malicious Attacker*, Studies in Computational Intelligence (SCI) **118**, 517–528 (2008)
www.springerlink.com

and information sharing to support coalition in which different organizations and nations must share some, but not all, information. Information privacy thus becomes extremely important: all the parties of the collaboration promise to provide their private data to the collaboration, but neither of them wants each other or any other party to learn much about their private data.

Without privacy concerns, all parties can send their data to a trusted central place to conduct the mining. However, in situations with privacy concerns, the parties may not trust anyone. We call this type of problem the *Privacy-preserving Collaborative Data Mining problem*. As stated above, in this paper we are interested in homogeneous collaboration where each party has the same sets of attributes [15] but has different sets of instances.

Data mining includes a number of different tasks, such as association rule mining, classification, and clustering, etc. This paper studies how to learn support vector machines. In the last few years, there has been a surge of interest in Support Vector Machines (SVM) [28, 29]. SVM is a powerful methodology for solving problems in nonlinear classification, function estimation and density estimation which has also led to many other recent developments in kernel based learning methods in general [7, 24, 25]. SVMs have been introduced within the context of statistical learning theory and structural risk minimization. As part of the SVM algorithm, one solves convex optimization problems, typically quadratic programs. It has been empirically shown that SVMs have good generalization performance on many applications such as text categorization [13], face detection [20], and handwritten character recognition [16]. Based on the existing SVM learning technologies, we study the problem of learning Support Vector Machines on private data. More precisely, the problem is defined as follows: multiple parties want to build support vector machines on a data set that consists of private data of all the parties, but none of the parties is willing to disclose her raw data to each other or any other parties. We develop a secure protocol, based on homomorphic cryptography and random perturbation techniques, to tackle the problem. An important feature of our approach is its distributed character, i.e. there is no single, centralized authority that all parties need to trust. Instead, the computation is distributed among parties, and its structure and the use of homomorphic encryption ensures privacy of the data.

The paper is organized as follows: The related work is discussed in Sect. 2. We describe the SVMs training procedure in Sect. 3. We then present our proposed secure protocols in Sect. 4. We give our conclusion in Sect. 5.

2 Related Work

2.1 Secure Multi-Party Computation

A Secure Multi-party Computation (SMC) problem deals with computing any function on any input, in a distributed network where each participant holds one of the inputs, while ensuring that no more information is revealed to a

participant in the computation than can be inferred from that participant's input and output. The SMC problem literature was introduced by Yao [31]. It has been proved that for any polynomial function, there is a secure multi-party computation solution [12]. The approach used is as follows: the function F to be computed is firstly represented as a combinatorial circuit, and then the parties run a short protocol for every gate in the circuit. Every participant gets corresponding shares of the input wires and the output wires for every gate. This approach, though appealing in its generality and simplicity, is highly impractical for large datasets.

2.2 Privacy-Preserving Data Mining

In early work on privacy-preserving data mining, Lindell and Pinkas [17] propose a solution to privacy-preserving classification problem using oblivious transfer protocol, a powerful tool developed by secure multi-party computation (SMC) research. The techniques based on SMC for efficiently dealing with large data sets have been addressed in [14], where a solution to the association rule mining problem for the case of two parties was proposed.

Randomization approaches were firstly proposed by Agrawal and Srikant in [3] to solve privacy-preserving data mining problem. In addition to perturbation, aggregation of data values [26] provides another alternative to mask the actual data values. In [1], authors studied the problem of computing the kth-ranked element. Dwork and Nissim [9] showed how to learn certain types of boolean functions from statistical databases in terms of a measure of probability difference with respect to probabilistic implication, where data are perturbed with noise for the release of statistics. In this paper, we focus on privacy-preserving among the inter-party computation.

Homomorphic encryption [21], which transforms multiplication of encrypted plaintexts into the encryption of the sum of the plaintexts, has recently been used in secure multi-party computation. For instance, Freedmen, Nissim and Pinkas [10] applied it for set intersection. For computing set intersection, unlike [2,10,27] proposed an approach based on commutative encryption. The work most related to ours is [30], where Wright and Yang applied homomorphic encryption [21] to the Bayesian networks induction for the case of *two* parties. The work that are closely related ours is [33], where Zhan et al. present secure protocols for learning support vector machine over vertically partitioned data. In this paper, we develop a secure protocol, based on homomorphic encryption and random perturbation techniques, for multiple parties to build SVMs over horizontally partitioned data without compromising their data privacy.

3 Learning SVMs on Private Data

Support vector machines were invented by Vapnik [29] in 1982. The idea consists of mapping the space of input examples into a high-dimensional feature space, so that the optimal separating hyperplane built on this space allow

a good generalization capacity. The input examples become linearly or almost linearly separable in the high dimensional space through selecting an adequate mapping [28]. Research on SVMs is extensive since it was invented. However, to our best knowledge, there is no effort on learning SVMs on private data. In this paper, our goal is to provide a privacy-preserving algorithm for multi-parties to collaboratively learn SVMs without compromising their data privacy.

3.1 Problem

We consider the scenario where multiple parties P_1, P_2, ..., and P_n, each having a private data set (denoted by D_1, D_2, ... and D_n respectively), want to collaboratively learn SVMs on the concatenation of their data sets. Because they are concerned about their data privacy, neither party is willing to disclose its actual data set to others. Specially, we consider the homogeneous collaboration where each data set contains the same number of attributes but different set of instances. Let m be the total number of attributes in each data set. Let N be the total number of instances, N_1 is the number of instances for P_1, N_2 is the number of instances for P_2, ..., and N_n is the number of instances for P_n. We further assume that the class labels are shared but the instance identifiers and actual attribute values are kept confidential.

3.2 Overview of Support Vector Machine

SVM is primarily a two-class classifier for which the optimization criterion is the width of the margin between the different classes. In the linear form, the formula for output of a SVM is

$$u = \overrightarrow{w} \cdot \overrightarrow{x} + b, \tag{1}$$

where $\overrightarrow{w}$ is the normal vector to the hyperplane and $\overrightarrow{x}$ is the input vector. To maximize margin, we need minimize the following [5]:

$$\min_{\mathrm{w,b}} \frac{1}{2} ||\overrightarrow{w}||^2, \tag{2}$$

subject to $y_i(\overrightarrow{w} \cdot \overrightarrow{x_i} + b) \geq 1, \forall i$, where $\overrightarrow{x_i}$ is the ith training example, and y_i is the correct output of the SVM for the ith training example. The value y_i is $+1$ (resp. -1) for the positive (resp. negative) examples in a class.

Through introducing Lagrangian multipliers, the above optimization can be converted into a dual quadratic optimization problem.

$$\min_{\overrightarrow{\alpha}} \Psi(\overrightarrow{\alpha}) = \min_{\alpha_\mathrm{i}, \alpha_\mathrm{j}} \frac{1}{2} \sum_{\mathrm{i,j=1}}^{\mathrm{N}} \alpha_\mathrm{i} \alpha_\mathrm{j} \mathrm{y_i y_j K}(\overrightarrow{x_i}, \overrightarrow{x_j}) - \sum_{\mathrm{i=1}}^{\mathrm{N}} \alpha_\mathrm{i}, \tag{3}$$

where α_i are the Lagrange multipliers, $\overrightarrow{\alpha} = \alpha_1, \alpha_2, \ldots, \alpha_N$, subject to inequality constraints: $\alpha_i \geq 0, \forall i$, and linear equality constraint: $\sum_{i=1}^{N} y_i \alpha_i = 0$.

By solving the dual optimization problem, one obtains the coefficients α_i, $i = 1, \ldots, N$, from which the normal vector $\overrightarrow{w}$ and the threshold b can be derived [22].

To deal with non-linearly separable data in feature space, Cortes and Vapnik [6] introduced slack-variables to relax the hard-margin constraints. The modification is:

$$\min \frac{1}{2}||\overrightarrow{w}||^2 + \mathrm{C}\sum_{\mathrm{i=1}}^{\mathrm{N}} \xi_{\mathrm{i}} \tag{4}$$

subject to $y_i(\overrightarrow{w} \cdot \overrightarrow{x_i} + b) \geq 1 - \xi_i, \forall i$, where ξ_i is slack variable that allows margin failure and constant $C > 0$ determines the trade-off between the empirical error and the complexity term. This leads to dual quadratic problem involving [3] subject to the constraints $C \geq \alpha_i \geq 0, \forall i$ and $\sum_{i=1}^{N} y_i \alpha_i = 0$.

To solve the dual quadratic problem, we apply sequential minimal optimization [22] which is a very efficient algorithm for training SVMs.

3.3 Sequential Minimal Optimization

Sequential Minimal Optimization (SMO) [22] is a simple algorithm that can efficiently solve the SVM quadratic optimization (QO) problem. Instead of directly tackle the QO problem, it decomposes the overall QO problem into QO sub-problems based on Osunna's convergence theorem [20]. At each step, SMO chooses two Lagarange multipliers to jointly optimize, find the optimal values for these multipliers, and updates the SVM to reflect the new optimal values.

In order to solve for the two Lagrange multipliers, SMO firstly computes the constraints on these multipliers and then solves for the constrained minimum. Normally, the objective function is positive definite, SMO computes the minimum along the direction of the linear constraints $\sum_{i=1}^{2} y_i \alpha_i = 0$ within the boundary $C \geq \alpha_i \geq 0,\ i = 1, 2$.

$$\alpha_2^{\mathrm{new}} = \alpha_2 + \mathrm{y_2}(\mathrm{E_1} - \mathrm{E_2})\,\eta, \tag{5}$$

where $E_i = y_i \alpha_i K(\overrightarrow{x_i}, \overrightarrow{x}) - y_i$ is the error on the ith training example, $\overrightarrow{x_i}$ is the stored training vector and $\overrightarrow{x}$ is the input vector, and η is the second derivative of [3] along the direction of the above linear constraints:

$$\eta = K(\overrightarrow{x_1}, \overrightarrow{x_1}) + K(\overrightarrow{x_2}, \overrightarrow{x_2}) - 2K(\overrightarrow{x_1}, \overrightarrow{x_2}). \tag{6}$$

Next step, the constrained minimum is found by clipping the unconstrained minimum to the ends of the line segment: $\alpha_2^{new,clipped}$ is equal to H if $\alpha_2^{new} \geq H$, is equal to α_2^{new} if $L < \alpha_2^{new} < H$, and is equal to $\alpha_2^{new,clipped} = L$ if $\alpha_2^{new} \leq L$. If the target y_1 is not equal to the target y_2, $L = \max(0, \alpha_2 - \alpha_1)$,

$H = \min(C, C + \alpha_2 - \alpha_1)$. If the target y_1 is equal to the target y_2, $L = \max(0, \alpha_2 + \alpha_1 - C)$, $H = \min(C, \alpha_2 + \alpha_1)$.

The value of α_1 is computed from the new, clipped, α_2:

$$\alpha_1^{\text{new}} = \alpha_1 + \text{s}(\alpha_2 - \alpha_2^{\text{new, clipped}}), \tag{7}$$

where $s = y_1 y_2$.

In the procedure of sequential minimal optimization, the only step accessing the actual attribute values is the computation of the kernel function K. Kernel functions have various forms. Three types of kernel functions are considered: they are the linear kernel function $K = (\overrightarrow{a} \cdot \overrightarrow{b})$, the polynomial kernel function $K = ((\overrightarrow{a} \cdot \overrightarrow{b}) + \theta)^d$, where $d \in N, \theta \in R$ are constants, and the sigmoid kernel function $K = \tanh((\kappa(\overrightarrow{a} \cdot \overrightarrow{b})) + \theta)$, where $\kappa, \theta \in R$ are constants, for instances $\overrightarrow{a}$ and $\overrightarrow{b}$.

To compute these types of kernel functions, one needs to compute the inner product between two instances. If the two instances belong to the same party, this party can compute the inner product by herself; if one instance (e.g., $\overrightarrow{x_1}$) belongs to one party (e.g., P_1), and the other instance (e.g., $\overrightarrow{x_2}$) belongs to another party (e.g., P_2), then P_1 can compute $(\overrightarrow{x_1} \cdot \overrightarrow{x_1})$ and P_2 can compute $(\overrightarrow{x_2} \cdot \overrightarrow{x_2})$. However, to compute $(\overrightarrow{x_1} \cdot \overrightarrow{x_2})$, different parties have to collaborate. How to conduct this inner product computation across parties without compromising each party's data privacy presents a great challenge. In next section, secure protocols are developed to tackle this challenge.

4 Protocols

To securely compute the inner product between two different parties, we present secure protocols against both semi-honest and malicious party models defined as follows.

Semi-honest Party Model: In this model, all the parties are assumed to use their actual vectors as the inputs and exactly follow the predefined steps in the protocol.

Malicious Party Model: In this model, we categorize the malicious behaviors into two cases: (I) The malicious actions of one party *does not* intend to gain any useful information from the other party; (II) The purpose of the malicious actions of one party *does* intend to gain useful information from the other party. The first category usually contains: (I_1) refusing to participate in the protocol; (I_2) prematurely aborting the protocol; (I_3) substituting an input with an arbitrary value; (I_4) providing more numbers or less numbers than necessary. For example, assume P_2 should receive m numbers for further computation, but P_1 sends m_1 numbers of encrypted terms with $m_1 \neq m$. The second category contains: (II_1) substituting an input vector with a purposely chosen vector; for example, P_1 wants to know the ith element, x_{2i}, of P_2's

instance $\overrightarrow{x_2}$, she changes her vector $\overrightarrow{x_1}$ to $\overrightarrow{x_1}'$ by setting the jth element $x_{1j} = 0$ for all $j \in [1, m]$, $j \neq i$, and the ith element x_{1i} be any value ϖ except 0. As we will discuss, after protocol 1, P_1 obtains $\varpi \times x_{2i}$. She can easily know x_{2i} by computing the following: $(\varpi \times x_{2i}) \div \varpi$. If she wants to know $\sum_{i=1}^{m} x_{2i}$, she may set $x_{1j} = 1$ for all $j \in [1, m]$. Since $\overrightarrow{x_1} \cdot \overrightarrow{x_2} = \sum_{i=1}^{m} x_{2i}$, after Protocol 1, she obtains $\sum_{i=1}^{m} x_{2i}$.

In the following, we firstly discuss secure protocol against semi-honest parties, and then provide a secure protocol against malicious parties.

4.1 Introducing Homomorphic Encryption

The concept of homomorphic encryption was originally proposed in [23]. Since then, many such systems have been proposed [4, 18, 19, 21]. We observe that some homomorphic encryption schemes, such as [8], are not robust against chosen plaintext attacks. However, we base our secure protocols on [21], which is semantically secure [11].

In our secure protocols, we use additive homomorphism offered by [21]. In particular, we utilize the following characterizer of the homomorphic encryption functions: $e(a_1) \times e(a_2) = e(a_1 + a_2)$ where e is an encryption function; a_1 and a_2 are the data to be encrypted. Because of the property of associativity, $e(a_1 + a_2 + \cdots + a_n)$ can be computed as $e(a_1) \times e(a_2) \times \cdots \times e(a_n)$ where $e(a_i) \neq 0$. That is

$$d(e(a_1 + a_2 + \cdots + a_n)) = d(e(a_1) \times e(a_2) \times \cdots \times e(a_n)) \tag{8}$$

$$d(e(a_1)^{a_2}) = d(e(a_1 a_2)) \tag{9}$$

4.2 Secure Protocol Against Semi-Honest Parties

Let's assume that P_1 has an instance vector $\overrightarrow{x_1}$ and P_2 has an instance vector $\overrightarrow{x_2}$. Both vectors have m elements. We use x_{1i} to denote the ith element in vector $\overrightarrow{x_1}$, and x_{2i} to denote the ith element in vector $\overrightarrow{x_2}$. In order to compute the $K(\overrightarrow{x_1}, \overrightarrow{x_2})$, the key issue is how P_1 and P_2 compute the inner product between $\overrightarrow{x_1}$ and $\overrightarrow{x_2}$ without disclosing them to each other. In our secure protocol, P_1 adds a random number to each of her actual data values, encrypts the masked values, and sends the encrypted masked terms to P_2. By adding the random numbers, P_2 is prevented from guessing P_1's actual values based on encryption patterns. Firstly, one of parties is randomly chosen as a key generator. For simplicity, let's assume P_1 is selected as the key generator. P_1 generates a cryptographic key pair (d, e) of a semantically-secure homomorphic encryption scheme and publishes its public key e. P_1 applies the encryption key to each element of x_1 (e.g., $e(x_{1i} + r_i)$). P_2 computes $e(\overrightarrow{x_1} \cdot \overrightarrow{x_2})$. He then sends $e(\overrightarrow{x_1} \cdot \overrightarrow{x_2})$ to P_1 who decrypts it and gets $(\overrightarrow{x_1} \cdot \overrightarrow{x_2})$.

We describe this more formally as

Protocol 1. *INPUT: P_1's input is a vector $\overrightarrow{x_1} = \{x_{11}, x_{12}, \ldots, x_{1m}\}$, and P_2's input is a vector $\overrightarrow{x_2} = \{x_{21}, x_{22}, \ldots, x_{2m}\}$. The elements in the input vectors are taken from the real number domain.*

1. *P_1 performs the following operations:*
   a) *She computes $e(x_{1i}+r_i)s$ ($i \in [1,m]$) and sends them to P_2. r_i, known only by P_1, is a random number in real domain.*
   b) *She computes $e(-r_i)s$ ($i \in [1,m]$) and sends them to P_2.*
2. *P_2 performs the following operations:*
   a) *He computes $t_1 = e(x_{11}+r_1)^{x_{21}} = e(x_{11}\ldots x_{21}+r_1x_{21})$, $t_2 = e(x_{12}+r_2)^{x_{22}} = e(x_{12}\ldots x_{22}+r_2x_{22}), \ldots, t_m = e(x_{1m})^{x_{2m}} = e(x_{1m}\ldots x_{2m}+r_mx_{2m})$.*
   b) *He computes $t_1 \times t_2 \times \cdots \times t_m = e(x_{11}\cdot x_{21} + x_{12}\cdot x_{22} + \cdots + x_{1m}\cdot x_{2m} + r_1x_{21} + r_2x_{22} + \cdots + r_mx_{2m}) = e(\overrightarrow{x_1}\cdot\overrightarrow{x_2} + \sum_{i=1}^{m} r_ix_{2i})$.*
   c) *He computes $e(-r_i)^{x_{2i}} = e(-r_ix_{2i})$ for $i \in [1,m]$.*
   d) *He computes $e(\overrightarrow{x_1}\cdot\overrightarrow{x_2} + \sum_{i=1}^{m} r_ix_{2i}) \times e(-r_1x_{21}) \times e(-r_2x_{22}) \times \cdots \times e(-r_mx_{2m}) = e(\overrightarrow{x_1}\cdot\overrightarrow{x_2})$.*

We need to show that the above protocol is correct, and that it preserves the privacy of P_1 and P_2 as postulated in Sect. 3.1.

Lemma 1. *(Correctness). Protocol 1 correctly computes the inner product $(\overrightarrow{x_1}\cdot\overrightarrow{x_2})$ against semi-honest parties.*

Proof. When P_2 receives each encrypted element $e(x_{1i}+r_i)$ and $e(-r_i)$, he computes $\sum_{i=1}^{m} e(x_{1i}+r_i)^{x_{2i}}$ which, according to (9), is equal to $e(\sum_{i=1}^{m} x_{1i}\cdot x_{2i} + \sum_{i=1}^{m} r_ix_{2i})$ for all $i \in [1,m]$. He then computes $e(\overrightarrow{x_1}\cdot\overrightarrow{x_2} + \sum_{i=1}^{m} r_ix_{2i}) \times e(-r_1x_{21}) \times e(-r_2x_{22}) \times \cdots \times e(-r_mx_{2m}) = e(\overrightarrow{x_1}\cdot\overrightarrow{x_2})$ according to (8). After that, he sends it to P_1 who computes $d(e(\overrightarrow{x_1}\cdot\overrightarrow{x_2})) = (\overrightarrow{x_1}\cdot\overrightarrow{x_2})$. Therefore, $(\overrightarrow{x_1}\cdot\overrightarrow{x_2})$ is correctly computed.

Lemma 2. *(Privacy-Preserving). Assuming the parties follow the protocol, the private data are securely protected.*

Proof. There are two points we need analyze. (1) Whether P_1 can obtain P_2's private data. There is no information that P_2 sends to P_1, thus, P_2's private data cannot be disclosed to P_1. (2) Whether P_2 can obtain P_1's private data. What P_2 receives from P_1 is encrypted and masked element of P_1's data. Since P_2 has no decryption key and doesn't know the random number used by P_1, it is impossible that P_2 can obtain P_1's private data.

Lemma 3. *(Efficiency). Protocol 1 is efficient from both computation and communication point of view.*

Proof. To prove the efficiency, we need conduct complexity analysis of the protocol. The bit-wise communication cost of this protocol is $(2m+1)\alpha$ where α is the number of bits for each transmitted element. The following contributes to the computational cost: (1) 2m encryptions; (2) 2m exponentiations; (2) 2m−1 multiplications. Therefore, the protocol is sufficient fast.

4.3 Privacy Against Malicious Parties

In the last section, we provide the secure protocol for semi-honest parties. In practice, providing a protocol against malicious parties is demanding. We cannot hope to stop all the attacks I_1, I_2, and I_3. I_4 can be easily detected by counting the number of terms received and comparing with the legal number of terms. In this section, we will mainly deal with the second category. We also think the attack in this category (i.e., II_1) is critical from privacy protection point of view. To deal with II_1, the collaborative parties need to predefine the malicious pattern for their input vectors,[2] they then detect during the protocol whether each other's input is legal by the following protocol:

Protocol 2. *(Oblivious K Computation)*

1. *P_2 performs the following:*
   a) *P_2 generates a vector $\overrightarrow{x_2}'$ with m elements. (Note that $\overrightarrow{x_2}'$ cannot be a malicious vector predefined by the collaborative parties.)*
   b) *After P_2 receives the encrypted terms from P_1, i.e., $e(x_{1i})$ for all $i \in [1, m]$, he uses x_2 to conduct Step 2 (protocol 1) except that he doesn't send $e(\overrightarrow{x_1} \cdot \overrightarrow{x_2})$ to P_1; he then uses x_2' to conduct Step 2 (Protocol 1) except that he doesn't send $e(\overrightarrow{x_1} \cdot \overrightarrow{x_2}')$ to P_1.*
   c) *P_2 flips a fair coin to decide the order in which $e(\overrightarrow{x_1} \cdot \overrightarrow{x_2})$ and $e(\overrightarrow{x_1} \cdot \overrightarrow{x_2}')$ are to be sent to P_1, e.g. if it is heads, he firstly sends $e(\overrightarrow{x_1} \cdot \overrightarrow{x_2})$, and then $e(\overrightarrow{x_1} \cdot \overrightarrow{x_2}')$ to P_1; if it is tails, he firstly sends $e(\overrightarrow{x_1} \cdot \overrightarrow{x_2}')$, and then $e(\overrightarrow{x_1} \cdot \overrightarrow{x_2})$.*

2. *P_1 performs the following:*
   a) *P_1 decrypts the $e(\overrightarrow{x_1} \cdot \overrightarrow{x_2})$ and $e(\overrightarrow{x_1} \cdot \overrightarrow{x_2}')$ and obtains $\overrightarrow{x_1} \cdot \overrightarrow{x_2}$ and $\overrightarrow{x_1} \cdot \overrightarrow{x_2}'$. She checks whether P_2 uses a malicious input vector (a vector from a closed list of predefined, special vector values). If she detects that P_2 uses a malicious input vector, she doesn't send $\overrightarrow{x_1} \cdot \overrightarrow{x_2}$ and $\overrightarrow{x_1} \cdot \overrightarrow{x_2}'$ to P_2 and halts the protocol. Otherwise, she sends P_2 $\overrightarrow{x_1} \cdot \overrightarrow{x_2}$ and $\overrightarrow{x_1} \cdot \overrightarrow{x_2}'$ in the same order P_2 used.*
3. *P_2 checks whether P_1 uses a malicious input vector. If yes, he halts the protocol without telling P_1 which of the two vectors is $\overrightarrow{x_1} \cdot \overrightarrow{x_2}$, and which one is $\overrightarrow{x_1} \cdot \overrightarrow{x_2}'$. Otherwise, he sends $\overrightarrow{x_1} \cdot \overrightarrow{x_2}$ to P_1 (he can distinguish between $\overrightarrow{x_1} \cdot \overrightarrow{x_2}$ and $\overrightarrow{x_1} \cdot \overrightarrow{x_2}'$ based on the order he received from P_1.)*

We now discuss that this protocol preserves the privacy in the presence of the attack of type II_1. Since P_1 is the first to obtain the results, i.e. $\overrightarrow{x_1} \cdot \overrightarrow{x_2}$ and $\overrightarrow{x_1} \cdot \overrightarrow{x_2}'$, she checks whether P_2 is using a malicious input vector. If she finds that he does, she keeps the final results and halts the protocol. Therefore, P_1's privacy is preserved since P_2 does not obtain the decrypted results. On the other hand, since P_2 has the other pseudo vector x_2', P_1 can't distinguish which of the two vectors gives correct result; she has to send the two results e.g.,

[2]These patterns should be detectable through the targeted computation results.

$\overrightarrow{x_1} \cdot \overrightarrow{x_2}$ and $\overrightarrow{x_1} \cdot \overrightarrow{x_2}'$, back to P_2 in order to get the right one. After he receives $\overrightarrow{x_1} \cdot \overrightarrow{x_2}$ and $\overrightarrow{x_1} \cdot \overrightarrow{x_2}'$ from P_1, P_2 checks whether P_1 used a malicious input vector. If he finds that she does, he will not let her know the correct result. In this case, the probability that she correctly guesses P_2's information is $\frac{1}{2}$. To decrease this probability, P_2 can generate more pseudo-random vectors. The computation overhead is $O(\omega m)$, and the communication overhead will be $2\beta(m + 1 + \omega)$ where ω is number of pseudo-random vectors generated. Furthermore, we could let P_1 and P_2 take turns as the key generator. When computing the first half of their vector inner product, P_1 is selected as the key generator; when computing the second half of the vector inner product, P_2 is selected as the key generator.

The two protocols presented in this section compute the real domain inner product between two instances belonging to different parties, without revealing the actual data of one of the parties to the other party. Once the inner product is obtained, the kernel function can be easily computed. After the necessary kernel functions are computed, training SVMs can be done following SMO procedure.

5 Conclusion

In this paper, we consider the problem of collaboratively learning Support Vector Machines, by using linear, polynomial or sigmoid kernel functions, on private data. We develop a secure collaborative protocol based on semantically secure homomorphic encryption scheme. In our protocol, the parties do not need to send all their data to a central, trusted party. Instead, we use the homomorphic encryption and random perturbation techniques to conduct the computations across the parties without compromising their data privacy. We develop two secure protocols to deal with semi-honest and malicious models respectively. Privacy analysis is provided. Correctness of our protocols is shown and complexity of the protocols is addressed as well. As future work, we will develop secure protocols for the cases where other kernel functions are applied. We will also apply our technique to other data mining computations, such as secure collaborative clustering.

References

1. G. Aggarwal, N. Mishra, and B. Pinkas. Secure computation of the k th-ranked element. In *EUROCRYPT pp 40–55*, 2004.
2. R. Agrawal, A. Evfimievski, and R. Srikant. Information sharing across private databases. In *Proceedings of ACM SIGMOD ICMD, pp 86–97*, 2003.
3. R. Agrawal and R. Srikant. Privacy-preserving data mining. In *Proceedings of the ACM SIGMOD Conference on Management of Data, pp 439–450*, ACM, May 2000.

4. J. Benaloh. Dense probabilistic encryption. In *Proceedings of the Workshop on Selected Areas of Cryptography, pp 120–128*. Kingston, Ontario, May, 1994.
5. Christopher J.C. Burges. A tutorial on support vector machines for pattern recognition. In *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.
6. C. Cortes and V. Vapnik. Support-vector networks. In *Machine Learning*, 20(3):273–297, 1995.
7. J. Shawe-Taylor and N. Cristianini. An introduction to support vector machines. In *Cambridge University Press*.
8. J. Domingo-Ferrer. A provably secure additive and multiplicative privacy homomorphism. In *Information Security Conference, pp 471–483, 2002*.
9. C. Dwork and K. Nissim. Privacy-preserving datamining on vertically partitioned databases.
10. M. Freedman, K. Nissim, and B. Pinkas. Effiicent private matching and set intersection. In *EUROCRYPT, pp 1–19*, 2004.
11. B. Goethals, S. Laur, H. Lipmaa, and T. Mielikainen. On secure scalar product computation for privacy-preserving data mining. In *Proceedings of the 7th Annual International Conference in Information Security and Cryptology (ICISC 2004), volume 3506 of Lecture Notes in Computer Science, pp 104–120*, Seoul, Korea, December 2–3, 2004, Springer, Berlin Heidelberg New York, 2004.
12. O. Goldreich. Secure multi-party computation (working draft). http://www.wisdom.weizmann.ac.il /home/oded/public_html/foc.html, 1998.
13. T. Joachims. Text categorization with support vector machines: learning with many relevant features. In *Proceedings of ECML-98, 10th European Conference on Machine Learning, C. Nédellec and C. Rouveirol, eds., no 1398, pp 137–142*, Chemnitz, DE, 1998. Springer, Berlin Heidelberg New York, DE.
14. J. Vaidya and C.W. Clifton. Privacy preserving association rule mining in vertically partitioned data. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, July 23–26, 2002, Edmonton, Alberta, Canada.
15. M. Kantarcioglu and C. Clifton. Privacy preserving data mining of association rules on horizontally partitioned data. In *Transactions on Knowledge and Data Engineering, IEEE Computer Society Press*, Los Alamitos, CA.
16. Y. LeCun, L. Botou, L. Jackel, H. Drucker, C. Cortes, J. Denker, I. Guyon, U. Muller, E. Sackinger, P. Simard, and V. Vapnik. Learning algorithms for classification: a comparison on handwritten digit recognition, 1995.
17. Y. Lindell and B. Pinkas. Privacy preserving data mining. In *Advances in Cryptology – Crypto2000, Lecture Notes in Computer Science*, volume 1880, 2000.
18. D. Naccache and J. Stern. A new public key cryptosystem based on higher residues. In *Proceedings of the 5th ACM Conference on Computer and Communication Security, pp 59–66*, San Francisco, California, United States, 1998.
19. T. Okamoto and S. Uchiyama. A new public-key cryptosystem as secure as factoring. In *Eurocrypt'98, LNCS 1403, pp 308–318*, 1998.
20. R. Freund, F. Girosi, and E. Osuna. Training support vector machines: an application to face detection. In *Proceedings of Computer Vision and Pattern Recognition, pp 130–136*.
21. P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Advances in Cryptography – EUROCRYPT '99, pp 223–238*, Prague, Czech Republic, May 1999.

22. J. Platt. Sequetial minimal optimization: a fast algorithm for training support vector machines. In *Technical Report MST-TR-98-14, Microsoft Research, 1998.*
23. R. Rivest, L. Adleman, and M. Dertouzos. On data banks and privacy homomorphisms. In *Foundations of Secure Computation, eds. R. A. DeMillo et al., pp 169–179*, Academic Press, 1978.
24. K.-R. Mller, B. Schlkopf, and A.J. Smola. Nonlinear component analysis as a kernel eigenvalue problem. In *Neural Computation*, 10, 1299–1319.
25. Smola A. (Eds.) Schlkopf B., and Burges C. Advances in kernel methods – support vector learning. MIT.
26. L. Sweeney. k-anonymity: a model for protecting privacy. In *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10(5), 557–570, 2002.
27. J. Vaidya and C. Clifton. Secure set intersection cardinality with application to association rule mining. In *Journal of Computer Security*, IOS, to appear.
28. V. Vapnik. The nature of statistical learning theory. Springer, Berlin Heidelberg New York, 1995.
29. V. N. Vapnik. Estimation of dependences based on empirical data. Springer, Berlin Heidelberg New York, 1982, 22.
30. R. Wright and Z. Yang. Privacy-preserving bayesian network structure computation on distributed heterogeneous data. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2004.
31. A. C. Yao. Protocols for secure computations. In *Proceedings of the 23rd Annual IEEE Symposium on Foundations of Computer Science*, 1982.
32. J. Zhan, L. Chang, and S. Mawin. How to prevent private data from being disclosed to a malicious attacker. In *IEEE International Workshop on Foundations of Semantic Oriented Data and Web Mining*, Houston, Texas, USA, November 27–30, 2005.
33. Z. Zhan, S. Matwin, and L. Chang. Building support vector machines on private data. In *International Conference on Artificial Intelligence, to appear*, 2005.

Privacy-Preserving Naive Bayesian Classification over Horizontally Partitioned Data

Justin Zhan[1], Stan Matwin[2], and LiWu Chang[3]

[1] Carnegie Mellon University
justinzh@andrew.cmu.edu
[2] University of Ottawa
stan@site.uottawa.ca
[3] Naval Research Laboratory
lchang@itd.nrl.navy.mil

1 Introduction

Recent advances[1] in computer networking and database technologies have resulted in creation of large quantities of data which are located in different sites. Data mining is a useful tool to extract valuable knowledge from this data. Well known data mining algorithms include association rule mining, classification, clustering, outlier detection, etc. However, extracting useful knowledge from distributed sites is often challenging due to real world constraints such as privacy, communication and computation overhead.

In this paper, we focus on privacy-preserving data mining in a distributed setting where the different sites have diverse sets of records. Specially, we consider the problem of privacy-preserving naive Bayesian classification that is one of the most successful algorithms in many classification domains. A Bayesian network is a high-level representation of a probability distribution over a set of variables that are used for constructing a model of the problem domain. It has been widely used in sales decision making, marketing systems, risk analysis, cost benefit factor inference in E-services, and other business applications. A Bayesian network have many applications. For instance it can be used to compute the predictive distribution on effects of possible actions since it is a model of the problem domain probability distribution.

A naive Bayesian classifier is one of Bayesian classifiers under conditional independence assumption of different features. Over the last decade, the naive Bayesian classification has been widely utilized. Although the techniques that have been developed are effective, new techniques dealing with naive Bayesian classification over private data are required. In other words, we need methods

[1] The preliminary version of this paper has been published [22].

J. Zhan et al.: *Privacy-Preserving Naive Bayesian Classification over Horizontally Partitioned Data*, Studies in Computational Intelligence (SCI) **118**, 529–538 (2008)
www.springerlink.com

to learn a naive Bayesian classifier over distributed private data. In this paper, we develop a new scheme based on homomorphic encryption without compromising data privacy.

2 Related Work

In early work on privacy-preserving data mining, Lindell and Pinkas [13] propose a solution to privacy-preserving classification problem using oblivious transfer protocol, a powerful tool developed by secure multi-party computation (SMC) research [10, 21]. The techniques based on SMC for efficiently dealing with large data sets have been addressed in [20]. Randomization approaches were firstly proposed by Agrawal and Srikant in [2] to solve privacy-preserving data mining problem. Researchers proposed more random perturbation-based techniques to tackle the problems (e.g., [5, 7, 18]). In addition to perturbation, aggregation of data values [19] provides another alternative to mask the actual data values. In [1], authors studied the problem of computing the kth-ranked element. Dwork and Nissim [6] showed how to learn certain types of boolean functions from statistical databases in terms of a measure of probability difference with respect to probabilistic implication, where data are perturbed with noise for the release of statistics. The problem we are studying is actually a special case of a more general problem, the Secure Multi-party Computation (SMC) problem. Briefly, a SMC problem deals with computing any function on any input, in a distributed network where each participant holds one of the inputs, while ensuring that no more information is revealed to a participant in the computation than can be inferred from that participant's input and output [12]. The SMC problem literature is extensive, having been introduced by Yao [21] and expanded by Goldreich, Micali, and Wigderson [11] and others [8]. It has been proved that for any function, there is a secure multi-party computation solution [10]. The approach used is as follows: the function F to be computed is first represented as a combinatorial circuit, and then the parties run a short protocol for every gate in the circuit. Every participant gets corresponding shares of the input wires and the output wires for every gate. This approach, though appealing in its generality and simplicity, means that the size of the protocol depends on the size of the circuit, which depends on the size of the input. This is highly inefficient for large inputs, as in data mining. It has been well accepted that for special cases of computations, special solutions should be developed for efficiency reasons.

3 Building Naive Bayesian Classifiers

3.1 Notations

- e: public key.
- d: private key.
- P_i: the ith party.

- n: the total number of parties. Assuming $n > 2$.
- m: the total number of class.
- τ: the total number attribute.
- α is the number of bits for each transmitted element in the privacy-preserving protocols.
- N: the total number of records.

3.2 Cryptography Tools

Our scheme is based on homomorphic encryption which was originally proposed in [17]. Since then, many such systems have been proposed [3, 14–16]. We observe that some homomorphic encryption schemes, such as [4], are not robust against chosen cleartext attacks. However, we base our secure protocols on [16], which is semantically secure [9].

In our secure protocols, we use additive homomorphism offered by [16]. In particular, we utilize the following characterizer of the homomorphic encryption functions: $e(a_1) \times e(a_2) = e(a_1 + a_2)$ where e is an encryption function; a_1 and a_2 are the data to be encrypted. Because of the property of associativity, $e(a_1 + a_2 + \cdots + a_n)$ can be computed as $e(a_1) \times e(a_2) \times \cdots \times e(a_n)$ where $e(a_i) \neq 0$. That is

$$d(e(a_1 + a_2 + \cdots + a_n)) = d(e(a_1) \times e(a_2) \times \cdots \times e(a_n)) \tag{1}$$

$$d(e(a_1)^{a_2}) = d(e(a_1 a_2)) \tag{2}$$

3.3 Introducing Naive Bayesian Classification

The naive Bayesian classification is one of the most successful algorithms in many classification domains. Despite of its simplicity, it is shown to be competitive with other complex approaches, especially in text categorization and content based filtering. The naive Bayesian classifier applies to learning tasks where each instance x is described by a conjunction of attribute values and where the target function f(x) can take on any value from some finite set V. A set of training examples of the target function is provided, and a new instance is presented, described by the tuple of attribute values $< a_1, a_2, \ldots, a_n >$. The learner is asked to predict the target value for this new instance. Under a conditional independence assumption, i.e., $Pr(a_1, a_2, \ldots, a_n | v_j) = \prod_{i=1}^{n} Pr(a_i | v_j)$, a naive Bayesian classifier can be derived as follows:

$$V_{NB} = argmax_{v_j \in V} Pr(v_j) \prod_{i=1}^{\tau} Pr(a_i | v_j) \tag{3}$$

In this paper, we will design a privacy-preserving system to show how to compute a naive Bayesian classifier. The goal of our privacy-preserving

classification system is to disclose no private data in every step. We firstly select a key generator who produces the encryption and decryption key pairs. The computation of the whole system is under encryption. For the purpose of illustration, let's assume that P_n is the key generator who generates a homomorphic encryption key pair (e, d). To build a NB classifier, we need conduct the following major steps: (1) To compute $e(\prod_{i=1}^{n} Pr(a_i|v_j))$; (2) To compute $e(Pr(v_j))$; (3) To compute $argmax_{v_j \in V} Pr(v_j) \prod_{i=1}^{n} Pr(a_i|v_j)$. Next, we will show how to conduct each step.

3.4 Privacy-Preserving Naive Bayesian Classification

To Compute $e(\prod_{i=1}^{\tau} Pr(a_i|v_j))$

Protocol 1.

1. *P_n computes $e(\prod_{a_i \in P_n} Pr(a_i|v_j))$ denoted by $e(G_n)$ and sends it to P_1.*
2. *P_1 computes $e(G_n)^{G_1} = e(G_1 G_n)$ where $G_1 = \prod_{a_i \in P_1} Pr(a_i|v_j)$, then sends $e(G_1 G_n)$ to P_2.*
3. *P_2 computes $e(G_1 G_n)^{G_2} = e(G_1 G_2 G_n)$ where $G_2 = \prod_{a_i \in P_2} Pr(a_i|v_j)$, then sends $e(G_1 G_2 G_n)$ to P_3.*
4. *Continue until P_{n-1} obtains $e(G_1 G_2 \cdots G_n) = e(\prod_{i=1}^{\tau} Pr(a_i|v_j))$.*

Theorem 1. *(Correctness). Protocol 1 correctly computes $e(\prod_{i=1}^{\tau} Pr(a_i|v_j))$.*

Proof. When P_1 receives $e(G_n)$, he computes $e(G_n)^{G_1}$ which is equal to $e(G_1 G_n)$ according to (2). He sends it to P_2 who computes $e(G_1 G_n)^{G_2}$ which is equal to $e(G_1 G_2 G_n)$ according to (2). Continuing to send the result to the next party. Finally, P_{n-1} obtains $e(G_1 G_2 \cdots G_n) = e(\prod_{i=1}^{\tau} Pr(a_i|v_j))$. Therefore, Protocol 1 correctly computes $e(\prod_{i=1}^{\tau} Pr(a_i|v_j))$.

Theorem 2. *(Privacy-Preserving). Assuming the parties follow the protocol, the private data are securely protected.*

Proof. In Protocol 1, all the data transmission are hidden under encryption. The parties who are not the key generator can't see other parties' private data. On the other hand, the key generator doesn't obtain the encryption of other parties' private data. Therefore, Protocol 1 discloses no private data.

Theorem 3. *(Efficiency). Protocol 1 is efficient in terms of computation and communication complexity.*

Proof. To prove the efficiency, we need conduct complexity analysis of the protocol. The bit-wise communication cost of this protocol is $\alpha(n-1)$. The computation cost is upper bounded by $N + nt$. Therefore, the protocol is sufficient fast.

To Compute $e(Pr(v_j))$

Protocol 2.

1. *Each party computes the share $Pr(v_j)$ for their own class label set. Let assume that P_1 has the share s_1, P_2 has the share $s_2, \ldots, P_n$ has the share s_n. Our goal is to compute $\sum_{i=1}^{n} s_i$.*
2. *P_n computes $e(s_n)$ and sends it to P_1.*
3. *P_1 computes $e(s_n) \times e(s_1) = e(s_1 + s_n)$, then sends it to P_2.*
4. *P_2 computes $e(s_1 + s_n) \times e(s_2) = e(s_1 + s_2 + s_n)$.*
5. *Repeat until P_{n-1} obtains $e(\sum_{i=1}^{n} s_i) = e(Pr(v_j))$.*

Theorem 4. *(Correctness). Protocol 2 correctly computes $e(Pr(v_j))$.*

Proof. When P_1 receives $e(s_n)$, he computes $e(s_n) \times e(s_1)$ which is equal to $e(s_1 + s_n)$ according to (2). He sends it to P_2 who computes $e(s_1 + s_n) \times e(s_2)$ which is equal to $e(s_1 + s_2 + s_n)$ according to (2). Continuing to send the result to the next party. Finally, P_{n-1} obtains $e(s_1 + s_2 \cdots s_n) = e(Pr(v_j))$. Therefore, Protocol 2 correctly computes $e(Pr(v_j))$.

Theorem 5. *(Privacy-Preserving). Assuming the parties follow the protocol, the private data are securely protected.*

Proof. In Protocol 2, all the data transmission are hidden under encryption. The parties who are not the key generator can't see other parties' private data. On the other hand, the key generator doesn't obtain the encryption of other parties' private data. Therefore, Protocol 2 discloses no private data.

Theorem 6. *(Efficiency). Protocol 2 is efficient in terms of computation and communication complexity.*

Proof. To prove the efficiency, we need conduct complexity analysis of the protocol. The bit-wise communication cost of this protocol is $\alpha(n-1)$. The computation cost is upper bounded by $N + n$. Therefore, the protocol is sufficient fast.

To Compute $e(Pr(v_j) \prod_{i=1}^{\tau} Pr(a_i|v_j))$

Protocol 3.

1. *P_{n-1} generates a set of random numbers: $r_1, r_2, \ldots, r_t$. He then sends $e(Pr(v_j))$, $e(r_1), \ldots, r_t$ to P_n in a random order.*
2. *P_n decrypts each element in the sequence, then sends them to P_1 in the same order as P_{n-1} did.*
3. *P_{n-1} sends $e(\prod_{i=1}^{\tau} Pr(Pr(a_i, v_j)))$ to P_1.*
4. *P_1 computes $e(\prod_{i=1}^{\tau} Pr(Pr(a_i, v_j)))^{Pr(v_j)}$, $e(\prod_{i=1}^{\tau} Pr(Pr(a_i, v_j)))^{r_1}, \ldots,$ $e(\prod_{i=1}^{\tau} Pr(Pr(a_i, v_j)))^{r_t}$. P_1 then sends them to P_{n-1}.*
5. *P_{n-1} obtains $e(Pr(v_j) \prod_{i=1}^{\tau} Pr(a_i|v_j))$.*

Theorem 7. *(Correctness). Protocol 3 correctly computes* $e(Pr(v_j)\prod_{i=1}^{\tau} Pr(a_i|v_j))$.

Proof. In step 4, P_1 computes $e(\prod_{i=1}^{\tau} Pr(Pr(a_i, v_j)))^{Pr(v_j)}$, $e(\prod_{i=1}^{\tau} Pr(Pr(a_i, v_j)))^{r_1}, \ldots, e(\prod_{i=1}^{\tau} Pr(Pr(a_i, v_j)))^{r_t}$. They are equal to $e(Pr(v_j)\prod_{i=1}^{\tau} Pr(Pr(a_i, v_j)))$, $e(r_1\prod_{i=1}^{\tau} Pr(Pr(a_i, v_j))), \ldots, e(r_t\prod_{i=1}^{\tau} Pr(Pr(a_i, v_j)))$ respectively according to (2). In step 5, P_{n-1} gets $e(Pr(v_j)\prod_{i=1}^{\tau} Pr(Pr(a_i, v_j)))$ since he knows the permutations.

Theorem 8. *(Privacy-Preserving). Assuming the parties follow the protocol, the private data are securely protected.*

Proof. In Protocol 3, all the data transmission, among the parties who have no decryption key, are hidden under encryption. Therefore, these parties cannot know the private data. In step 2, P_n obtains the sequence of $Pr(v_j), r_1, \ldots, r_t$. Since it is in a random order, P_n cannot identify $Pr(v_j)$.

Theorem 9. *(Efficiency). Protocol 3 is efficient in terms of computation and communication complexity.*

Proof. The bit-wise communication cost of this protocol is upper bounded by $\alpha(3t + 4)$. The computation cost is upper bounded by $5t$. Therefore, the protocol is sufficient fast.

Through the above protocol, $e(Pr(v_j)\prod_{i=1}^{n} Pr(a_i|v_j))$ can be computed for each $v_j \in V$. Without loss of generality, let's assume P_1 gets $e(V_{NB_1}), e(V_{NB_2}), \ldots, e(V_{NB_k})$ The goal is to find the largest one.

To Compute V_{NB}

Protocol 4.

1. P_1 *computes* $e(V_{NB_i}) \times e(V_{NB_j})^{-1} = e(V_{NB_i} - V_{NB_j})$ *for all* $i, j \in [1, k], i > j$, *and sends the sequence denoted by* φ *to* P_n *in a random order.*
2. P_n *decrypts each element in the sequence* φ. *He assigns the element* $+1$ *if the result of decryption is not less than* 0, *and* -1, *otherwise. Finally, he obtains a* $+1/-1$ *sequence denoted by* φ'.
3. P_n *sends* $+1/-1$ *sequence* φ' *to* P_1 *who computes the largest element.*

Theorem 10. *(Correctness). Protocol 4 correctly computes* V_{NB}.

Proof. P_1 is able to remove permutation effects from φ' (the resultant sequence is denoted by φ'') since she has the permutation function that she used to permute φ, so that the elements in φ and φ'' have the same order. It means that if the qth position in sequence φ denotes $e(V_{NB_i} - V_{NB_j})$, then the qth position in sequence φ'' denotes the evaluation results of V_{NB_i}-V_{NB_j}. We encode it as $+1$ if $V_{NB_i} \geq V_{NB_j}$, and as -1 otherwise. P_1 has two sequences:

Table 1.

	V_{NB_1}	V_{NB_2}	V_{NB_3}	$\cdots$	V_{NB_k}
V_{NB_1}	$+1$	$+1$	-1	$\cdots$	-1
V_{NB_2}	-1	$+1$	-1	$\cdots$	-1
V_{NB_3}	$+1$	$+1$	$+1$	$\cdots$	$+1$
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
V_{NB_k}	$+1$	$+1$	-1	$\cdots$	$+1$

Table 2.

	S_1	S_2	S_3	S_4	Weight
S_1	$+1$	-1	-1	-1	-2
S_2	$+1$	$+1$	-1	$+1$	$+2$
S_3	$+1$	$+1$	$+1$	$+1$	$+4$
S_4	$+1$	-1	-1	$+1$	0

one is the φ, the sequence of $e(V_{NB_i} - V_{NB_j})$, for $i, j \in [1, k](i > j)$, and the other is φ'', the sequence of $+1/-1$. The two sequences have the same number of elements. P_1 knows whether or not V_{NB_i} is larger than V_{NB_j} by checking the corresponding value in the φ'' sequence. For example, if the first element φ'' is -1, P_1 concludes $V_{NB_i} < V_{NB_j}$. P_1 examines the two sequences and constructs the index table (Table 1) to compute the largest element.

In Table 1, $+1$ in entry ij indicates that the information gain of the row (e.g., V_{NB_i} of the ith row) is not less than the information gain of a column (e.g., V_{NB_j} of the jth column); -1, otherwise. P_1 sums the index values of each row and uses this number as the weight of the information gain in that row. She then selects the one that corresponds to the largest weight.

To make it clearer, let's illustrate it by an example. Assume that: (1) there are four information gains with $V_{NB_1} < V_{NB_4} < V_{NB_2} < V_{NB_3}$; (2) the sequence φ is $[e(V_{NB_1} - V_{NB_2}), e(V_{NB_1} - V_{NB_3}), e(V_{NB_1} - V_{NB_4}), e(V_{NB_2} - V_{NB_3}), e(V_{NB_2} - V_{NB_4}), e(V_{NB_3} - V_{NB_4})]$. The sequence φ'' will be $[-1, -1, -1, -1, +1, +1]$. According to φ and φ'', P_1 builds the Table 2. From the table, P_1 knows V_{NB_3} is the largest element since its weight, which is $+4$, is the largest.

Theorem 11. *(Privacy-Preserving). Assuming the parties follow the protocol, the private data are securely protected.*

Proof. In Protocol 4, we need prove it from two aspects: (1) P_1 doesn't get information gain (e.g.,V_{NB_i}) for each attribute. What P_1 gets are $e(V_{NB_i} - V_{NB_j})$ for all $i, j \in [1, k], i > j$ and $+1/-1$ sequence. By $e(V_{NB_i} - V_{NB_j})$, P_1 cannot know each information gain since it is encrypted. By $+1/-1$ sequence, P_1 can only know whether or not V_{NB_i} is greater than P_j. (2) P_n doesn't obtain information gain for each attribute either. Since the sequence of $e(V_{NB_i} - V_{NB_j})$ is randomized before being send to P_n who can only know

the sequence of $V_{NB_i} - V_{NB_j}$, he can't get each individual information gain. Thus private data are not revealed.

Theorem 12. *(Efficiency). The computation of Protocol 4 is efficient from both computation and communication point of view.*

Proof. The total communication cost is upper bounded by αm^2. The total computation cost is upper bounded by $m^2 + m + 1$. Therefore, the protocols are very fast.

4 Overall Discussion

Our privacy-preserving classification system contains several components. In Sect. 3.4, we show how to correctly compute $e(Pr(a_i|v_j)))$. In Sect. 3.4, we discuss how to compute $e(Pr(v_j))$. In Sect. 3.4, we present protocols to compute $e(Pr(v_j)\prod_{i=1}^{n}\frac{Pr(a_i,v_j)}{Pr(v_j)})$ for each $v_j \in V$. In Sect. 3.4, we show how to compute the final naive Bayesian classifier. We discussed the correctness of the computation in each section. Overall correctness is also guaranteed.

As for the privacy protection, all the communications between the parties are encrypted, therefore, the parties who has no decryption key cannot gain anything out of the communication. On the other hand, there are some communication between the key generator and other parties. Although the communications are still encrypted, the key generator may gain some useful information. However, we guarantee that the key generator cannot gain the private data by adding random numbers in the original encrypted data so that even if the key generator get the intermediate results, there is little possibility that he can know the intermediate results. Therefore, the private data are securely protected with overwhelming probability.

In conclusion, we provide a novel solution for naive Bayesian classification over horizontally partitioned private data. Instead of using data transformation, we define a protocol using homomorphic encryption to exchange the data while keeping it private. Our classification system is quite efficient that can be envisioned by the communication and computation complexity. The total communication complexity is upper bounded by $\alpha(m^2 + 2n + 3t + 2)$. The computation complexity is upper bounded by $2N + m^2 + (5 + n)t + n + m + 1$.

References

1. G. Aggarwal, N. Mishra, and B. Pinkas. Secure computation of the k th-ranked element. In *EUROCRYPT pp 40–55*, 2004.
2. R. Agrawal and R. Srikant. Privacy-preserving data mining. In *Proceedings of the ACM SIGMOD Conference on Management of Data, pp 439–450.* ACM, May 2000.

3. J. Benaloh. Dense probabilistic encryption. In *Proceedings of the Workshop on Selected Areas of Cryptography, pp 120–128*, Kingston, Ontario, May 1994.
4. J. Domingo-Ferrer. A provably secure additive and multiplicative privacy homomorphism. In *Information Security Conference, pp 471–483*, 2002.
5. W. Du and Z. Zhan. Using randomized response techniques for privacy-preserving data mining. In *Proceedings of The 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Washington, DC, USA, August 24–27 2003.
6. C. Dwork and K. Nissim. Privacy-preserving datamining on vertically partitioned databases. In *CRYPTO 2004, pp 528–544*.
7. A. Evfmievski, J. Gehrke, and R. Srikant. Limiting privacy breaches in privacy preserving data mining. In *Proceedings of the 22nd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, pp 211–222*, San Diego, CA, June 9–12, 2003.
8. M. Franklin, Z. Galil, and M. Yung. An overview of secure distributed computing. Technical Report TR CUCS-00892, Department of Computer Science, Columbia University, 1992.
9. B. Goethals, S. Laur, H. Lipmaa, and T. Mielikainen. On secure scalar product computation for privacy-preserving data mining. In *Proceedings of The 7th Annual International Conference in Information Security and Cryptology (ICISC 2004), Volume 3506 of Lecture Notes in Computer Science, pp 104–120*, Seoul, Korea, December 2–3, 2004, Springer, Berlin Heidelberg New York, 2004.
10. O. Goldreich. Secure multi-party computation (working draft). http://www.wisdom.weizmann.ac.il /home/oded/public_html/foc.html, 1998.
11. O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, pp 218–229*, 1987.
12. S. Goldwasser. Multi-party computations: Past and present. In *Proceedings of the 16th Annual ACM Symposium on Principles of Distributed Computing*, Santa Barbara, CA USA, August 21–24, 1997.
13. Y. Lindell and B. Pinkas. Privacy preserving data mining. In *Advances in Cryptology – Crypto2000, Lecture Notes in Computer Science, Volume 1880*, 2000.
14. D. Naccache and J. Stern. A new public key cryptosystem based on higher residues. In *Proceedings of the 5th ACM conference on Computer and Communication Security, pp 59–66*, San Francisco, California, United States, 1998.
15. T. Okamoto and S. Uchiyama. A new public-key cryptosystem as secure as factoring. In *Eurocrypt'98, LNCS 1403, pp 308–318*, 1998.
16. P. Paillier. Public key cryptosystems based on composite degree residuosity classes. In *In Advances in Cryptology – Eurocrypt'99 Proceedings, LNCS 1592, pp 223–238*, Springer, Berlin Heidelberg New York, 1999.
17. R. Rivest, L. Adleman, and M. Dertouzos. On data banks and privacy homomorphisms. In *Foundations of Secure Computation, eds. R. A. DeMillo et al., Academic Press, pp 169–179*, 1978.
18. S. Rizvi and J.R. Haritsa. Maintaining data privacy in association rule mining. In *Proceedings of the 28th VLDB Conference*, Hong Kong, China, 2002.
19. L. Sweeney. k-anonymity: a model for protecting privacy. In *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems 10(5), 557–570*, 2002.

20. J. Vaidya and C. Clifton. Privacy preserving association rule mining in vertically partitioned data. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp 639–644*, Edmonton, Alberta, Canada, July 23–26, 2002.
21. A. C. Yao. Protocols for secure computations. In *Proceedings of the 23rd Annual IEEE Symposium on Foundations of Computer Science*, 1982.
22. J. Zhan and S. Matwin. Privacy-preserving nave bayesian classification over vertically partitioned data. In *IEEE ICDM Workshop on Foundations of Semantic Oriented Data and Web Mining*, Houston, Texas, USA, November 27–30, 2005.

Using Association Rules for Classification from Databases Having Class Label Ambiguities: A Belief Theoretic Method*

S.P. Subasingha[1], J. Zhang[2], K. Premaratne[1], M.-L. Shyu[1], M. Kubat[1], and K.K.R.G.K. Hewawasam[1]

[1] Department of Electrical and Computer Engineering, University of Miami, Coral Gables, FL, USA
s.subasingha@umiami.edu, kamal@miami.edu, shyu@miami.edu, mkubat@miami.edu, k.hewawasam@umiami.edu

[2] Hemispheric Center for Environmental Technology (HCET), Florida International University, Miami, FL, USA
zhangj@hcet.fiu.edu

Summary. This chapter introduces a belief theoretic method for classification from databases having class label ambiguities. It uses a set of association rules extracted from such a database. It is assumed that a training data set with an adequate number of pre-classified instances, where each instance is assigned with an integer class label, is available. We use a modified association rule mining (ARM) technique to extract the interesting rules from the training data set and use a belief theoretic classifier based on the extracted rules to classify the incoming feature vectors. The ambiguity modelling capability of belief theory enables our classifier to perform better in the presence of class label ambiguities. It can also address the issue of the training data set being unbalanced or highly skewed by ensuring that an approximately equal number of rules are generated for each class. All these capabilities make our classifier ideally suited for those applications where (1) different experts may have conflicting opinions about the class label to be assigned to a specific training data instance; and (2) the majority of the training data instances are likely to represent a few classes giving rise to highly skewed databases. Therefore, the proposed classifier would be extremely useful in security monitoring and threat classification environments where conflicting expert opinions about the threat level are common and only a few training data instances would be considered to pose a heightened threat level. Several experiments are conducted to evaluate our proposed classifier. These experiments use several databases from the UCI data repository and data sets collected from the airport terminal simulation platform developed at the Distributed Decision Environments (DDE) Laboratory at the Department of Electrical and Computer Engineering, University of Miami. The experimental results show that, while the proposed classifier's performance is comparable to some existing

*The preliminary version of the current draft was published in [31].

S.P. Subasingha et al.: *Using Association Rules for Classification from Databases Having Class Label Ambiguities: A Belief Theoretic Method*, Studies in Computational Intelligence (SCI) **118**, 539–562 (2008)
www.springerlink.com

classifiers when the databases have no class label ambiguities, it provides superior classification accuracy and better efficiency when class label ambiguities are present.

1 Introduction

Classification is one of the most widely addressed tasks in data mining and machine learning. Numerous interesting approaches have been proposed to handle this problem in the literature. However, in most of the classification problems, the knowledge about the conditional probability density function on each class label is usually unavailable and in such cases, several of the most widely used classifiers, e.g., Bayesian classifiers [13], may not be suitable. To address the issue when there is no evidence to support one form of the density function or another, a good solution is to build up a training data set of correctly classified feature vectors or samples and to classify each new incoming feature vector using the evidence provided by the 'nearby' samples from the training data set.

One important contribution along this approach was *voting k nearest neighbor classifier* [9], which is commonly referred to as the *KNN classifier*. The KNN classifier assigns a class to an unclassified feature vector based on the majority class of the k nearest neighbors in the training data set. It implicitly assumes that the neighbors for an incoming feature vector are concentrated in a small 'volume' in the feature space, and thus ensures sufficiently good resolution in the estimates of the different conditional densities. The KNN approach is very popular in the data mining and pattern recognition communities, and achieves good performance for most applications. As mentioned in [4], when the number of samples and the number of neighbors (N and k respectively) both tend to infinity while $k/N \rightarrow 0$, the error rate of the KNN classifier approaches the optimal Bayes error rate. However, this algorithm is only applicable for databases that do not possess any ambiguities.

Throughout the past decades, different modifications have been proposed to improve the performance of the KNN classifier [5,6]. For example, the work in [6] addresses one of the main drawbacks of its 'crisp' version cited in [4] by adding the capability of using training data sets with class label ambiguities. This classifier, which was based on the belief theory [16, 20, 21, 24, 28], not only handles training data sets with class label ambiguities, but also offers an improvement in classification accuracy. However, the classifier selects the neighbors by searching through the whole training data set. Hence, it requires a higher computational overhead, which is one of its main disadvantages.

With this background work in mind, consider a system that is used for threat detection and assessment purposes. Such a system would necessarily receive information from heterogeneous sensors (e.g., radiation sensors, ultrasound sensors, metal detectors, fume detectors, etc.). This information would then have to be utilized to classify potential threat targets to different threat classes. The training data set for such a system can be constructed with the

help of several domain experts who would classify the feature vectors (of the instances) into different threat level classes. In such a situation, it is likely that the domain experts would arrive at conflicting threat levels, which essentially introduces ambiguities into the class labels of the instances in the training data set. In addition, although the number of training data instances that are classified as having a heightened threat level would likely be very small, identification of targets possessing a heightened threat level would be of critical importance. For example, suppose the threat classes for an airport terminal security monitoring system are the following:

$$\{\texttt{NotDangerous}, \texttt{OfConcern}, \texttt{Dangerous}, \texttt{ExtremelyDangerous}\}. \quad (1)$$

In the training data set, one is likely to encounter a larger number of instances labeled as `NotDangerous` and very few labeled as `ExtremelyDangerous`. The classification results then may be biased toward the majority class.

In essence, a classifier for such a scenario needs to effectively address the following characteristics:

(C1) The training data set may contain ambiguities in the class labels due to the conflicting conclusions made by different domain experts.
(C2) The computational and storage requirements should be tolerable so that classification can be carried out in real-time.
(C3) The threat class distribution in the training data set can be highly skewed.

In this chapter, a classifier that can effectively take into consideration the above characteristics typical of a threat detection and assessment scenario is proposed [31]. To address (C1), several different and effective approaches are available, for example, rough set theory [29,30] and belief theory. The relationship between belief theory and other mechanisms can be found on [8,15,17,26]. In our proposed classifier, belief theoretic notions are adopted. This is mainly motivated by the fact that belief theory provides an easy and convenient way for handling ambiguities. A classifier facilitated with belief theoretic notions can improve the overall classification accuracy while providing a quantitative 'confidence interval' on the classification results.

To address (C2), the classifier is developed to operate on a rule set extracted by an ARM algorithm that has been appropriately modified to handle class label ambiguities. This rule set is significantly smaller than the size of the original database. This is the main difference between our proposed classifier and the *KNN-BF classifier* in [4]. ARM has demonstrated its capability of discovering interesting and useful co-occurring associations among data in large databases [1,14,19,25]. In the classifier mentioned in [18], it uses a modified ARM method to extract the association rules. However, it does not effectively address (C2).

To address (C3), the proposed ARM algorithm is applied to different partitions of the database where the partitioning is based on the class labels. This

simple modification results in an algorithm that generates an approximately equal number of rules from each class irrespective of whether it is a majority class or not.

The rest of this chapter is organized as follows. Our proposed classifier, which we refer to as the *ARM-KNN-BF classifier*, is discussed in Sect. 2; a primer on belief theory and the strategy we employ to accommodate highly skewed databases are also discussed in Sect. 2. Section 3 presents the experimental results. Conclusion, which includes several interesting research directions, appears in Sect. 4.

2 The Proposed ARM-KNN-BF Classifier

Although ARM in its original form can be deployed for extracting rules from large databases based on minimum support and minimum confidence conditions [1], it does not effectively address all the requirements (C1–C3) stated in Sect. 1. For example, one may develop a classifier based on rules that are generated by simply ignoring all the training data instances possessing class label ambiguities. But this strategy can potentially exclude a large portion of the training data instances that would have otherwise provided extremely crucial information. Moreover, since the training data set is highly skewed, a classifier built on it tends to favor the majority classes at the expense of the minority classes. Avoidance of this scenario is of paramount importance since this could result in devastating consequences in a threat classification environment.

As mentioned previously, we use belief theoretic notions to address (C1). One could alleviate the computational and storage burdens (C2) as well as the problems due to skewness of the database (C3) significantly by using a coherent set of rules in the classifier that effectively captures the re-occurring patterns in the database [31]. An effective ARM mechanism, as demonstrated in [18], can produce such a set of rules.

Each stage of the proposed algorithm can be summarized as follows: The training phase consists of partitioned ARM, rule pruning and rule refinement. The partitioned ARM mechanism generates an approximately equal number of rules in each class. The rule pruning and refinement processes use the training data set to select the important rules. The Dempster–Shafer belief theoretic notions [24] are utilized in the classification stage where a classifier that is capable of taking certain types of ambiguities into account when classifying an unknown instance has been introduced.

2.1 Belief Theory: An Introduction

Let $\Theta = \{\theta_1, \theta_2, \ldots, \theta n\}$ be a finite set of mutually exclusive and exhaustive 'hypotheses' about the problem domain. It signifies the corresponding 'scope

of expertise' and is referred to as its *frame of discernment (FoD)* [24]. A hypothesis θ_i represents the lowest level of discernible information in this FoD; it is referred to as a *singleton*. Elements in 2^{Θ}, the power set of Θ, form all hypotheses of interest. A hypothesis that is not a singleton is referred to as a *composite hypothesis*, e.g., (θ_1, θ_2). From now on, we use the term 'proposition' to denote both singleton and composite hypotheses. Cardinality of Θ is denoted by $|\Theta|$.

A *mass* function or a *basic probability assignment (BPA)* is a function $m : 2^{\Theta} \mapsto [0, 1]$ that satisfies

$$m(\emptyset) = 0 \text{ and } \sum_{A \subseteq \Theta} m(A) = 1. \tag{2}$$

Thus $m(A)$ can be interpreted as a measure that one is willing to commit *explicitly* to proposition A and not to any of its subsets. Committing support for a proposition does not necessarily imply that the remaining support is committed to its negation, thus relaxing the additivity axiom in the probability formalism. Propositions for which there is no information are not assigned an a priori mass. Rather, the mass of a composite proposition is allowed to move into its constituent singleton propositions only with the reception of further evidence.

The set of propositions each of which receives a non-zero mass is referred to as the focal elements; we denote it via $\mathcal{F}_{\Theta}$. The triple $\{\Theta, \mathcal{F}_{\Theta}, m(.)\}$ is referred to as the corresponding *body of evidence (BoE)*. The *vacuous BPA* that enables one to characterize complete ignorance is

$$m(A) = \begin{cases} 1, & \text{if } A = \Theta; \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

The belief of a proposition takes into account the support one has for all its proper subsets and is defined as

$$Bel(A) = \sum_{B \subseteq A} m(B). \tag{4}$$

It is a measure of the unambiguous support one has for A.

The notion of *plausibility* is used as a measure of the extent one finds the proposition A plausible and is defined as

$$Pl(A) = 1 - Bel(\overline{A}) = \sum_{B \cap A \neq \emptyset} m(B). \tag{5}$$

This indicates how much one's belief could be 'swayed' with further evidence.

A probability distribution $Pr(.)$ that satisfies $Bel(A) \leq Pr(A) \leq Pl(A)$, $\forall A \subseteq \Theta$, is said to be *compatible* with the underlying BPA $m(.)$. An example of such a probability distribution is the *pignistic probability distribution Bp* defined for each singleton $\theta_i \in \Theta$ [27] as

$$Bp(\theta_i) = \sum_{\theta_i \in A} \frac{m(A)}{|A|}. \tag{6}$$

When the information is provided by two independent BoEs $\{\Theta, \mathcal{F}_1, m_1(.)\}$ and $\{\Theta, \mathcal{F}_2, m_2(.)\}$ that span the *same* FoD Θ, they can be combined or 'fused' to create a single BoE $\{\Theta, \mathcal{F}, m(.)\}$ by using *Dempster's rule of combination (DRC)* [24]:

$$m(C) = \begin{cases} 0, & \text{for } C = \emptyset; \\ \dfrac{\sum_{A \cap B = C} m_1(A)\, m_2(B)}{K}, & \text{for } C \neq \emptyset, \end{cases} \tag{7}$$

where $K \equiv 1 - \sum_{A \cap B = \emptyset} m_1(A)\, m_2(B) \neq 0$. DRC is one of the most widely used belief theoretic evidence combination functions. This fusion operation is denoted as $m_1 \oplus m_2$ and referred to as the *orthogonal sum of* $m_1(.)$ *and* $m_2(.)$.

2.2 Data Model

The training data set (database) is denoted by $D_{TR} = \{T_i\}$, where T_i, $i = \overline{1, N_{TR}}$, is a data instance in the training data set; N_{TR} is the cardinality of D_{TR}. Assume that there are N_F features in the database. An instance can then be represented as follows:

$$T_i = < F_i, C_i >, \text{ where } F_i = < f_{1i}, f_{2i}, \ldots, f_{N_F i} > . \tag{8}$$

The i-th feature vector is represented by F_i; its features are denoted by f_{ji}, $j = \overline{1, N_F}$. The class label assigned to the i-th data instance is denoted by C_i.

With this notation in place, for all $i = \overline{1, N_{TR}}$, the FoD of the class label is taken to be identical, finite and equal to

$$\Theta_C = \{\theta_C^{(1)}, \theta_C^{(2)}, \ldots, \theta_C^{(N_C)}\}, \tag{9}$$

where N_C is the number of discernible class labels. Taking the example given in Sect. 1, we would have $\Theta_C = \{\texttt{NotDangerous}, \texttt{OfConcern}, \texttt{Dangerous}, \texttt{ExtremelyDangerous}\}$. We refer to the class label C_i as a *partially ambiguous* class label if it can be represented as a single composite proposition and $C_i \neq \Theta_C$; if $C_i = \Theta_C$, we refer to it as a *completely ambiguous* class label.

For all $i = \overline{1, N_{TR}}$, the FoD of each feature f_{ji} is also considered to be identical, finite and equal to

$$\Theta_{f_{ji}} = \{\Theta_{f_{ji}}^{(1)}, \Theta_{f_{ji}}^{(2)}, \ldots, \Theta_{f_{ji}}^{(n_{f_{ji}})}\}, \tag{10}$$

where $n_{f_{ji}}$ represents the number of possible values that the j-th feature may take. Thus the possible values that each feature vector F_i may take is a subset of the N_F-fold cross product of $2^{\Theta_{f_{ji}}}$, that is, $2^{\Theta_{f_{1i}}} \times 2^{\Theta_{f_{2i}}} \times \cdots \times 2^{\Theta_{f_{N_F i}}}$ [7].

2.3 Partitioning the Training Data Set

As we discussed in Sect. 1, special care has to be taken to account for the skewness of the database. To this end, we propose to apply the ARM algorithm to certain partitions of D_{TR}. The partitions are constructed based on the class labels of the training data instances that have been pre-classified. A separate partition is created for each class label, irrespective of whether the class label is a singleton or a composite proposition from Θ_C. Thus, we enumerate the 'newly created' class labels as $C^{(k)},\ k = \overline{1, N_{TC}}$, where $|\Theta_C| \leq N_{TC} \leq 2^{|\Theta_C|}$. Note that N_{TC} attains its upper bound when the class labels of the training data set span all possible subsets from Θ_C.

Denoting each partition by $P^{(k)},\ k = \overline{1, N_{TC}}$, the training data set can be represented as the union of the partitions, viz.,

$$D_{TR} = \bigcup_{k=1}^{N_{TC}} P^{(k)}. \tag{11}$$

It is clear that the partitions are mutually exclusive, i.e., $P^{(k_1)} \cap P^{(k_2)} = \emptyset$, whenever $k_1 \neq k_2$.

Recall the example in Sect. 1. Suppose certain training data instances have been classified as (`OfConcern`, `Dangerous`) due to the conflicting options of the experts. Thus, the training data set could be subdivided into five partitions, $\{P^{(1)}, P^{(2)}, P^{(3)}, P^{(4)}, P^{(5)}\}$, where the first four partitions would contain the training data instances with labels `NotDangerous`, `OfConcern`, `Dangerous` and `ExtremelyDangerous`, respectively, and $P^{(5)}$ would correspond to the ambiguous class label (`OfConcern`, `Dangerous`).

2.4 Partitioned-ARM

The ARM algorithm generates rules r_i of the form $X \rightarrow Y$, where the *antecedent* is $X \subseteq \Theta_F$ and *consequence* is $Y \subseteq \Theta_C$. The 'quality' of a rule is characterized by the *support* and *confidence* measures. To achieve an approximately equal number of rules inside each partition, we modify the support measure as

$$support = \frac{Count((X \rightarrow Y), P^{(k)})}{|P^{(k)}|}, \tag{12}$$

i.e., we calculate the support for a rule based on the partition. This is in contrast to the usual practice of calculating it based on the whole database. Here, $Count((X \rightarrow Y), P^{(k)})$ is the number of data instances $< X, Y >$ inside the partition $P^{(k)}$. We define the confidence of the rule using

$$confidence = \frac{Count((X \rightarrow Y), P^{(k)})}{\sum_{Z \subseteq \Theta_C} Count((X \rightarrow Z), D_{TR})} \tag{13}$$

Note that the confidence is calculated based on the whole database. Then we apply the Apriori algorithm [2] within each partition $P^{(k)}$ to generate the *class rule set* $R^{(k)} = \{r_\ell^{(k)}\}$:

$$R^{(k)} = \textbf{Apriori}\left(P^{(k)}, MinSupp(k)\right),\ k = \overline{1, N_{TC}}; \tag{14}$$

$$= \{r_\ell^{(k)}\} : F_\ell^{(k)} \to C^{(k)},\ \ell = \overline{1, N_{R^{(k)}}}, \tag{15}$$

where $N_{R^{(k)}}$ denotes the number of rules in the rule set $R^{(k)}$ and $MinSupp(k)$ denotes the minimum support value that each rule in $R^{(k)}$ must satisfy.

The final rule set R_{ARM} is the union of the class rule sets, that is,

$$R_{ARM} = \bigcup_{k=1}^{N_{TC}} R^{(k)}. \tag{16}$$

This procedure is what we refer to as *partitioned-ARM*.

An antecedent $F_\ell^{(k)}$ of a rule $r_\ell^{(k)}$ is of the form

$$F_\ell^{(k)} = < f_{1\ell}^{(k)}, f_{2\ell}^{(k)}, \ldots, f_{N_F\ell}^{(k)} >, \tag{17}$$

and we assume that each feature value $f_{j\ell}^{(k)}$ can assume either a singleton value $\Theta_{f_{j\ell}}^{(i)},\ i = \overline{1, n_{f_{j\ell}}}$, or the completely ambiguous value $\Theta_{f_{j\ell}}$. In other words, the only feature imperfection we allow is a missing value which is modeled as a complete ambiguity. This is in contrast to the class label ambiguity where both partial and complete ambiguities are allowed.

2.5 Rule Pruning

While the classification accuracy of the classifier is a function of the integrity of the rule set, its efficiency and the computational burden it imposes depends on the size of the rule set. Hence, to increase efficiency and reduce the associated computational burden, we propose to prune the rules.

Consider the antecedent $F_\ell^{(k)}$ of the rule $r_\ell^{(k)}$. Then, $f_{j\ell}^{(k)} = \Theta_{f_j}$ indicates complete ambiguity regarding the feature value $f_{j\ell}^{(k)}$. In this case, the feature value does not provide any valuable information. To formalize the relevant notions, we introduce

Definition 1 (Level of Abstraction (LoA)). *For a given rule* $r_\ell^{(k)} : F_\ell^{(k)} \to C^{(k)},\ \ell = \overline{1, N_{R^{(k)}}},\ k = \overline{1, N_{TC}}$, *define*

$$LoA[r_\ell^{(k)}] = \left| \bigcup_j \left\{ f_{j\ell}^{(k)} : f_{j\ell}^{(k)} = \Theta_{f_j} \right\} \right|. \tag{18}$$

Then

1. *Rule* $r_{\ell_1}^{(k)}$ *(and corresponding feature vector* $F_{\ell_1}^{(k)}$*) is said to be more* abstract *than rule* $r_{\ell_2}^{(k)}$ *(and corresponding feature vector* $F_{\ell_2}^{(k)}$*) if* $LoA[r_{\ell_1}^{(k)}] > LoA[r_{\ell_2}^{(k)}]$*; and*
2. *Rule* $r_{\ell}^{(k)}$ *is said to* cover *the training data instance* $T_i =< F_i, C_i >$, $i = \overline{1, N_D}$, *if*

$$f_{ji} = f_{j\ell}^{(k)},\ \forall j = \overline{1, N_F},\ \textit{whenever}\ f_{j\ell}^{(k)} \neq \Theta_{f_j}. \tag{19}$$

A rule $r_{\ell}^{(k)}$ having a higher LoA means that it contains more insignificant features. In our pruning scheme, we assign a lower importance to such rules. Furthermore, there may be more than one rule which may cover a certain data instance. These redundant rules can be safely removed from the training data set. To this end, once we have more than one rule that covers a certain training data instance, we allocate a higher importance to the rules that possess higher confidence and lower LoA. With this in mind, the rule set R_{ARM} is sorted using the following criteria:

1. First, sort the rule set based on descending order of the confidence values.
2. Second, for those rules having the same confidence value, sort the rules by ascending order of the LoA values.

This sorting scheme differs from that used in [18] where the second level of sorting is determined by the support value. The reason for our choice lies in our desire to accommodate class label ambiguities.

Then, starting from the first rule from the sorted rule set, the instances that can be covered by each rule are removed from the training data set. If the set of removed training data instances corresponding to a rule is not empty, that particular rule will be added to the final rule set; otherwise it is pruned. This process is continued until either all rules in R_{ARM} are exhausted or no training data instances are left in the training data set. At termination, if the training data set is not empty, all its remaining data instances are added to the rule set with confidence 1.0 since they provide evidence that could not be captured by the rules in R_{ARM}.

This process, which is referred to as *RulePruning* algorithm, can be described as follows:

RulePruning$(R_{ARM}, D_{TR})\{$
while $(\neg$ empty(R_{ARM}) & $\neg$ empty$(D_{TR}))\{$
r=Top(R_{ARM});
if $(\neg$ empty$(D_r =$ **covered**$(r, D_{TR})))\{$
$R_{Pruned} = R_{Pruned} \cup r$;
$R_{ARM} = R_{ARM} \setminus r$;
$D_{TR} = D_{TR} \setminus D_r;\}\}$
if $(\neg$ empty$(D_{TR}))$ $\{$
$R_{Pruned} = R_{Pruned} \cup D_{TR};\}\}$

Here, R_{Pruned} is the rule set eventually selected from the pruning algorithm. The function **covered**(r, D_{TR}) generates D_r, the set of all training data instances covered by the rule r.

2.6 The Classifier

We now describe how our ARM-KNN-BF classifier is developed based on the rule set developed above.

Let $F^\sharp =< f_1^\sharp, f_2^\sharp, \ldots, f_{N_F}^\sharp >$ be an incoming feature vector that needs to be classified into one of the classes from Θ_C. We view each rule $r_\ell^{(k)}$ as a piece of evidence that alters our belief about how well the unclassified feature vector $F^\sharp$ belongs to the class $C^{(k)} \subseteq \Theta_C$. We would be able to make this claim with a higher 'confidence,' if:

1. The confidence value $c_\ell^{(k)}$ of the rule $r_\ell^{(k)}$ is higher; and
2. The distance between $F^\sharp$ and the antecedent $F_\ell^{(k)}$ of the rule $r_\ell^{(k)}$ is smaller.

With these observations in mind, we define the following BPA:

$$m_\ell^{\sharp,(k)}(A) = \begin{cases} \alpha_\ell^{(k)}, & \text{if } A = C^{(k)}; \\ 1 - \alpha_\ell^{(k)}, & \text{if } A = \Theta_C, \end{cases} \tag{20}$$

where

$$\alpha_\ell^{(k)} = \beta\, c_\ell^k e^{\gamma\, Dist[F^\sharp, F_\ell^{(k)}]}. \tag{21}$$

Here, $\beta \in [0,1]$ and $\gamma < 0$ are parameters to be chosen; the distance function $Dist[F^\sharp, F_\ell^{(k)}]$ is an appropriate measure that indicates the distance between $F^\sharp$ and the antecedent $F_\ell^{(k)}$ of the rule. We choose it as

$$Dist[F^\sharp, F_\ell^{(k)}] = \left\| \left[d_1\ d_2 \cdots d_{N_F} \right]^T \div N_{\overline{F}} \right\|, \tag{22}$$

where, for $j = \overline{1, N_F}$,

$$d_j = \begin{cases} |f_j^\sharp - f_{j\ell}^{(k)}| & \text{whenever } f_{j\ell}^{(k)} \neq \Theta_{f_j}; \\ 0 & \text{otherwise.} \end{cases} \tag{23}$$

Here $[\centerdot]^T$ denotes matrix transpose and $N_{\overline{F}}$ denotes the number of non-ambiguous feature values.

2.7 Fused BPAs and Rule Refinement

At this juncture, we have created a BPA $m_\ell^{\sharp,(k)}(\centerdot) : 2^{\Theta_C} \to [0,1]$ corresponding to each rule $r_\ell^{(k)},\ \ell = \overline{1, N_{R^{(k)}}},\ k = \overline{1, N_{TC}}$. Now, we may use DRC to combine these BPAs to get a fused BPA $m^\sharp$. When the generated rule set is large, and if the computational burden associated with the application of DRC is of

concern, we may use a certain number of rules, say K, whose antecedents are the closest to the new feature vector $F^\sharp$. Then, only K BPAs would need to be fused using the DRC.

To ensure the integrity of the generated rule set, the classifier developed as above was used to classify the feature vectors $\{F_\ell^{(k)}\}$, $\ell = \overline{1, N_{R^{(k)}}}$, $k = \overline{1, N_{TC}}$, of the training data set itself. The classification error associated with $F_\ell^{(k)}$ can be considered to reveal that the rule set does not possess sufficient information to correctly identify the training data set. With this observation in mind, R_{Pruned} was supplemented by each training data instance whose feature vector $F_\ell^{(k)}$ was not correctly classified; the confidence measure of such a rule was allocated a value of 1.0. This refined rule set is what constitutes our proposed ARM-KNN-BF classifier.

2.8 An Example of Rule Generation

In this section, we use a slightly modified variation of a data set from [22] to clarify and illustrate the various steps involved in our proposed rule generation algorithm. The data set being considered possesses three features and two classes. See Table 1. The FoDs corresponding to the features are as follows:

$$\begin{array}{ll} \text{Outlook}: & \Theta_{f_1} = \{\text{sunny}, \text{overcast}, \text{rainy}\}; \\ \text{Humidity}: & \Theta_{f_2} = \{\text{LOW}, \text{MEDIUM}, \text{HIGH}\}; \\ \text{Windy}: & \Theta_{f_3} = \{\text{TRUE}, \text{FALSE}\}; \\ \text{Decision}: & \Theta_C = \{\text{Play}, \text{Don't Play}\}. \end{array} \tag{24}$$

Table 1. The training data set under consideration

Outlook	Humidity	Windy	Decision
sunny	MEDIUM	TRUE	Play
sunny	LOW	FALSE	Play
sunny	MEDIUM	FALSE	Play
overcast	HIGH	FALSE	Don't play
rainy	MEDIUM	FALSE	Don't play
rainy	HIGH	TRUE	Θ_C
overcast	LOW	FALSE	Play
sunny	MEDIUM	TRUE	Play
rainy	HIGH	FALSE	Don't play
overcast	MEDIUM	FALSE	Don't play
overcast	HIGH	TRUE	Θ_C
sunny	HIGH	TRUE	Θ_C
sunny	LOW	TRUE	Play
overcast	LOW	TRUE	Play
sunny	MEDIUM	FALSE	Play
overcast	HIGH	TRUE	Don't play
sunny	LOW	TRUE	Play

Table 2. Partitioned data set from Table 1

Outlook	Humidity	Windy	Decision
sunny	MEDIUM	TRUE	Play
sunny	LOW	FALSE	Play
sunny	MEDIUM	FALSE	Play
overcast	LOW	FALSE	Play
sunny	MEDIUM	TRUE	Play
sunny	LOW	TRUE	Play
overcast	LOW	TRUE	Play
sunny	MEDIUM	FALSE	Play
sunny	LOW	TRUE	Play
overcast	HIGH	TRUE	Don't play
rainy	HIGH	FALSE	Don't play
overcast	MEDIUM	FALSE	Don't play
overcast	HIGH	FALSE	Don't play
rainy	MEDIUM	FALSE	Don't play
rainy	HIGH	TRUE	Θ_C
overcast	HIGH	TRUE	Θ_C
sunny	HIGH	TRUE	Θ_C

The data sets, after being partitioned based on the class label, are given in Table 2. To retain simplicity of this example, we set the support value at 0.25 which is greater than the values that were used in our simulations in Sect. 3. Table 3 shows the rules generated. For this simple example, at this stage, the number of rules generated were 10, 10 and 15 rules corresponding to the classes 'Play', 'Don't Play' and Θ_C, respectively.

The pruning process (see Sect. 2.5) produces a reduced rule set. This final rule set is shown in Table 4. The support value is not indicated in this table since it does not play a critical role in the classification stage. As can be seen from Table 4, the rules with lower levels of abstraction appear to have had a better chance of being selected to the pruned rule set.

Next, this pruned rule set is used in the rule refinement stage (see Sect. 2.7). The rule set generated at the conclusion of the rule refinement stage is given in Table 5. Note that the last rule in Table 5 was added into the final rule set because its corresponding training instance was incorrectly classified at the rule refinement stage. When conflicting rules are present in the final rule set, masses to those rules are assigned based on their confidence values, and then the Dempster's rule of combination (shown in (7)) takes this into account when making the final decision in the classification stage.

Table 3. The generated rule set

Outlook	Humidity	Windy	Decision	Support	Confidence
	LOW		Play	0.5556	1.0000
sunny	MEDIUM		Play	0.4444	1.0000
sunny	LOW		Play	0.3333	1.0000
sunny		FALSE	Play	0.3333	1.0000
	LOW	TRUE	Play	0.3333	1.0000
sunny			Play	0.7778	0.8750
sunny		TRUE	Play	0.4444	0.8000
	MEDIUM		Play	0.4444	0.6667
		TRUE	Play	0.5556	0.5556
		FALSE	Play	0.4444	0.5000
rainy		FALSE	Don't play	0.4000	1.0000
	HIGH	FALSE	Don't play	0.4000	1.0000
rainy			Don't play	0.4000	0.6667
overcast	HIGH		Don't play	0.4000	0.6667
overcast		FALSE	Don't play	0.4000	0.6667
		FALSE	Don't play	0.8000	0.5000
overcast			Don't play	0.6000	0.5000
	HIGH		Don't play	0.6000	0.5000
	MEDIUM	FALSE	Don't play	0.4000	0.5000
	MEDIUM		Don't play	0.4000	0.3333
sunny	HIGH		Θ_C	0.3333	1.0000
rainy		TRUE	Θ_C	0.3333	1.0000
rainy	HIGH	TRUE	Θ_C	0.3333	1.0000
sunny	HIGH	TRUE	Θ_C	0.3333	1.0000
	HIGH	TRUE	Θ_C	1.0000	0.7500
	HIGH		Θ_C	1.0000	0.5000
rainy	HIGH		Θ_C	0.3333	0.5000
overcast	HIGH	TRUE	Θ_C	0.3333	0.5000
		TRUE	Θ_C	1.0000	0.3333
rainy			Θ_C	0.3333	0.3333
overcast	HIGH		Θ_C	0.3333	0.3333
overcast		TRUE	Θ_C	0.3333	0.3333
sunny		TRUE	Θ_C	0.3333	0.2000
overcast			Θ_C	0.3333	0.1667
sunny			Θ_C	0.3333	0.1250

3 Experimental Results

Several experiments on two different groups of databases (databases with and without class label ambiguities) were conducted to compare the performance of our proposed ARM-KNN-BF classifier with some existing classification methods.

Table 4. Pruned rule set

Outlook	Humidity	Windy	Decision	Confidence
sunny	MEDIUM		Play	1.0000
sunny		FALSE	Play	1.0000
	LOW	TRUE	Play	1.0000
	LOW		Play	1.0000
rainy		FALSE	Don't play	1.0000
	HIGH	FALSE	Don't play	1.0000
overcast	HIGH		Don't play	0.6667
overcast		FALSE	Don't play	0.6667
rainy	HIGH	TRUE	Θ_C	1.0000
sunny	HIGH	TRUE	Θ_C	1.0000
overcast	HIGH	TRUE	Θ_C	0.5000

Table 5. Final rule set generated at the conclusion of the rule refinement stage

Outlook	Humidity	Windy	Decision	Confidence
sunny	MEDIUM		Play	1.0000
sunny		FALSE	Play	1.0000
	LOW	TRUE	Play	1.0000
	LOW		Play	1.0000
rainy		FALSE	Don't play	1.0000
	HIGH	FALSE	Don't play	1.0000
overcast	HIGH		Don't play	0.6667
overcast		FALSE	Don't play	0.6667
rainy	HIGH	TRUE	Θ_C	1.0000
sunny	HIGH	TRUE	Θ_C	1.0000
overcast	HIGH	TRUE	Θ_C	0.5000
overcast	HIGH	TRUE	Don't play	1.0000

These experiments use several databases from the UCI data repository [3] and data sets collected from the airport terminal simulation platform developed at the Distributed Decision Environments (DDE) Laboratory at the Department of Electrical and Computer Engineering, University of Miami. The databases contain both numerical and nominal attributes. All the classification accuracies are presented with 10-fold sub-sampling where the training and testing data sets are constructed by taking 70% and 30% of the data instances in the database respectively. The training data set was used to generate the classification rules and the testing data set was used to evaluate its performance.

3.1 UCI Databases Without Class Label Ambiguities

Several UCI databases [3] were used to compare the classification accuracy of the proposed ARM-KNN-BF classifier with the KNN classifier [9], c4.5rules [23], KNN-BF classifier [5] and ARM classifier [18]. The parameter values and accuracy results for the ARM classifier were borrowed from [18]. Table 6 shows the support and confidence parameters used by the ARM-KNN-BF classifier for different databases.

The support and confidence parameters play a vital role in the performance evaluation. For most of the databases, the support values are between 0.01 and 0.10; the exception is the `Zoo` database for which we use 0.5. The lower the support value is, the larger the number of rules that can be captured and more information can be found in the rule set. However, this may result in a rule set with noise. Therefore, when the support value is too small, it may deteriorate the integrity of the overall rule set. On the other hand, when the support value is too large, some interesting rules may not be captured. Therefore, empirical studies are needed to determine the best support value for each database. For the confidence values, they were kept between 0.3 and 0.8, a relatively higher value compared to the support. To keep the processing complexity at a tolerable level, we wanted to keep the number of neighbors K limited to 10. For these values, we observed no significant change in performance. Hence, $K = 7$ was selected for all the experiments.

As the classifier is based on belief theoretic notions, it generally assigns a 'soft' class label. For purposes of comparison, a 'hard' decision (i.e., a singleton class label) was desired. Different strategies have been proposed in the literature to achieve this [10]; we used the pignistic probability distribution [27].

The number of generated rules directly relates to the efficiency of the algorithm. Table 7 compares the average number of rules generated per class in the ARM-KNN-BF with three other classifiers.

Table 6. Non-ambiguous UCI databases – parameters (support and confidence values) used by the ARM-KNN-BF classifier

Database	Support	Confidence
`Breast cancer`	0.05	0.3
`Car`	0.03	0.5
`Diabetes`	0.03	0.3
`Iris`	0.08	0.8
`Monks`	0.06	0.6
`Post-operation patient`	0.05	0.5
`Scale`	0.06	0.6
`Solar flares`	0.06	0.8
`Tic-Tac-Toe`	0.04	0.5
`Voting`	0.03	0.8
`Wine`	0.07	0.3
`Zoo`	0.50	0.5

Table 7. Non-ambiguous UCI databases – average number of rules generated per class

Database	KNN & KNN-BF	ARM	ARM-KNN-BF
Breast cancer	242	49	76
Car	302	N/A	71
Diabetes	258	57	169
Iris	32	5	18
Monks	200	N/A	48
Post-operation patient	23	N/A	18
Scale	438	N/A	61
Solar flares	743	N/A	42
Tic-Tac-Toe	334	8	70
Voting	297	N/A	57
Wine	41	7	30
Zoo	63	7	5

Table 8. Non-ambiguous UCI databases – classification accuracy

Database	KNN	c4.5rules	ARM	KNN-BF	ARM-KNN-BF
Breast cancer	0.97	0.95	0.93	0.96	0.97
Car	0.92	0.93	N/A	0.93	0.93
Diabetes	0.70	0.72	0.71	0.72	0.76
Iris	0.94	0.94	0.96	0.93	0.95
Monks	0.92	0.98	N/A	0.97	0.95
Post-operation patient	0.69	0.76	N/A	0.74	0.76
Scale	0.83	0.85	N/A	0.84	0.84
Solar flares	0.82	0.83	N/A	0.82	0.81
Tic-Tac-Toe	0.92	0.98	0.93	1.00	0.99
Voting	0.91	0.92	N/A	0.90	0.93
Wine	0.94	0.91	0.96	0.92	0.96
Zoo	0.90	0.92	0.95	0.96	0.98

Although the number of rules generated for the ARM classifier is significantly less compared to others, it fails to handle class label ambiguities. Along with the proposed ARM-KNN-BF classifier, the KNN and KNN-BF classifiers are the only classifiers that are applicable in such a situation. Among these, the ARM-KNN-BF classifier possesses a significantly fewer number of rules.

Tables 8 and 9 give the classification accuracy and the standard deviation corresponding to these different UCI databases. For the ARM classifier, the best average accuracy was used (i.e., CBA-CAR plus infrequent rules reported in [18]). Table 8 shows that the ARM-KNN-BF classifier performs comparatively well with the other classifiers. Furthermore, it operates on a much smaller rule set compared to the KNN and KNN-BF classifiers.

Table 9. Non-ambiguous UCI databases – standard deviation of classification accuracy

Database	KNN	c4.5rules	KNN-BF	ARM-KNN-BF
`Breast cancer`	1.08	0.81	3.03	1.16
`Car`	1.12	1.51	0.97	1.57
`Diabetes`	2.13	4.23	2.22	3.99
`Iris`	2.10	2.74	2.00	2.25
`Monks`	1.23	0.81	1.74	1.32
`Post-operation patient`	4.31	1.34	3.21	2.16
`Scale`	1.24	0.96	1.29	1.05
`Solar flares`	1.73	1.51	1.03	1.82
`Tic-Tac-Toe`	2.22	2.15	3.03	1.81
`Voting`	2.76	1.93	1.97	2.14
`Wine`	2.50	3.71	2.10	2.83
`Zoo`	3.98	1.47	2.15	1.03

Of course, the real strength of a belief theoretic classifier lies in its ability of performing well even in the presence of ambiguities. The experiments in the next section are conducted to demonstrate this claim.

3.2 UCI Databases with Class Label Ambiguities

Since the UCI databases do not possess ambiguities, to test the performance of the proposed classifier, ambiguities were artificially introduced. Different types of ambiguities one may encounter in databases are discussed in [28]. Although the most natural way is to introduce these ambiguities randomly, we used a more reasonable strategy motivated by the fact that experts are likely to assign ambiguous class labels whose constituents are 'close' to each other. For example, while it is likely that an expert would allocate the ambiguous label (`OfConcern`, `Dangerous`), the label (`OfConcern`, `ExtremelyDangerous`) is highly unlikely.

With this in mind, we proceed as follows to introduce class label ambiguities: Consider an instance T_i which has been allocated the class label C_i. Let us refer to N of its closest neighbors as the *N-neighborhood* of T_i; here, N is a pre-selected parameter. If the class label C_j occurs more than a pre-specified percentage $p\%$ among the instances in this N-neighborhood of T_i, the class label of T_i is made ambiguous by changing its class label to (C_i, C_j) from C_i. For example, suppose T_i is labeled as C_3. With $N = 10$, suppose the class labels of the 10-neighborhood of T_i are distributed as follows: 3 belong to class C_2, 6 belong to class C_3 and 1 belong to class C_4. With $p = 25\%$, both C_2 and C_3 exceed the pre-specified percentage. Hence, T_i is assigned the ambiguous class label (C_2, C_3). The level of ambiguity could be controlled by varying the value of p. In our experiments, we used $p = 25\%$.

Table 10. Ambiguous UCI databases – parameters (support and confidence values) used by the ARM-KNN-BF classifier

Database	Support	Confidence
Breast cancer	0.10	0.8
Car	0.02	0.5
Diabetes	0.10	0.6
Iris	0.06	0.8
Monks	0.06	0.6
Post-operation patient	0.05	0.8
Scale	0.05	0.5
Solar flares	0.06	0.8
Tic-Tac-Toe	0.04	0.5
Voting	0.30	0.8
Wine	0.07	0.3
Zoo	0.50	0.5

Table 11. Ambiguous UCI databases – average number of rules generated per class

Database	KNN & KNN-BF	ARM-KNN-BF	% Reduction
Breast cancer	242	68	72%
Car	302	103	66%
Diabetes	258	120	53%
Iris	32	19	41%
Monks	341	62	82%
Post-operation patient	20	17	15%
Scale	162	60	67%
Solar flares	760	45	94%
Tic-Tac-Toe	333	65	80%
Voting	123	63	49%
Wine	53	40	25%
Zoo	10	5	50%

We first compared the proposed ARM-KNN-BF classifier with the KNN and KNN-BF classifiers because they are capable of handling class label ambiguities. Table 10 shows the support and confidence parameters used by the ARM-KNN-BF classifier for different databases. Similar to the non-ambiguous databases case (see Table 6), the support values were kept low (0.1 or less) for most of the databases and the corresponding confidence values were kept fairly high. As before, the number of neighbors K was chosen as 7 except for the Zoo databases (for which $K = 3$) because it possesses a significantly fewer number of rules per class.

As can be seen from Tables 11 and 12, the ARM-KNN-BF classifier requires a significantly fewer number of rules and achieves a better classification accuracy compared with the other two classifiers.

Table 12. Ambiguous UCI databases – classification accuracy (score)

Database	KNN	KNN-BF	ARM-KNN-BF
`Breast cancer`	0.92	0.82	0.94
`Car`	0.89	0.83	0.88
`Diabetes`	0.78	0.76	0.88
`Iris`	0.91	0.94	0.92
`Monks`	0.86	0.87	0.89
`Post-operation patient`	0.82	0.76	0.85
`Scale`	0.81	0.79	0.84
`Solar flares`	0.80	0.80	0.84
`Tic-Tac-Toe`	0.80	0.82	0.85
`Voting`	0.86	0.81	0.89
`Wine`	0.87	0.87	0.91
`Zoo`	0.89	0.91	0.93

The '% Reduction' column in Table 11 shows the percentage reduction in the number of rules used by the ARM-KNN-BF classifier when compared with the others. In calculating the classification accuracy in Table 12, some strategy needs to be developed to compare the 'correctness' of ambiguous classifications. For example, suppose the actual class label is (C_1, C_2). How does the accuracy calculation be done if the classified label is C_1, (C_1, C_2) or (C_2, C_3)? Clearly, the classification (C_1, C_2) must be considered 'perfect' and should be assigned the maximum score. How does one evaluate the classification accuracies corresponding to the classifications C_1 and (C_2, C_3)? To address this issue, the following measure, which we refer to as the *score*, is employed:

$$score = \frac{|\text{True label} \cap \text{Assigned label}|}{|\text{True label} \cup \text{Assigned label}|} \tag{24}$$

With this measure, the scores of the class labels $C1$, (C_1, C_2) and (C_2, C_3) would be 1/2, 1/1 and 1/3, respectively. Table 12 gives the score values for the three classifiers.

3.3 Experiments on the Airport Terminal Simulation Platform

We have developed a simple simulation platform to mimic an airport terminal in our Distributed Decision Environment (DDE) Laboratory at the Department of Electrical and Computer Engineering, University of Miami, to test the performance of the algorithms we have developed. The platform consists of three areas; each area has two gates at its entrance and exit. The potential threat carriers carry different combinations of features which we refer to as *property packs*. Carriers entering and exiting each area are tracked using an overhead camera. Each gate of the platform contains a stationary sensor module that extracts the intensity of the features in the property pack. Based on

Table 13. Airport terminal simulation experiment – class label distribution

Class label	Number of instances
`NotDangerous`	154
`Dangerous`	66
`(NotDangerous,Dangerous)`	88
Total	308

Table 14. Airport terminal simulation experiment – classification accuracy (score)

Algorithm	Score	Rules/Class
KNN	0.89	68
KNN-BF	0.92	68
ARM-KNN-BF	0.94	31

these properties, each carrier is assigned a carrier type by a program module that takes into account expert knowledge.

For our set of experiments, we concentrated on the potential threat carriers within only one area of the platform. This area was subdivided into nine sub-areas. A data record possessing nine features, each corresponding to one sub-area, was periodically (at 0.5 s intervals) generated for this region. Each feature contains the carrier type located within its corresponding subregion. Endowing five carriers with different property packs, various scenarios were created to represent `NotDangerous` and `Dangerous` environments. For instance, carriers each having 'weak' or no property pack can be considered to reflect `NotDangerous` conditions; on the other hand, even a single carrier carries a 'strong' property pack can be considered a `Dangerous` condition. Other combinations were allocated appropriate class labels. Clearly, allocation of a 'crisp' class label for certain combinations of number of carriers and property packs would be difficult; such situations warrant the ambiguous class label `(NotDangerous,Dangerous)`.

The test database contains a total number of 308 instances whose class labels are distributed as in Table 13.

The classification scores, again based on 10-fold sub-sampling, are shown in Table 14. It is evident that the proposed ARM-KNN-BF classifier achieves better performance with a much smaller set of rules than that of the KNN and KNN-BF classifiers.

4 Conclusion

In this chapter, we have developed a novel belief theoretic ARM based classification algorithm that addresses the following concerns:

- Class label ambiguities in training databases;
- Computational and storage constraints; and
- Skewness of the databases.

Class label ambiguities naturally arise in application scenarios especially when domain expert knowledge is sought for classifying the training data instances. We use a belief theoretic technique for addressing this issue. It enables the proposed ARM-KNN-BF classifier to conveniently model the class label ambiguities. Each generated rule is then treated as a BoE providing another 'piece of evidence' for purposes of classifying an incoming data instance. The final classification result is based upon the fused BoE generated by DRC. Skewness of the training data set can also create significant difficulties in ARM because the majority classes tend to overwhelm the minority classes in such situations. The partitioned-ARM strategy we employ creates an approximately equal number of rules for each class label thus solving this problem. The use of rules generated from only the nearest neighbors (instead of using the complete rule set) enables the use of a significantly fewer number of rules in the BoE combination stage. This makes our classifier more computationally efficient. Applications where these issues are of critical importance include threat detection and assessment scenarios.

As opposed to the other classifiers (such as c4.5 and KNN), belief theoretic classifiers capture a much richer information content in the decision making stage. Furthermore, how neighbors are defined in the ARM-KNN-BF classifier is different than the strategy employed in the KNN-BF and KNN classifiers. Due to the fact that the rules in the ARM-KNN-BF classifier are generated via ARM, the rules capture the associations within the training data instances. Thus, it is able to overcome 'noise' effects that could be induced by individual data instances. This results in better decisions. Of course. a much smaller rule set in the classification stage significantly reduces the storage and computational requirements, a factor that plays a major role when working with huge databases.

The work described above opens up several interesting research issues that warrant further study. In security monitoring and threat classification, it is essential that one errs on the side of caution. In other words, it is always better to overestimate the threat level than under-estimate it. So, development of strategies that overestimate threat level at the expense of under-estimating it is warranted.

Another important research problem involves the extension of this work to accommodate more general types of imperfections in both class labels and features. The work described herein handles ambiguities in class labels only; ways to handle general belief theoretic class label imperfections [28] would be extremely useful. Development of strategies that can address general belief theoretic imperfections in features would further enhance the applicability of this work. Some initial work along this line appears in [11, 12].

Acknowledgment

This work is based upon work supported by NSF Grants IIS-0325260 (ITR Medium), IIS-0513702 and EAR-0323213. The work of J. Zhang was conducted while he was at the University of Miami.

References

1. R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In *Proceedings of ACM SIGMOD International Conference on Management of Data*, pages 207–216, Washington DC, May 1993
2. R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In *Proceedings of International Conference on Very Large Data Bases (VLDB'94)*, pages 487–499, Santiago de Chile, Chile, September 1994
3. C. L. Blake and C. J. Merz. UCI repository of machine learning databases, 1998
4. T. M. Cover and P. E. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, January 1967
5. T. Denoeux. The k-nearest neighbor classification rule based on Dempster-Shafer theory. *IEEE Transactions on Systems, Man and Cybernetics*, 25(5):804–813, May 1995
6. S. A. Dudani. The distance-weighted k-nearest-neighbor rule. *IEEE Transactions on Systems, Man and Cybernetics*, 6(4):325–327, April 1976
7. S. Fabre, A. Appriou, and X. Briottet. Presentation and description of two classification methods using data fusion on sensor management. *Information Fusion*, 2:49–71, 2001
8. R. Fagin and J. Y. Halpern. A new approach to updating beliefs. In P. P. Bonissone, M. Henrion, L. N. Kanal, and J. F. Lemmer, editors, *Proceedings of Conference on Uncertainty in Artificial Intelligence (UAI'91)*, pages 347–374. Elsevier Science, New York, NY, 1991
9. E. Fix and J. L. Hodges. Discriminatory analysis: nonparametric discrimination: consistency properties. Technical Report 4, USAF School of Aviation Medicine, Randolph Field, TX, 1951
10. S. L. Hegarat-Mascle, I. Bloch, and D. Vidal-Madjar. Introduction of neighborhood information in evidence theory and application to data fusion of radar and optical images with partial cloud cover. *Pattern Recognition*, 31(11):1811–1823, November 1998
11. K. K. R. G. K. Hewawasam, K. Premaratne, M.-L. Shyu, and S. P. Subasingha. Rule mining and classification in the presence of feature level and class label ambiguities. In K. L. Priddy, editor, *Intelligent Computing: Theory and Applications III*, volume 5803 of *Proceedings of SPIE*, pages 98–107. March 2005
12. K. K. R. G. K. Hewawasam, K. Premaratne, S. P. Subasingha, and M.-L. Shyu. Rule mining and classification in imperfect databases. In *Proceedings of International Conference on Information Fusion (ICIF'05)*, Philadelphia, PA, July 2005
13. H.-J. Huang and C.-N. Hsu. Bayesian classification for data from the same unknown class. *IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics*, 32(2):137–145, April 2002

14. T. Karban, J. Rauch, and M. Simunek. SDS-rules and association rules. In *Proceedings of ACM Symposium on Applied Computing (SAC'04)*, pages 482–489, Nicosia, Cyprus, March 2004
15. M. A. Klopotek and S. T. Wierzchon. A new qualitative rought-set approach to modeling belief functions. In L. Polkowski and A. Skowron, editors, *Proceedings of International Conference on Rough Sets and Current Trends in Computing (RSCTC'98)*, volume 1424 of *Lecture Notes in Computer Science*, pages 346–354. Springer, Berlin Heidelberg New York, 1998
16. E. C. Kulasekere, K. Premaratne, D. A. Dewasurendra, M.-L. Shyu, and P. H. Bauer. Conditioning and updating evidence. *International Journal of Approximate Reasoning*, 36(1):75–108, April 2004
17. T. Y. Lin. Fuzzy partitions II: Belief functions. A probabilistic view. In L. Polkowski and A. Skowron, editors, *Proceedings of International Conference on Rough Sets and Current Trends in Computing (RSCTC'98)*, volume 1424 of *Lecture Notes in Computer Science*, pages 381–386. Springer, Berlin Heidelberg, New York, 1998.
18. B. Liu, W. Hsu, and Y. M. Ma. Integrating classification and association rule mining. In *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'98)*, pages 80–86, New York, NY, August 1998
19. A. A. Nanavati, K. P. Chitrapura, S. Joshi, and R. Krishnapuram. Mining generalized disjunctive association rules. In *Proceedings of International Conference on Information and Knowledge Management (CIKM'01)*, pages 482–489, Atlanta, GA, November 2001
20. S. Parsons and A. Hunter. A review of uncertainty handling formalisms. In A. Hunter and S. Parsons, editors, *Applications of Uncertainty Formalisms*, volume 1455 of *Lecture Notes in Artificial Intelligence*, pages 8–37. Springer, Berlin Heidelberg New York, 1998
21. K. Premaratne, J. Zhang, and K. K. R. G. K. Hewawasam. Decision-making in distributed sensor networks: A belief-theoretic Bayes-like theorem. In *Proceedings of IEEE International Midwest Symposium on Circuits and Systems (MWSCAS'04)*, volume II, pages 497–500, Hiroshima, Japan, July 2004
22. J. R. Quinlan. Decision trees and decision-making. *IEEE Transactions on Systems, Man and Cybernetics*, 20(2):339–346, March/April 1990l
23. J. R. Quinlan. *C4.5: Programs for Machine Learning*. Representation and Reasoning Series. Morgan Kaufmann, San Francisco, CA, 1993
24. G. Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, Princeton, NJ, 1976
25. M.-L. Shyu, S.-C. Chen, and R. L. Kashyap. Generalized affinity-based association rule mining for multimedia database queries. *Knowledge and Information Systems (KAIS), An International Journal*, 3(3):319–337, August 2001
26. A. Skowron and J. Grzymala-Busse. From rough set theory to evidence theory. In R. R. Yager, M. Fedrizzi, and J. Kacprzyk, editors, *Advances in the Dempster-Shafer Theory of Evidence*, pages 193–236. Wiley, New York, NY, 1994
27. P. Smets. Constructing the pignistic probability function in a context of uncertainty. In M. Henrion, R. D. Shachter, L. N. Kanal, and J. F. Lemmer, editors, *Proceedings of Conference on Uncertainty in Artificial Intelligence (UAI'89)*, pages 29–40. North Holland, 1989
28. P. Vannoorenberghe. On aggregating belief decision trees. *Information Fusion*, 5(3):179–188, September 2004

29. H. Xiaohua. Using rough sets theory and databases operations to construct a good ensemble of classifiers for data mining applications. In *Proceedings of IEEE International Conference on Data Mining (ICDM'01)*, pages 233–240, San Jose, CA, November/December 2001
30. Y. Yang and T. C. Chiam. Rule discovery based on rough set theory. In *Proceedings of International Conference on Information Fusion (ICIF'00)*, volume 1, pages TUC4/11–TUC4/16, Paris, France, July 2000
31. J. Zhang, S. P. Subasingha, K. Premaratne, M.-L. Shyu, M. Kubat, and K. K. R. G. K. Hewawasam. A novel belief theoretic association rule mining based classifier for handling class label ambiguities. In *the Third Workshop in Foundations of Data Mining (FDM'04)*, in conjunction with the Fourth IEEE International Conference on Data Mining (ICDM04), pp. 213–222, November 1, 2004, Birghton, UK