uOttawa

L'Université canadienne
Canada's university

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTOCTORALES

u Ottawa

L'Université canadienne
Canada's university

FACULTY OF GRADUATE AND POSDOCTORAL STUDIES

Hongyu GUO

AUTEUR DE LA THÈSE / AUTHOR OF THESIS

Ph.D. (Computer Science)

GRADE / DEGREE

School of Information Technology and Engineering

FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

Learning from Multirelational Data through Multiple Views

TITRE DE LA THÈSE / TITLE OF THESIS

Herna Viktor

DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

EXAMINATEURS (EXAMINATRICES) DE LA THÈSE / THESIS EXAMINERS

Charles Ling

Éric Paquet

Nathalie Japkowicz

Peter Liu

Gary W. Slater

Le Doyen de la Faculté des études supérieures et postdoctorales / Dean of the Faculty of Graduate and Postdoctoral Studies

Learning from Multirelational Data through Multiple Views

A Dissertation Presented

by

HONGYU GUO

B.Eng. Shanghai Jiao Tong University
M.Sc. University of Ottawa

Submitted to the Faculty of Graduate and Postdoctoral Studies
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in Department of Computer Science

University of Ottawa, Canada
January 2008

Your file Votre référence
ISBN: 978-0-494-41632-7
Our file Notre référence
ISBN: 978-0-494-41632-7

Abstract

LEARNING FROM MULTIRELATIONAL DATA THROUGH MULTIPLE VIEWS

HONGYU GUO

COMPUTER SCIENCE, UNIVERSITY OF OTTAWA

Directed by: Professor Herna L. Viktor

Since their first release in the 1970s, relational databases have been routinely used to collect and organize real-world data—from financial transactions, marketing surveys, to health informatics observations. Traditional data mining methods expect data in the form of a single table, thus resulting in an inability to deal with such relational repositories. Multirelational Data Mining, on the other hand, aims to discover useful patterns across multiple inter-connected relations in a relational database. To this end, this work focuses on how to build classification models for relational databases through multiple views (feature sets).

This study developed four multiple view strategies for mining multirelational data. The thesis firstly introduces the Multi-View Relational Classification (MRC) framework, for constructing hypotheses from sets of attributes of the presented data. The MRC strategy distinguishes itself from existing multirelational mining algorithms by excluding the need to either transform multiple relations into a universal single table or to devise new techniques for direct relational learning. The MRC algorithm offers both predictive performance and efficiency gains over current relational models, when mining diverse relational databases.

Secondly, the MRC-IM method extends the MRC approach in order to deal with skew-class multirelational data. Here, the number of examples from one class is much higher than the others and correctly classifying the underrepresented examples is of prime importance. The MRC-IM method offers performance gains over

current relational model not only against majority class instances, but also against underrepresented examples. While the MRC and MRC-IM methods construct an individual view using features within a sole relation, the third multi-view strategy formulated by this work, namely the MRC-Cross approach, enables the search and collection of relevant attributes across multiple relations when constructing individual views. Finally, we present the SESP technique for pre-pruning uninteresting relations of complex relational databases. Through identifying uninteresting views from the MRC framework, our SESP method creates a pruned structure, while minimizing predictive performance loss on the final classification model.

The results of this study thus suggest that learning from multiple views sets a new direction for efficiently mining data in many relational forms, including relational databases, graphs and social networks.

Acknowledgements

三人行,必有我師焉.
I can always be certain of learning from those I am with.
-Confucius (551-479 B.C. China)

My time at uOttawa has been influenced and guided by a number of people to whom I am deeply indebted.

First, I would like to express my deep gratitude to my advisor, Herna L. Viktor. I cannot thank her enough for introducing me to the exciting world of research. Working with Herna has always been a delightful, rewarding experience. She had been a tremendous mentor, collaborator, and friend, providing me with invaluable insights about research, teaching, writing, presentation, and academic skills in general. I feel exceedingly privileged to have had her guidance and I owe her a great many heartfelt thanks.

I would also like to thank the other members of my committee: Nathalie Japkowicz, Eric Paquet, and Peter X. Liu for their insightful comments and advice. Their helpful comments and technical guidance facilitated the process. Exposure to their supplementary views helped to broaden and deepen my sense of research and academia. I feel most fortunate to have had the opportunity to receive their support. In addition, my deep thanks also go to Charles X. Ling for serving as an external examiner of my thesis.

I also owe a great deal of thanks to my many collaborations and friends in the Text Analysis and Machine Learning Group (TAMALE) and the Intelligent Decision and Data Analysis Laboratory (IDeAL) at the uOttawa. They have been great to knock ground ideas with, to learn from, as well as being good friends. I would also like to thank the many people in my department, support staff and faculty, for always being helpful over the years.

My deepest gratitude and appreciation is reserved for my parents and my wife. Without their constant love, support, and encouragement to keep my feet firmly on the ground, I would never have been able to produce this thesis. I dedicate this thesis to them.

Contents

ABSTRACT iii

ACKNOWLEDGEMENTS v

LIST OF TABLES viii

LIST OF FIGURES x

CHAPTER 1

1 Introduction **2**

1.1 Data Mining 3

1.2 Mining Structured Data 5

1.3 Multirelational Data Mining 7

1.3.1 Relational Database 8

1.3.2 Multirelational Classification 9

1.4 Motivation 10

1.4.1 Weaknesses of Existing Approaches 10

1.4.2 Learning from Multiple Views 11

1.5 Thesis Focus 13

1.6 Contributions 15

1.7 Thesis Outline 16

2 Mining Multirelational Data **20**

2.1 Multi-view Learning 20

2.2 Multirelational Data Mining 25

2.2.1 "Upgrading" Approach 25

2.2.2 Propositionalization Strategy 35
2.2.3 "Upgrading" versus "Flattening" Strategies 42
2.2.4 Learning Structured Data in Other Forms 45
2.2.5 Some Other Studies on Relational Learning 48
2.3 Summary 49

3 Multiple View Multirelational Classification 52
3.1 The Multiple View Approach 53
3.2 Learning from Individual Views 55
3.2.1 Information Propagation 55
3.2.2 Aggregation-based Feature Generation 59
3.2.3 View Learner Construction 64
3.3 Multiple View Combination 64
3.3.1 View Validation 64
3.3.2 View Combination 78
3.3.3 Summary 80
3.4 Experimental Study 81
3.4.1 Methodology 82
3.4.2 The Data Sets Used for Experimental Comparison 83
3.4.3 Experimental Results 85
3.5 Conclusions 95

4 Learning from Imbalanced Multirelational Data 97
4.1 Introduction 98
4.1.1 Related Work 101
4.1.2 Measurement Metrics 104
4.2 The MRC-IM Approach 105
4.2.1 View Validation 105
4.2.2 View Combination 109
4.3 Experimental Study 113
4.3.1 Methodology 113
4.3.2 Databases Used 114
4.3.3 Experimental Results 115
4.3.4 Discussion 126
4.4 Summary 129

5 Constructing Views across Relations **131**
5.1 Motivation . . . 132
5.2 Preliminary Concept . . . 132
5.3 The MRC-Cross Algorithm . . . 136
5.3.1 View Construction of the MRC-Cross Algorithm . . . 137
5.4 Experimental Study . . . 145
5.4.1 Experiments against Benchmarking Databases . . . 145
5.4.2 Experiments against Synthetic Databases . . . 154
5.5 Conclusions . . . 157

6 Pre-pruning Relations of Multi-relational Databases **160**
6.1 Overview . . . 161
6.1.1 Problem Setting and Example Task . . . 162
6.2 The SESP Pruning Approach . . . 164
6.2.1 Subgraph Construction . . . 164
6.2.2 Subgraph Evaluation . . . 166
6.3 Experimental Studies . . . 171
6.3.1 Benchmarking Databases . . . 171
6.3.2 Synthetic Databases with Large Numbers of Relations . . . 174
6.4 Conclusion . . . 178

7 Conclusion and Future Work **181**
7.1 Summary . . . 182
7.2 Contributions . . . 182
7.3 Future Work . . . 183
7.4 Conclusion . . . 185

A ER Diagrams and Specifications for Databases Used **187**

B Publications **190**

List of Tables

1.1 A flat file data source 4

2.1 An sample ILP problem 27
2.2 An sample ILP problem for TILDE 31
2.3 An example of information propagate in CrossMine 32
2.4 Training examples and background knowledge for LINUS 36
2.5 Flat file after being flattened by LINUS 36
2.6 Aggregate tuples using *Max, Min,* and *Average* from SQL 39
2.7 Comparison summary of "flattening" and "upgrading" methods . . 42

3.1 Summary of the data sets used 83
3.2 Accuracies obtained using C4.5 as view learners and meta learners . 86
3.3 Running time required using C4.5 as view learners and meta learners 86
3.4 Accuracies obtained using C4.5 and Naive Bayes 89
3.5 Running time required using C4.5 and Naive Bayes 89
3.6 Accuracies obtained using Naive Bayes and C4.5 91
3.7 Running time required using Naive Bayes and C4.5 91

4.1 Confusion Matrix 104
4.2 Summary of the data sets used 114
4.3 AUCs obtained using C4.5 119
4.4 AUCs obtained using Naive Bayes 121

6.1 Accuracy obtained against the real-world databases 172
6.2 Execution time against the real-world databases 173

A.1 Specification of the FINANCIAL Database 187
A.2 Specification of the MUTAGENISIS Database 188

A.3 Specification of the THROMBOSIS Database 188
A.4 Specification of the WAREHOUSE Database 189

List of Figures

1.1 Mining from graph data 6
1.2 Mining from an example database 7
1.3 A simple database from PKDD 99 8

2.1 Search across multiple relations by FOIL 29
2.2 A logic decision tree constructed by TILDE 31
2.3 Target tuple and its related background tuples 39
2.4 Flattening through join of *Loan* $\bowtie$ *Account* 40
2.5 Final flat file of propositionalization in RelAggs 41
2.6 Finding frequent subgraphs in a molecule database 46

3.1 The MRC framework 53
3.2 A directed foreign key chain 56
3.3 An undirected foreign key chain 58
3.4 Example tuples of the sample database 62
3.5 Propagation of essential information 63
3.6 Aggregation of the *Order* relation 63
3.7 The effects among $\overline{R_{cf}}$, $\overline{R_{ff}}$, and K of the "goodness" formula C . . 70
3.8 Sample view feature set generated by the *Loan* and *Order* views . . 74
3.9 Various learning regions in the target relation 79
3.10 Real data sets from the sample database 79

4.1 Different dimensional projections of feature space 100
4.2 ROC curve and AUC 105
4.3 Detailed performance information of the four view learners 107
4.4 Learning regions in the target relation 111
4.5 A majority voting example 112

4.6 ROC curves for the Mutagenesis database with C4.5 115
4.7 ROC curves for the F400AC database with C4.5 116
4.8 ROC curves for the F234AC database with C4.5 116
4.9 ROC curves for the F682AC database with C4.5 117
4.10 ROC curves for the ECML_I database with C4.5 117
4.11 ROC curves for the ECML_II database with C4.5 118
4.12 ROC curves for the Mutagenesis database with Naive Bayes 122
4.13 ROC curves for the F400AC database with Naive Bayes 123
4.14 ROC curves for the F234AC database with Naive Bayes 123
4.15 ROC curves for the F682AC database with Naive Bayes 124
4.16 ROC curves for the ECML_I database with Naive Bayes 124
4.17 ROC curves for the ECML_II database with Naive Bayes 125
4.18 ROC which would be achieved by hypo-ROC model combination . . 127
4.19 Sampling in related relations 128

5.1 An example join path 133
5.2 An example undirected graph 139
5.3 Travel the undirected graph 141
5.4 All subgraphs constructed from the example database 142
5.5 Sample data after generating relational features for Join Path 1 . . 143
5.6 Accuracies obtained using C4.5 as view learners and meta learners . 147
5.7 Running time required using C4.5 as view learners and meta learners 147
5.8 Accuracies obtained using C4.5 and Naive Bayes 149
5.9 Running time required using C4.5 and Naive Bayes 149
5.10 Accuracies obtained using Naive Bayes and C4.5 151
5.11 Running time required using Naive Bayes and C4.5 152
5.12 Dominating features in induced decision trees 153
5.13 Maximum length of Join Path 155
5.14 Performance against the synthetic databases 156

6.1 A sample database 163
6.2 The core idea of the SESP approach 164
6.3 Search and construct subgraphs 166
6.4 Sample subgraph feature set generated by the six subgraphs 168
6.5 Identified subgraph feature set 169

6.6 Subgraph evaluation and pruning 170
6.7 The original and pruned structure of the synthetic databases 175
6.8 The pruned schemas of databases SynR80, SynR100, and SynR150 . 175
6.9 Accuracy obtained using the MRC method 176
6.10 Accuracy obtained using the CrossMine method 176
6.11 Compression rate achieved using the SESP method 177
6.12 Running time required when constructing the relational models . . 177

A.1 The FINANCIAL Database 187
A.2 The MUTAGENISIS Database 188
A.3 The THROMBOSIS Database 188
A.4 The WAREHOUSE Database 189

CHAPTER 1
INTRODUCTION

Chapter 1

Introduction

学而不思则罔，思而不学则怠。

To learn without thinking is labor in vain, to think without learning is desolation

-Confucius (551-479 B.C. China)

The rapid accumulation of huge amounts of real-world data in relational databases has challenged knowledge discovery modelers. Over the past decades, data mining—the process of automatically looking for useful patterns from large collections of data—has been playing an important role on acquiring useful knowledge out of data. Unfortunately, most existing data mining algorithms operate on a single relation, while most of today's structured data—from financial transactions, marketing surveys, medical records, to health informatics observations—are routinely collected into, and organized in, multiple interconnected tables of a relational database. Hence, traditional data mining methods have shown their inability to deal with such relational repositories.

In addition, existing methods which are capable of learning from relational domains are facing severe efficiency and scalability challenges due to the large hypothesis space in relational data. Class categorizing is simpler if more useful attribute sets (views) are *properly* used in the classification systems. For example, separating

dogs from cats would be easier if one can make use of both auditory information and visual features. However, the large feature space in a relational repository is distributed in multiple interlinked relations. Exploiting useful features, therefore, often involves an exploration of different joins among various relations. Thus, such a process is extremely expensive, in terms of computation time, when dealing with a complex relational schema.

Facing the above mentioned challenges, the development of new techniques which can efficiently learn from relational repositories has become a field of strategic importance. The research presented here introduces a framework which can *directly* learn from a relational database *efficiently*.

This chapter provides the motivations and contributions of the thesis. The chapter is organized as follows. It first provides a concise introduction to data mining, along with a description of the problems to be addressed in the thesis: mining from relational databases. Next, it will present the key motivational aspects and areas of contribution.

1.1 Data Mining

Data mining is the process of automatically looking for useful patterns in large data repositories [102]. With the rapid advances in data collection and storage technologies, vast amounts of real-world data have been accumulated. To handle the massive amount of data available, data mining blends data process and analysis techniques from the fields of statistics, information retrieval, machine learning, and pattern recognition, amongst others. In fact, data mining techniques have been playing a key role in many data-driven decisions, such as credit card fraud detection, loan application, medical syndrome differentiation, and gene prediction.

Typical data mining tasks include classification, clustering, and association rule mining [232]. Classification [68] is a classic data mining task where classifiers are built from given examples. The constructed classifiers are then used to assign unseen instances into various categories. Clustering [104] tasks learn models in

order to group instances into clusters, where the association between members in the same cluster is strong and the association between different cluster members is weak. While classification and clustering systems aim to build predictive models to categorize unseen samples, association rules mining [102] identifies collections of features of the data that are statistically related.

Often, a data mining task is given a data set, i.e. training data set or training examples. Nevertheless, this data set can be stored in a variety of formats. Traditionally, data mining techniques rely on an important assumption: all data resides in a flat file or a single table. This single table consists of a set of individuals, i.e. instances or examples. Each instance is represented by a fixed number of attributes (features) for which values are given. Data mining methods use this single table or flat file as input to search for useful patterns. This type of data mining tasks is referred to as propositional data mining, single table data mining, or traditional data mining.

Table 1.1: A flat file data source

loan-id	account-id	amount	duration	payment	status
001	132	2000	12	100	good
002	132	4000	12	200	good
003	148	1000	24	80	bad
004	148	30000	24	1000	bad
005	98	10000	36	800	good

Example 1.1-1. {Data source in the form of flat file} Consider the *loan* information of customers in a bank. Each *loan* can be described by features such as *account-id, date, amount, duration,* and *payment.* Table 1.1 provides an example flat file (single table) data in a tabular form. Each tuple (record) in this table is considered as an example, which uses a number of attribute (feature) values to describe a *loan.* For example, the first tuple in Table 1.1 represents a *loan* record with *loan-id = 001, account-id = 132, amount = 2000, duration = 12, payment = 100,* and *status = good.* Traditional data mining algorithms, for example, in-

put the combination of these feature values to construct a classification model to distinguish *"bad" loans* from *"good" loans.*

Over the past decades, propositional learning techniques have been extensively investigated. Taking advantages of the simple structure of a flat file, a wide range of propositional data mining algorithms, such as decision trees [196], Support Vector Machines (SVMs) [30], and Artificial Neural Networks [15] have been developed. Many of these algorithms are used in commercial applications. In contrast to learning from a single file, there has recently been a surge of interest in the development of data mining algorithms for various types of structured data, as discussed next.

1.2 Mining Structured Data

Most of today's real-world data is structured: such data has no natural representation in a single tabular table and instances in these data are more naturally represented by structured terms than fixed-length feature vectors [84]. For example, molecules are naturally described through two or three dimensional structures of atoms and thus have graph structures; XML data could be naturally mapped on tree structures; images are spatially structured; and financial transactions are effectively managed using multiple tables in a relational database where the structures of such data are encoded in foreign key joins of the database.

Learning from structured data needs to take the information encoded in the structure of the data into account, since such structure presents how different objects in the data relate to one another and may demonstrate some useful patterns in mining tasks. Popular structured data mining applications include mining from data in the form of graphs [44, 97, 119, 143, 237], social networks [60, 212, 228], links [54, 88, 142, 245, 247], and relational databases [71, 89, 118, 219, 242].

Example 1.2-1 (Data source in the form of graph). Figure 1.1 provides an example of learning from structured data in the form of a graph (adopted from [239]). This figure depicts three training examples from a chemical database (left-hand

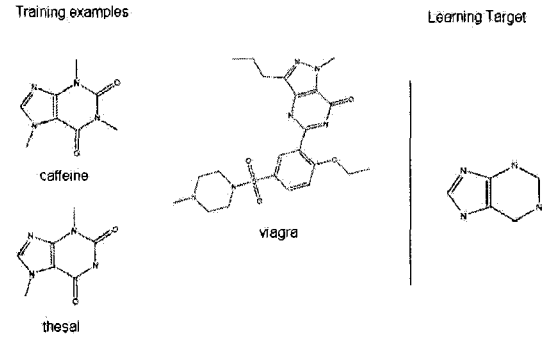


Figure 1.1: Mining from graph data [239]

side of Figure 1.1), as well as the subgraph pattern to learn (right-hand side of Figure 1.1). These three training examples describe the structures of three molecules, namely Caffeine, Thesal, and Viagra. The learning task here is to predict if a molecule contains the specific subgraph depicted in the right-hand side of the figure. In this learning setting, the sizes and shapes of molecules are naturally represented in two-dimensional space. Such graphs allow for a direct physical picture and interpretation of how atoms are held together through chemical bonds. In contrast, describing such structures with fix-length feature vectors would be difficult and unintuitive because of the variety of atoms and their complex chemistry bond connections.

Although there are many forms of structured data to be learned, recently, there has been a surge of interest in mining relational databases. This is due to their popularity as one of the chief repositories for real-world data.

Example 1.2-2 (Data source in the form of relational database). Consider the example database depicted in Figure 1.2 (adopted from [13]). As shown in Figure 1.2, the information in this data set is distributed into two parts, i.e. the *Loan* table and *Account* table. These two relations are linked through the *account-id* feature. In a typical data mining task, data mining algorithms aim to use information from both parts and the link between these two tables to search for certain patterns. Therefore, new methods to directly mine these repositories, while taking this inter-

Account Table

account-id	frequency	balance
00132	monthly	3521
00132	weekly	2000
00240	monthly	20
00100	weekly	50
0098	weekly	10000

Loan Table

loan-id	account-id	amount	duration	payment	status
001	00132	2000	12	100	Good
002	00132	4000	12	200	Good
003	00148	1000	24	80	Bad
004	00148	30000	24	1000	Bad
005	0098	10000	36	800	Good

Figure 1.2: Mining from an example database [13]

relationship into account, should prove worthwhile. This is where multirelational data mining comes to play.

In the next section, we will provide a detailed introduction to multirelational data mining, along with the related issues which need to be addressed.

1.3 Multirelational Data Mining

Multirelational Data Mining (MRDM) aims to discover useful patterns across multiple relations (tables) in a relational database. In general, a relational database repository is represented by its multiple relations, which are inter-connected by means of foreign key joins. Since their first release in the 1970s, commercial relational databases have become one of the most widely used repositories for real-world data. Many real-world data—from financial transactions, marketing surveys, medical records, to health informatics observations—are routinely collected and organized in relational databases. Such data offer unique opportunities to data miners and potentially contain precious knowledge to data-driven decision makers. The accumulation of the massive amount of such data results in an urgent search for knowledge discovery techniques which can efficiently exploit the multiple interconnected relations residing in a relational database.

1.3.1 Relational Database

Multirelational data mining takes a relational database as input. A schema for a relational database $\Re$ describes a set of tables $\Re = \{T_i\}_1^n$ and a set of relationships between pairs of tables. A table T_i consists of a set of tuples, a primary key (denoted by $T_{.key}$), and a set of foreign keys (in this paper, we refer to primary key and foreign keys as *key attributes*); the other attributes are *descriptive attributes.* Foreign key attributes[1] link to the primary keys of other tables: this link specifies a *join* between two tables, and a foreign key attribute of table T_2 that references table T_1 is denoted as $T_2.T_{1\_key}$.

Figure 1.3 gives a simple example of a relational database schema with eight tables [13]. All relationships among tables are represented with lines. The lines show the foreign key links, which relate the key attributes of one table to the corresponding key attributes of another table. For example, the *Account* table has a foreign key attribute *account-id*, which is linked to the key attribute *account-id* in the *Loan* table.

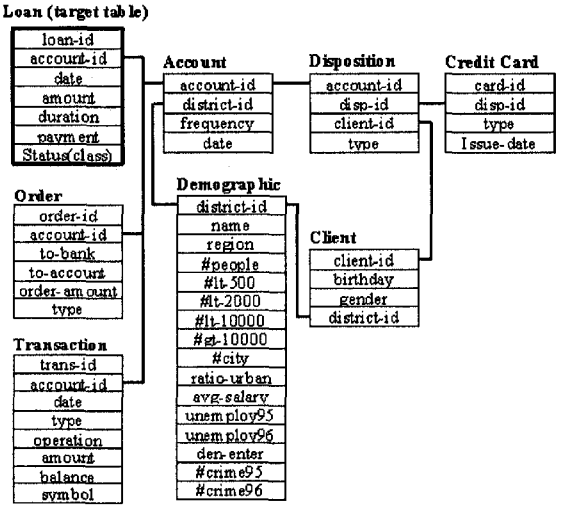


Figure 1.3: A simple database from PKDD 99 [13]

[1]For simplicity, we only consider single attribute keys here. It follows that this definition generalizes to multi-attribute keys.

1.3.2 Multirelational Classification

Using a relational database as input, multirelational data mining approaches can be initiated to search for patterns. In a multirelational classification setting, there is a database $\Re$, which consists of a target table T_{target} and a set of background relations $\{T_i\}_1^n$. In addition, the target relation T_{target} has a target variable Y. That is, each tuple in this table (target tuple) is associated with a class label which belongs to Y. Typically, the relational classification task is to find a function $F(x)$ which maps each tuple x from the target table T_{target} to the category Y:

$$Y = F(x, T_{1\cdots n}, T_{target}), x \in T_{target} \tag{1.1}$$

In the above scenario, the T_{target} table is considered the target relation while all others are background relations. Consider a two-class problem (e.g. positive and negative). The task of multirelational classification is to identify relevant information (features) across different relations (i.e. both from T_{target} and $\{T_i\}_1^n$) to separate the positive and negative tuples of the target relation T_{target}.

Example 1.3-1 (Multirelational classification in a relational database). In the example database of Figure 1.3, the *Loan* table is the target table and the target variable is the *status* attribute. This target variable indicates whether a tuple in the target table is *good* or *bad*. It follows that all other tables, i.e. tables *Account, Order, Transaction, Credit Card, Disposition, Client,* and *Demographic* are background relations. All background relations are associated with the target relation by foreign key joins. The problem task is to find relevant features from all these eight tables to distinguish *good* tuples from *bad* tuples in the target relation *Loan*.

This section provided a brief introduction of data mining and described the setting of multirelational data mining. The next section will present our research motivation.

1.4 Motivation

One of the main issues when mining relational domains is the method by which the information captured by the multiple relations is included. That is, research has shown that, when mining relational domains, it is important to consider not only the knowledge from the target relation, but also the information in the relations which relate to the target, the so-called background relations [70, 71, 102, 124, 136, 242, 243]. To tackle this problem, in the past decade, existing approaches *either* "upgrade" conventional learning algorithms to deal with relational presentation (the so-called first-order upgrade methods) *or* "flatten" the multiple relations to a universal single flat file, which is then used as the input to propositional learning methods (the so-called propositionalization approaches). Unfortunately, both of these strategies have demonstrated some shortcomings, resulting in certain barriers which have prevented a wider application thereof.

1.4.1 Weaknesses of Existing Approaches

Drawbacks have been found in the existing "upgrade" methods. These approaches, especially those using Inductive Logic Programming (ILP) techniques, usually suffer from poor scalability when dealing with complex database schemas. ILP techniques tend to be time-consuming when building hypotheses. Thus, data sets used together with ILP methods are typically small [21]. In addition, using ILP approaches usually requires considerable effort when deciding on the right representation for the training information (such as training examples and background knowledge) and the right trade-off between expressive power of the hypothesis language and efficiency of the learning process (e.g. hypothesis language bias). Research has shown that it is not easy to accomplish these tasks in order to properly start the learning process [136]. Another drawback of these ILP-based approaches is that, due to their reliance on logical expressions, ILP strategies usually give unsatisfactory predictive performance when applied to real-world business applications, which often contain noisy and numeric values [125, 183].

The second category of algorithm, namely propositionalization approaches, relies on transforming multiple relations into a universal single table. Unfortunately, shortcomings resulting from such "flattening" process have also been identified. Firstly, it usually takes considerable time and effort to convert the multiple relations into the required "flat" format. Another drawback of this method is that the need for extensive pre-processing can cause statistical skew [70, 82] in the resulting data. Importantly, such conversion also tends to create a big table with an exponentially large number of additional attributes and a large number of NULL values (missing values). Such a table may thus mislead the propositional learning algorithms and result in further efficiency, overfitting, and scalability issues. In fact, these shortcomings have resulted in the development of many "upgrade" propositional approaches [71].

1.4.2 Learning from Multiple Views

Encountering the above mentioned obstacles with existing strategies, our research aims to design new solutions for multirelational data mining. The developed framework should allow propositional mining techniques to learn from multirelational databases, excluding the need to "flatten" the multiple relations. Using this framework results in two chief benefits. First, we can apply efficient and accurate propositional mining methods to learn from multirelational data, but exclude the need to "re-invent the wheel" (as "upgrade" strategies usually do). Second, we can also avoid the negative consequences that result from "flattening" multiple relations in the propositionalization methods, namely a huge flat file with many additional attributes and missing values. Our task is therefore to answer the following question: How can we employ single table mining algorithms to learn from relational data without "flattening" the tables?

Our methods for multirelational classification are inspired by a promising new strategy: multi-view learning. Multi-view learning has received significant attention in current literature [23, 57, 171]. This framework has been applied success-

fully to many real-world applications such as information extraction [23] and face recognition [57]. As in the example given by Blum and Mitchell [23], one can classify segments of televised broadcast based *either* on the video *or* on the audio information.

In fact, as will be discussed in Sections 2.1 and 3.3.1, a multi-view learning problem with n views may be seen as n *uncorrelated* or *weakly dependent* feature sets [1, 170, 171]. Here, the concept of *correlation* [235] is used to measure the departure of two variables (views) from independence [235]. For example, if we say two views V^x and V^y are correlated, it means knowing V^x helps in predicting V^y; if knowing V^x tells us nothing about V^y, we say they are uncorrelated or their correlation coefficient should be zero. When considering applications within a relational database context, such sets of disjoint features are typically distributed in the multiple relations of a relational database. A relational database $\Re$ normally contains n inter-connected relations. Often, a relational database is designed by domain experts using an Entity Relationship (ER) model, where each different relation usually groups a set of features with very close semantic meaning [72, 83, 203, 243]. In other words, each relation in a database usually has a naturally divided disjoint feature set, providing various attributes (information) contributing to the target concepts to be learned. As an example, consider the loan problem, as shown in Figure 1.3, from the PKDD 99 discovery challenge [13]. Here a banking database with eight relations is depicted. Each relation describes the different characteristics of a client. For example, the *Client* relation contains a customer's age, but the *Account* relation identifies a customer's banking account information. That is to say, each relation from this database provides different types of information, or views, for the concept to be learned, i.e. whether the customer is a *risk* or *not*. Thus, one can explore a relational database to construct multiple *disjoint* or highly *uncorrelated* feature sets from related relations. This problem is therefore a perfect candidate for multi-view learning.

1.5 Thesis Focus

This thesis focuses on how to build classification models for relational databases through multiple views (feature sets), moving beyond the traditional ways of obtaining hypotheses where each hypothesis is usually built using all features available in the data set. We aim to investigate the benefits of using various sets of features when dealing with multirelational data, and to develop strategies which construct hypotheses based on different observations, each yielded from one distinct view of the presented data.

First, we present the so-called Multiple View Relational Classification (MRC) framework, which learns, using conventional learning techniques, from multiple views of a relational database. Here, each view is constructed from one of the relations of the database. Next, knowledge acquired from individual views is then integrated to augment one another and to form the final classification model. This method distinguishes itself from existing multirelational mining algorithms by excluding the need to *either* transform multiple relations into a universal single table *or* to devise new techniques for direct relational learning. That is, existing "traditional" data mining technique may be employed within the MRC framework. The MRC algorithm offers both predictive performance and efficiency gains over current relational models, when mining diverse relational databases.

Second, many multirelational data mining methods implicitly assume that the classes to be learned are equally represented. Unfortunately, target tuples in many practical database applications, such as credit card fraud detection and disease diagnosis, are highly imbalanced, where the number of examples of one class in the target relation is much higher than the others. Often, correctly classifying the minority examples is of importance. In order to tackle the skew-class problem, which has hindered many real-world knowledge discovery applications in commercial databases, we devise the MRC-IM algorithm through extending the multiple view framework. Techniques which are more suitable for imbalanced class learning are devised for both the view validation and view combination components of the

MRC-IM algorithm. The MRC-IM method offers performance gains over current relational model not only against majority class instances, but also against under-represented examples. In particular, this study shows that the MRC-IM algorithm can achieve superior predictive results over popular relational learning methods, even when dealing with highly skewed data.

In addition, we explore an alternative view construction strategy for the multi-view learning framework. We extend the MRC method's view construction process by enabling views to search and collect relevant features across multiple relations. This MRC-Cross method constructs views using features across relations of a relational database. We examine the benefits of such view construction process. In particular, we show that constructing individual views across multiple relations achieves little performance gains on the final model over that of generating individual views from sole relation. This result indicates that, when mining real-world commercial relational databases, one does not need to combine features from different tables.

Finally, through identifying uninteresting views in the MRC framework, we have also developed a novel approach for pre-pruning uninteresting relations of a relational database. Multi-relational data mining application usually involves a large number of relations, where each may come from a different party. Acquiring such data is often expensive, in terms of data mining overheads such as cost of acquiring the data, preprocessing, building the model, computational overhead, and human effort. These problems can be mitigated by pruning uninteresting relations before constructing a relational model. By extending the MRC framework, we have developed the Subgraph Ensemble Structure Pruning (SESP) method for pre-pruning multi-relational databases. Our method creates a pruned structure while minimizing predictive performance loss on the final classification model. The study presented here also indicates that, when learning from a relational database, one can build an accurate relational classification model using only a small subset of the original database schema.

1.6 Contributions

In this work, we study existing approaches that perform multirelational data mining. We aim to identify their weaknesses and design new solutions to address them.

The primary contributions of this thesis are two-fold. First, the thesis illustrates the benefits of using sets of features when dealing with multirelational data. We incorporate the idea of learning from multiple views into relational domains. Our study implies that learning through multiple views sheds light on making good use of the rich feature space presented in structured domains.

Second, we develop four multiple view methods for learning from multirelational databases, namely MRC, MRC-IM, MRC-Cross, and SESP. The MRC strategy is capable of learning from multiple views of a relational database through the use of existing "traditional" data mining methods. It excludes the need of transforming multiple relations into a universal single table. The MRC algorithm offers both predictive performance and efficiency gains over current relational models, when mining diverse relational databases.

In addition to the MRC algorithm, we also devise a multiple view modeling technique for skewed class multirelational data, an important problem which hinders many knowledge discovery applications of commercial databases. The MRC-IM method offers performance gains over current relational model not only against majority class instances, but also against underrepresented examples. In particular, the MRC-IM approach can achieve very promising results even when dealing with highly skewed data.

Besides the MRC and MRC-IM algorithms, we devise the MRC-Cross approach. This method enables the search and collection of relevant attributes across multiple relations while constructing each individual view. Within this strategy, we also explore the effects of different view construction methods on the performance of the multiple view learning framework.

Finally, by extending the MRC framework, we develop the SESP strategy for

pre-pruning uninteresting substructures of a relational database. As a consequence of such a pre-pruning process, one can reduce the cost of acquiring and managing training data, enhance the data privacy preservation, and improve the scalability of the mining process, when learning from relational databases.

In summary, our MRC and MRC-Cross methods are the first to incorporate the idea of learning from multiple views into relational domains. Also, our MRC-IM strategy is the first algorithm available for dealing with imbalanced class problems in relational data. Moreover, our SESP approach sets a new direction for pre-pruning uninteresting substructures of relational databases for multi-relational classification tasks.

1.7 Thesis Outline

The remainder of this thesis is organized as follows. Chapter 2 provides related work, both in the fields of multirelational classification and multi-view learning. Next, a detailed discussion of the MRC framework is presented in Chapter 3. In Chapter 4, we describe a multiple view approach, called MRC-IM, which aims to learn from imbalanced multirelational data. Chapter 5 introduces the MRC-Cross algorithm. The MRC-Cross method is an extension of the MRC strategy that enables the MRC algorithm to search for relevant features across relations. The SESP method for pre-pruning uninteresting substructures of a relational database is then presented in Chapter 5. Finally, Chapter 6 summarizes the thesis.

Some of this material has appeared previously in workshop, conference, and journal papers. The work on the MRC framework presented in Chapter 3 first published in the papers:

- Hongyu Guo and Herna L. Viktor, Mining Relational Databases with Multi-view Learning, *in Proc. the 4th Multi-Relational Data Mining Workshop (KDD/MRDM'05)*, in conjunction with the Eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2005),

Chicago, IL, Aug. 2005. Pages 15-24. ACM Press.

- Hongyu Guo and Herna L. Viktor, Multirelational Classification: A Multiple View Approach, *Knowledge and Information Systems: An International Journal*, Springer Publlishers, 2008.

The applications of the MRC method appeared in the following publications:

- Herna L. Viktor, Eric Paquet and Hongyu Guo, Measuring to Fit: Virtual Tailoring through Cluster Analysis and Classification, *in Proc. 10th European Conf. on Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD 2006)*, Berlin, Germany, September 2006. Pages 395-406.

- Hongyu Guo and Herna L. Viktor, Multi-view Artificial Neural Networks for Multi-relational Classification, *in Proc. IEEE International Joint Conference on Neural Networks (IJCNN 2006)*, Vancouver, BC, July. 2006. Pages 5259-5266. IEEE Press.

- Eric Paquet, Herna L. Viktor, Hongyu Guo, and Isis Pena Sanchez, Constrained Virtual Tailoring from Anthropometric Data, 3-D Shape and Data Mining, *in Proc. of the Second International WEAR Conference (WEAR'02)*, Alberta, Canada, Aug. 2007.

The MRC-IM approach introduced in Chapter 4 was previously published in the paper:

- Hongyu Guo and Herna L. Viktor, Mining Imbalanced Classes in Multirelational Classification, *in Proc. of the 6th Multi-Relational Data Mining Workshop (PKDD/MRDM'07)*, in conjunction with 11th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'07), Warsaw, Poland, September 2007. Pages 46-57.

The work on the MRC-Cross method as discussed in Chapter 5 first appeared in the paper:

- Hongyu Guo and Herna L. Viktor, Mining Relational Data through Correlation-based Multiple View Validation, *in Proc. the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2006)*, Philadelphia, PA, August 2006. Pages 567-573.

The work on the SESP strategy described in Chapter 6 was previously published in the paper:

- Hongyu Guo, Herna L. Viktor, and Eric Paquet, Pruning Relations for Substructure Discovery of Multi-relational Databases, *in Proc. 11th European Conference on Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD 2007)*, Warsaw, Poland, September 2007. Pages 462-470.

CHAPTER 2

MINING MULTIRELATIONAL DATA

Chapter 2

Mining Multirelational Data

Recall that multirelational data mining algorithms search for useful patterns within relational databases. Over the past few years, there has been a surge of interest in mining multirelational data. This is due to the fact that, nowadays, commercial relational databases abound, and, subsequently, such relational repositories store most of today's real-world structured data.

Our research aims to design new solutions for multirelational data mining. In this chapter, the work related to this thesis is introduced. The framework presented here was inspired by multi-view learning and concerns multirelational data mining, thus the purpose of this introduction is twofold. We will first present the basic idea of learning from multiple views and follow that with a review of related work for multirelational data mining.

2.1 Multi-view Learning

Our research is inspired by a promising new learning strategy: multi-view learning. Multi-view learning learns from multiple independent views (feature sets) of the presented data [23, 55, 171, 241]. For example, one can separate dogs from cats based *either* on the auditory *or* on the visual information. This non-standard method draws significant attention in current literature by several related

fields including information extraction [23], text classification [42, 91], face recognition [57], protein analysis [138], and email classification [122, 127, 127], amongst others [49, 169, 171, 186, 209].

Multi-view learning is often applied to applications where features can be naturally divided into multiple subsets, any of which is sufficient to approximate the hypothesis function [23, 55]. Consider a multi-view learning setting with three views (V_1, V_2, and V_3). In this scenario, any instance x can be viewed as

$$\vec{x} = \langle x^1, x^2, x^3, y \rangle$$

where x^1, x^2, and x^3 are its descriptions in the three views V_1, V_2, and V_3, respectively. The variable y denotes the class label. Three view learners f^1, f^2, and f^3 are constructed using features from views V_1, V_2, and V_3, respectively.

In recent years, there has been a surge of interest in multi-view learning. Early studies on multi-view learning focus on semi-supervised classification problems, where one is given a small set of labeled data and a large set of unlabeled data. de Sa presented the idea of minimizing disagreement algorithm [55, 57]. That is, minimizing the disagreement of multiple hypotheses (built from different views) on unlabeled data is approximation to minimizing the error rate of the final classifier directly. Yarowsky's rule-based bootstrapping algorithm [241] for word sense disambiguation also falls in the context of multi-view learning. This method combines two classifiers: one is built using the local context of a word and another is trained by the documents where the words co-occur. Blum and Mitchell introduced the co-training strategy [23]. The idea of this approach is to learn two classifiers from disjoint views of the training examples. The two view learners then bootstrap one another by providing one another with labels for the unlabeled instances. This process iterates until some stopping criterion is met.

Continuing in the same semi-supervised classification setting, studies which adopt popular learning approaches to make use of multiple views of the presented data have recently been introduced. Nigam and Ghani presented the co-

EM method [176], which integrates multi-view learning with the probabilistic EM strategy [62]. The EM algorithm is an statistical technique for maximum likelihood estimation in applications with incomplete data [62]. Through iterations, the EM algorithm can locally maximize the likelihood of the parameters of the data generation model and give estimates for the missing values. In the co-EM scenario, the class labels of the unlabeled data are treated as the missing values in the EM algorithm. The co-EM approach probabilistically labels all unlabeled instances and iteratively exchanges those labels between views [91]. While Nigam and Ghani's method requires the learning algorithm be able to deal with probabilistically labeled data and output class probabilities, Brefeld and Scheffer developed a co-EM version of the Support Vector Machine (SVM) [26]. Following the same line of research, Collins and Singer [42] presented a strategy called CoBoost by adopting the boosting algorithm [81] in order to take advantage of multiple feature sets. The CoBoost approach iteratively constructs two classifiers: each is trained using either the spelling of the named entity or the context in which that entity occurs. In each iteration, the CoBoost method aims to minimize a disagree bound of the two classifiers, in terms of the number of unlabeled instances. More recently, Brefeld et al. [25] deployed the idea of multi-view learning into hidden Markov algorithms for discriminative sequential learning.

In addition to multi-view learning strategies studied under the context of classification, a number of multi-view clustering algorithms have also been proposed to deal with domains in which the available features can be divided into disjoint subsets. Bickel and Scheffer introduced a multi-view clustering algorithm [14], and successful applied it to text data. In [14], the authors also empirically observed that the multi-view versions of the k-means and EM algorithms greatly outperform their single-view counterparts. In their approach, the assignment of an example to a specific cluster is determined by the posterior over all views. Basing on the idea of the minimizing disagreement algorithm [55, 57], de Sa has also presented a spectral clustering method [56]. In addition, Gupta and Dasgupta combined multiple views to form a hybrid hierarchical clustering strategy [99]. Other related

multi-view clustering algorithms include [226], in which reinforcement learning is applied, and [116], which is considered as a multi-view version of the DBSCAN method [75].

Taking advantages of learning from multiple feature sets, the multi-view learning framework has also been deployed [42, 92, 170, 171] to enhance the active learning strategy [222]. In the active learning setting, learner gathers information about the learning domain by asking questions and receiving responses. Under a multi-view framework, multiple learners are first constructed, each learning from an independent subset of features. Unlabeled examples are then selected for the active learning, based on the disagreements of the trained multiple view learners [92]. For example, Muslea [171] presented a multi-view active learning algorithm called Co-Testing, in order to make use of multiple feature sets. In this method, each view leaner learns first from provided labeled data. Next, the most informative unlabeled instances are detected by the multiple view learners and then the user is asked to label them. In this way, one can reduce the user's data labeling burden. Instead of constructing view learners with only labeled instances, Jones et al. [92] proposed a strategy where the co-EM approach [176], active learning, and multi-view framework are combined. In their method, view learners are trained using both labeled and unlabeled data through the use of a bootstrapping algorithm. Following the same trend, Becker et al. [9] introduced an active learning approach to bootstrap a named entity recognition system. In this method, the disagreement of conditional Markov models, built using different feature sets, is used to tell if an unlabeled example is useful.

Some other varieties of multi-view learning strategies have also been investigated. These methods combine multiple learning models constructed using different feature subsets of the presented data. Becker and Hinton [10] maximized mutual information between the output of neural network models that look at separate, but related, parts of the input. Barnard et al. [8] and Blei and Jordan [17] combined multiple view learners separately built from images and their textual annotations. Yao et al. integrated multi-view analysis with granular com-

puting studies [36, 240]. Chen and Yao presented a multi-view framework for data analysis, where views are constructs based on modal-style data operators such as set assignment, dual sufficiency, and possibility operators [37]. Belkhouche and Lemus-Olalde applied multiple views analysis to software design [11, 12]. Zhong and Yao employed a multiple analysis idea to analyze brain informatics data [246]. In addition, multi-view approaches dealing with regression problems have also been studied [117, 158].

While most work on multi-view learning have empirically demonstrated this strategy's usefulness, research which aims to theoretically understand the performance gain of this framework has been studied as well. Dasupta et al. [49] provided a Probably Approximately Correct (PAC) style bound for the generalization error of co-training algorithms. This bound gives a PAC style performance guarantee in terms of the agreement rate of hypotheses in two disjoint views. Their work theoretically justifies the confidence of the use of the co-training strategy. Recent work of Abney [1] has also suggested that the disagreement rate of two independent hypotheses upper-bounds the error rate of individual view learner in a semi-supervised learning setting. In other words, one can minimize the disagreement rate of multiple view learners on unlabeled data, in order to minimize the error of the final induced classifier. This study also justifies Collins and Singer's approach of directly optimizing the agreement rate of learners over different views [42].

Besides studies on error bounds of the multi-view learning strategy, the fundamental assumption of this framework has also been investigated. The theoretical foundations of multi-view learning are based on the assumption that views are *independent* [23]. However, research has shown that in real-world domains, the ideal assumption of multiple strictly independent views is not fully satisfied. As pointed out by Muslea et al. [171], in real-world problems, one seldom encounters applications with independent views. They have also demonstrated that when the independence assumption is violated, namely in problems with correlated views, the multi-view approach can still perform well [170]. Furthermore, Abney [1] also argued that the ideal independence assumption is quite strong, and often violated

in the data. The author has also shown that a weaker assumption (such as views are weakly dependent) suffices for the multi-view learning framework.

In short, over the past few years, we have witnessed a surge of interest in making good use of multiple feature sets of the presented data. In our research, we adopt the idea of learning from multiple views to tackle the problems that arise in the multirelational data mining community. Next, related work for multirelational data mining is presented.

2.2 Multirelational Data Mining

Over the past few years, two main categories of approaches to multirelational data mining have been actively investigated: "upgrade" and "flattening" strategies. Recall from Section 1.4 that, existing "upgrade" methods (or first-order upgrade) usually integrate propositional data mining algorithms with ILP techniques. On the other hand, as introduced in Section 1.4, "flattening" (or propositionalization) strategies transform multiple relations to a universal attribute-value form (flat file), and then directly apply traditional learning methods to learn from the converted flat file.

In the next sections, we will introduce the "upgrade" algorithms and also provide a detailed discussion of propositionalization methods.

2.2.1 "Upgrading" Approach

The first category of algorithms for multirelational domains aims to upgrade propositional methods to handle relational representations. This method extends propositional learners to search features in multiple relations. That is, by designing new relational learning techniques for existing conventional data mining methods, these algorithms can be employed to explore relational data. Existing "upgrade" methods often use techniques from the Inductive Logic Programming (ILP) community to explore information across multiple relations. In fact, ILP is historically the

first strategy to deal with structured data. In addition, over the past 16 years, the most widely used algorithms to multirelational data mining have been ILP-based "upgrade" approaches.

In the remainder of this section, we first provide a concise introduction to ILP systems. Next, four well-known ILP-based algorithms are discussed. Finally, we conclude this section by briefly presenting ILP-related work in the statistical community.

ILP Systems

ILP systems are concerned with looking for useful patterns which are expressed as logic programs. ILP systems often aim to generalize hypotheses from training instances and background knowledge, and the inducted hypotheses are then used to predict unseen instances. Typically, ILP systems are expressed as logic programs, an important subset of first-order logic.

Formally, as introduced by Muggleton [165], an ILP problem is defined as follows:

Definition 2.2-1 (ILP Problem). Given background knowledge B, a set of positive examples P, and a set of negative examples N. The task of the ILP system is to find a hypothesis H such that:

$$B \wedge H \models e, \forall e \in P \tag{2.1}$$

and

$$B \wedge H \not\models e, \forall e \in N \tag{2.2}$$

where $\models$ stands for logical implication. Equation 2.1 describes the completeness of the hypothesis H, and Equation 2.2 describes the consistency of the hypothesis H.

The above definition is usually called *learning from entailment.* An alternative ILP setting (*called learning from interpretations*) is presented by De Raedt and

Dzeroski in [201]. This setting replaces the requirement defined in Equation 2.1 with the requirement of H be true in the minimal Herbrand model of $B \wedge e$.

Practically, e is often given by a tuple, B is a relational database, and H is a *definite logic program clauses.* H is defined through the *hypothesis language*, which is typically a subset of the language of program clauses. Most ILP methods explore, based on a top-down strategy, the hypothesis space of program clauses from general to specific hypotheses. Note that, since the complexity of learning rapidly grows with the expressiveness of the *hypothesis language*, restrictions or *hypotheses language bias* (e.g. exclusion of recursion and restriction of where the new variables can be introduced in the clauses) are introduced to restrict the hypothesized clauses in the hypothesis searching process. These restrictions force a trade-off between the expressiveness and the efficiency of the ILP systems.

Table 2.1: An sample ILP problem [71]

Training examples		Background knowledge	
daughter(mary,ann).	+	parent(ann,mary).	female(ann).
daughter(eve,tom).	+	parent(ann,tom).	female(mary).
daughter(tom,ann).	-	parent(tom,eve).	female(eve).
daughter(eve,ann).	-	parent(tom,ian).	

Example 2.2-2 (Running example of ILP). Let's consider a running example adopted from Section 2.2 in [71], where a sample ILP problem with target relation *daughter(X, Y)* and background relations *female* and *parent* is depicted in Table 2.1. The task here is to find the definition of the relation *daughter(X, Y)*, given two positive and two negative examples (denoted as + and −, respectively) in the training data set. In a definite program clauses setting, a consistent and complete definition (with respect to the background knowledge and the training examples) may be induced as follows:

$$daughter(X, Y) \leftarrow female(X), parent(Y, X)$$

This clause can be interpreted by a propositional rule: *if X is female and Y is X's father, then X is Y's daughter.*

A number of well-known methods, which are considered as first-order upgrades of existing propositional learners, have been proposed in the ILP community [168, 202]. These algorithms include the first-order inductive learner (FOIL) [199], the top-down induction of logical decision trees (TILDE) [20], the Progol method [163], and the CrossMine algorithm [242], amongst others.

FOIL

One of the best known approaches to deal with structured data is the FOIL algorithm [199], which upgrades the well-known CN2 method [40] to deal with first-order representations. The FOIL algorithm learns a Horn clause definition of a target relation [199]. This approach employs a general-to-specific search to build a set of rules to explain many positive examples and cover as few negative ones as possible.

Example 2.2-3 (Running example of FOIL). Consider a two-class problem such as the Customer Risk problem depicted in Figure 1.2. The FOIL algorithm constructs a set of *conjunctive rules* to distinguish "good" loans from "bad" loans (i.e. positive and negative classes) in the target relation *Loan*, through the use of information both from the *Loan* and *Account* relations.

The FOIL strategy repeatedly searches for the best rule to separate positive examples from negative ones. Each such rule consists of a set of *predicates*. A predicate is a constraint on an attribute in a table. For example, the predicate *"Loan(N,-,-,-,duration* $\geq$ *24,-)"* describes the loan N with a duration of no less than 24 months.

In order to build a good rule, the FOIL algorithm searches relevant features across the two relations through the *account-id* join between them (shown in Figure 2.1), and evaluates all possible predicates to find the one which has the best capability to distinguish positive and negative tuples in the *Loan* relation. For ex-

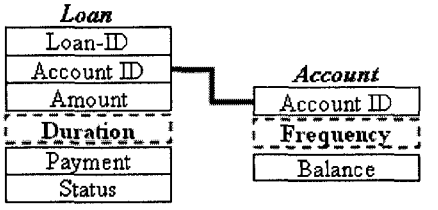


Figure 2.1: Search across multiple relations by FOIL [13]

ample, predicate *"Loan (N, A, -,-,duration ≥ 24,-), Account(A,-,balance ≥ 50) "* defines a predicate which uses information from both the *Loan* and *Account* tables. This predicate uses the join through the *account-id* attributes, and describes *loans* with *duration* of at lease 24 months and an *account balance* of no less than *$50.*

To evaluate a predicate, the FOIL algorithm appends this candidate predicate to the current rule and calculates the number of positive and negative tuples which satisfy the appended rule. This search process is very time-consuming in cases of the presented data having a large number of relations or possible predicates [242].

At the end of the learning process, a set of rules is constructed to distinguish "good" loans from "bad" loans. A rule may look like this: *"Loan(N,+):-Loan(N,A,-,D,-), D > 24, Account(A,-,B), B > 50."* This rule says that a loan will be a "good" loan if its *duration* is longer than 24 months, and its *account balance* is larger than $50.

Progol

Another well-known ILP approach is the Progol system, which is considered as an upgrade of the AQ algorithm [160]. The Progol approach was first presented in 1995. As FOIL does, the Progol method [163] employs a covering approach. In its induction process, the Progol system *selects* an example to be *generalized* and finds a consistent clause covering the example. Through repeating the example selection and generalization cycle, the Progol algorithm keeps constructing hypothesis clauses which are consistent with the training examples until the obtained clauses cover all examples.

The Progol algorithm combines Inverse Entailment [163] with a general-to-specific searching method to explore the hypothesis space. Inverse Entailment is employed to compute the most-specific clause (belonging to the hypothesis language specified by the means of mode declarations provided by the user) for each given example (seed example). The most specific clauses are then used to guide an A*-like search [59, 177] over clauses which subsume the most specific clause. As introduced in [94, 210], a best-first search strategy first evaluates all currently-available transitions (based on a heuristic measurement). Next, the transition which looks the most promising is explored. Like the best-first search, the A*-like searching strategy employs a heuristic estimation to guide the exploration process. However, unlike the greedy strategy employed by the best-first searching method, the A*-like approach also takes the distance already traveled into account. Such a consideration makes the A*-like algorithm complete and optimal [177]. An introductory tutorial on the Progol approach is presented by Muggleton and Firth in [167].

TILDE

In contrast to the general-to-specific searching approach, the TILDE method [20] employs the divide-and-conquer searching technique. The TILDE algorithm extends the popular C4.5 propositional learner to tackle relational representations. Research has shown that the divide-and-conquer approach embedded in the decision tree construction makes the TILDE method an efficient one, compared to many traditional ILP algorithms [242].

The major upgrade for the C4.5 method is as follows. First, unlike its propositional counterpart where attribute values are tested in the nodes of a decision tree, the TILDE algorithm applies logical queries in the nodes as the decision tree is being constructed. In addition, a node in the TILDE logic decision tree consists of a conjunction of predicates (literals) and different nodes may share variables. Also, tests at each node of the TILDE strategy are computed under a refinement

operator, which adds one or more literals to the current clause (the current clause consists of literals in a path from the root to the current node).

Table 2.2: An sample ILP problem for TILDE [18]

Training examples	Background knowledge
worn(gear), worn(chain): label(fix) worn(engine), worn(chain): label(sendback) worn(wheel): label(sendback) :label(keep)	replaceable(gear). replaceable(chain). not_replaceable(engine). not_replaceable(wheel).

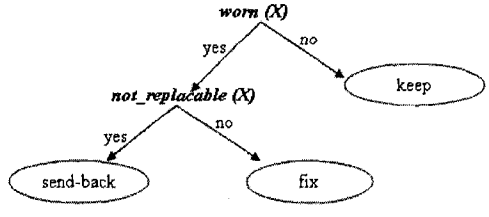


Figure 2.2: A logic decision tree constructed by TILDE [18]

Example 2.2-4 (Logic decision tree constructed by TILDE). Figure 2.2 depicts an example logic decision tree as constructed by the TILDE algorithm, using examples and background knowledge provided in Table 2.2. The task of this problem is to learn a hypothesis to predict the actions that should be taken on the condition of a machine X. An action is represented by one of the three class labels: *keep*, *send-back*, or *fix*. Like C4.5 decision trees, logic decision trees uses internal nodes to represent tests, and leaves for the class value predictions. In a logic decision tree, tests are executed as queries. Such queries are the conjunction of the literals in the current node and the literals on the path from the root of the tree to the current node. For example, logic decision clauses which represent the tree constructed in Figure 2.2 are as follows [18]:

$$label(keep) \leftarrow \forall X : \neg worn(X)$$

$$label(send\_back) \leftarrow \exists X : worn(X) \wedge not\_replacable(X)$$

$$label(fix) \leftarrow \exists X : worn(X) \wedge \forall Y : (\neg worn(Y) \vee \neg not\_replacable(Y)$$

CrossMine

Another important method for multirelational data mining is the CrossMine algorithm, which addressed the following problem. Scalability is a big challenge to approaches using ILP techniques [166, 242]. The scalability problem is observed since the size of the hypothesis search space increases rapidly with the number of relations. Also, the construction of a single hypothesis may require repeated join operations to be performed in the database. Each join operation is very costly in terms of run time and storage space. To tackle this problem, Yin et al. [242] present the CrossMine algorithm, an extension of the FOIL approach. The authors propose an idea to implement virtual joins in a database, insteading of physical joins. That is, information about the tuple IDs (primary keys) and classes from the target relation are passed and stored in background relations. In this way, the FOILGain, which is used to measure the goodness of a predicate in the FOIL strategy, can be calculated in each local relation, without physically joining tables. Through virtual joining, this approach demonstrates high scalability and accuracy, as presented in [242].

Table 2.3: An example of information propagate in CrossMine [242]

account-id	frequency	balance	id information	class label information
00132	monthly	3521	001,002	2+,0-
00132	weekly	2000	001,002	2+,0-
00240	monthly	20	-	0+,0-
00100	weekly	50	-	0+,0-
0098	weekly	10000	005	1+,0-

Example 2.2-5 (Information propagation in CrossMine). Consider the two relations depicted in Table 1.2 again. After the propagation, the *Account* relation

contains two new attributes (as shown in Table 2.3): one is the *"id information"* attribute, which stores the information about the tuple IDs from the target relation *Loan* (indicating which tuple/tuples from the target relation relate to a specific record in the *Account* table); another feature, i.e. *"class label information"*, stores information about the tuples from the target table (indicating the number of *positive* and *negative* examples which relate to a specific tuple in the *Account* relation). For example, consider the record with *account-id* 0098 (shown in the last row of the Table 2.3). There is one corresponding *good loan* with *loan-id* of 005 in the target relation *Loan*. Therefore, the *id information* for this tuple is 005, and the *class label information* is one "good" loan (depicted as +) and no "bad" loans (indicated as -). Using the information stored in the two extra columns, the CrossMine algorithm is able to evaluate the FOILGain information of each predicate in the *Account* relation, without physically joining with the target relation *Loan*. In this way, the efficiency of the algorithm is achieved.

Statistical Algorithms

In addition to the more well-known ILP-based approaches as discussed previously, the "first-order upgrade" method also attracts significant attention in the statistical relational learning (SRL) community [90]. These approaches usually are proposed to combine logic theory with the probability principle [5, 89, 91, 164, 172, 188, 189, 219]. In this way, SRL models integrate the expressivity of first-order logic and the capacity of reasoning under uncertainty, thus relaxing the requirement for logical expressions in the search space.

One of the best-known SRL methods is probabilistic relational models (PRMs) [89], which extends Bayesian networks to deal with relational domains. PRMs consist of the structure of the network and the parameters associated with it. The nodes of the network describe the attributes of a relational database and its edges represent the probabilistic dependences between nodes. The PRM algorithm is used to capture the probabilistic distribution between attributes within a table and between

attributes in different tables in a relational database [89]. Following the same trend, Taskar et al., in [219], described a relational version of Markov Networks, where the structure of the network is based on the relational schema of the relational domain. This method is known as Relational Markov Networks (RMNs). Richardson and Domingos also presented an approach called Markov logic network (MLN) [207], which combines first-order logic with Markov networks. This method considers a Markov logic network as a first-order knowledge base and uses it as a template for constructing Markov networks. Other approaches such as Stochastic Logic Programs (SLPs) [164, 211] and Bayesian Logic Programs (BLPs) [118] have been introduced as well. Continuing in the same trend, Relational Probability Trees (RPTs) [175], which is considered as a relational version of conventional probability estimation tree algorithm, has been investigated recently. In addition, Flach and Lachiche presented the 1BC approach [79], an alternative first-order upgrade to the Naive Bayes algorithm.

Besides studies on "upgrading" statistical learning methods, research which aims to directly integrate ILP systems with probabilistic propositional algorithms has also been conducted. The main idea here is that one can interpret the clauses of rules learned by the ILP systems as propositional "features". In this way, traditional probabilistic learning models can be applied to learn from these propositional "features" [52, 187]. For example, Landwehr et al. introduced the kFOIL method [145], an extension of the FOIL algorithm in order to integrate ILP systems with support vector learning. The kFOIL strategy dynamically applies a FOIL-like algorithm to induce a set of clauses and then uses these clauses as features to define a kernel for the support vector learning. Following the same line of research, Landwehr et al. provided the nFOIL and tFOIL algorithms [144], where ILP system and probabilistic models are integrated. In the two methods, the covering search for good clause features of the FOIL algorithm is guided directly by the criterion of Naive Bayes models. Another two systems, called SAYU [50] and SAYU-VISTA [53], were also developed and have similar properties as that of the nFOIL and tFOIL approaches. That is, first-order logic clauses generated by ILP

algorithms are used as "features" by statistical models. The SAYU and SAYU-VISTA systems are extension versions of the early approach described in [51]. Following the same research trend, Craven and Slattery proposed a strategy called Statistical Predicate Invention [47] in order to deal with hypertext data. In this approach, classifications predicted by Nave Bayes classifiers are used as predicates in the learning process of the FOIL algorithm.

2.2.2 Propositionalization Strategy

The second category of strategies used to handle relational domains is the so-called propositionalization methods. As introduced in Chapter 1, propositionalization algorithms flatten multiple relations into a universal single one. Subsequently, this flattened relation is presented to propositional learners. Formally, the transformation process can be defined as follows.

Definition 2.2-6. Propositionalization ("flattening") is the process of transforming structured data into an attribute-value flat file form. This single table presentation then serve as the input of a propositional learning method, which uses it to construct a model.

Since its first success, which was illustrated by the LINUS approach [147], propositionalization has been an active research area. The general idea is to enable wide range of efficient and accurate propositional learning algorithms from the single-table learning community, to be applied to mine structured data [70, 71].

The existing propositionalization approaches described in current literature can be categorized into two families: the Logic-oriented approaches and the Database-oriented strategies [137]. Logic-oriented propositionalization algorithms are introduced first and followed with Database-oriented methods.

Logic-Oriented Methods

A Logic-oriented propositionalization approach defines attribute-value features (propositional features) using pure logic, i.e. existential features. Systems such as LINUS,

DINUS, and SINUS (which will be discussed in detail next) employ this strategy [137, 147, 149].

LINUS

The LINUS algorithm was first presented in detail in 1990 [149], and is one of the pioneering systems of propositionalization strategies. The chief idea of the LINUS method is that the background knowledge can be used to introduce new attributes into the learning process. To solve an ILP problem, the LINUS algorithm involves the following three steps: 1) the structural data are transformed into an attribute-value, flat file presentation; 2) a single table learning algorithm is employed to learn from the converted flat file, constructing a propositional hypothesis, i.e. if-then rules; and 3) the induced if-then rules are transformed into ILP logic clauses.

Table 2.4: Training examples and background knowledge for LINUS [136]

Training examples		Background knowledge		
daughter(sue,eve).	+	parent(eve,sue).	female(ann).	male(pat).
daughter(ann,pat).	+	parent(ann,tom).	female(sue).	male(tom).
daughter(tom,ann).	-	parent(pat,ann).	female(eve).	
daughter(eve,ann).	-	parent(tom,sue).		

Table 2.5: Flat file after being flattened by LINUS [136]

			New	Created	Features					
x	y	class	f(x)	f(y)	m(x)	m(y)	p(x,x)	p(x,y)	p(y,x)	p(y,y)
sue	eve	+	1	1	0	0	0	0	1	0
sue	eve	+	1	1	0	0	0	0	1	0
sue	eve	+	1	1	0	0	0	0	1	0
sue	eve	+	1	1	0	0	0	0	1	0

Example 2.2-7 (Running Example of LINUS). Table 2.4 and 2.5 provides a running example (adopted from [136]) of the "flattening" process in the LINUS sys-

tem. (Table 2.4 provides the source data and Table 2.5 introduces the propositional data). In this learning problem, similar to the example introduced in Table 2.1, the target relation is *daughter(X, Y)*, denoting that person X is the daughter of person Y. The task here is to learn the definition *daughter(X, Y)* with the help of background knowledge relations *male*, *female*, and *parent*. All variables here, namely *mary*, *ann*, *eve*, *tom*, and *sue* are of the same type *person*.

In the propositionalization process of the LINUS system, new features are created to reflect the possible applications of the background predicates on the arguments of the target relation. This process also takes into account the argument types: a new feature is created to reflect each such application. Next, boolean values (1 for true, 0 for false) are used to indicate the background knowledge predicates (shown in Table 2.5, where newly constructed features are denoted on the right- hand side of the table). In Table 2.5, f, m, and p stand for *female*, *male*, and *parent*, respectively.

DINUS and SINUS

Following the same lines of thought, propositionalization approaches such DINUS [147] ("determinate LINUS") and SINUS were designed.

The DINUS method was first published in 1992, and is considered as the successor of the LINUS algorithm. DINUS accepts the same input as that of the LINUS system. On the other hand, the DINUS relaxed the hypothesis language bias to allow learning non-constrained clauses if they are determinate.

In the paper [137], the SINUS system, the successor of LINUS and DINUS approaches, is also introduced. Enhanced propositional feature construction methods are added, compared to that of the LINUS and DINUS methods. In addition, the SINUS algorithm is implemented in SICStus Prolog. As a result, the final propositional rules can be converted back into Prolog form, for further exploration.

Database-Oriented Algorithm

The second major category of propositionalization algorithms is composed of the Database-oriented approaches. These methods generate propositional features using *aggregation* from relational database systems, while flattening multiple relations. In the next three sections, we will present the idea of aggregation and then follow that up with the introduction of two aggregation-based propositionalization algorithms, namely, RollUp and RelAggs.

Aggregation

Aggregation functions are often used to statistically describe the properties of samples of populations. For example, *average* and *standard deviation* are two popular functions in this field. In fact, aggregation is commonplace in database systems [31]. In a database management system, an aggregate function (such as *min, max, sum, count*, or *mean*) takes a set of tuples as input, and then summarizes the properties of the set, producing a single value as output. In this way, aggregate functions, through joins in a relational database, can project information stored in related relations to a specific table. In other words, aggregate functions can be considered as a form of feature construction. By using aggregates, the newly constructed aggregate features can summarize related information in background knowledge belonging to (or relating to) the target tuple. For this reason, aggregation, in fact, is a common method for preprocessing structural data for propositionalization algorithms [182].

Example 2.2-8 (Aggregation in SQL). Figure 2.3 and Table 2.6 demonstrate the "flattening" by using the aggregation functions *Maximum, Minimum*, and *Average* from SQL. As shown in Table 2.6, information from the three tuples in the *Account* table is summarized into the new attributes *Min(balance), Max(balance)*, and *Avg(balance)*, which then become part of the information in the tuple with *loan-id* of 001 in the *Loan* table.

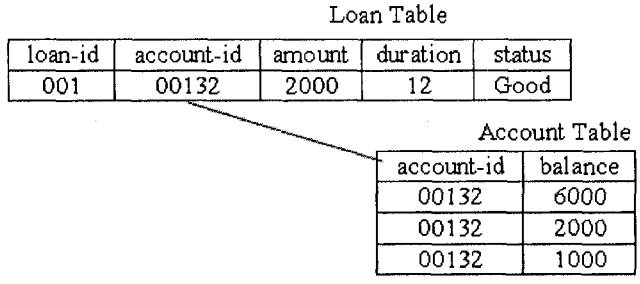


Figure 2.3: Target tuple and its related background tuples

Table 2.6: Aggregate tuples using *Max, Min,* and *Average* from SQL

loan-id	account-id	amount	duration	status	Min(balance)	Max(balance)	Avg(balance)
001	00132	2000	12	good	1000	6000	3000

Due to its capability of summarizing and squeezing multiple related tuples into a single tuple, aggregation functions provide an attractive way to create new attributes from relational data [184]. The RollUp and RelAggs methods, which employ aggregates in databases to implement the propositionalization process, are presented next.

RollUp

Knobbe and Siebes developed the RollUp algorithm to perform the "flattening" of multiple relations in a relational database [124]. The RollUp approach employs a depth-first search (DFS) in a relational database to aggregate the information in deep-level relevant tables (i.e. tables far from the target relation) to their parent tables (i.e. tables less far from the target relation), using aggregate functions, such as *count, min, sum, max,* and *mean*, found in SQL. This summarization procedure stops if attributes from all relevant tables in the DFS search are aggregated onto the target table. The resulting universal table is then used as input by a propositional learning algorithm. The drawbacks of this approach are that the "roll up" process may aggregate the same background table multiple times onto the target relation

and that a maximum search depth has to be specified [136].

RelAggs

Following the same ideas as the RollUp approach, Krogel and Wrobel introduced a so-called RelAggs propositionalization method [139]. This algorithm overcomes the above-mentioned limitations of the RollUp algorithm and has been applied to learn from business [139] and biology domains [38]. In the RelAggs strategy, sophisticated aggregate operators are used to construct new features for a single-table propositional learner. In this way, crucial information from background relations can be represented by the newly constructed attributes in the target table.

loan-id	account-id	amount	duration	payment	status	*New_frequency*	*New_balance*
001	00132	2000	12	100	Good	***monthly***	***3521***
						weekly	***2000***
002	00132	4000	12	200	Good	***monthly***	***3521***
						weekly	***2000***
003	00148	1000	24	80	Bad	***NULL***	***NULL***
004	00148	30000	24	1000	Bad	***NULL***	***NULL***
005	0098	10000	36	800	Good	***weekly***	***10000***

Figure 2.4: Flattening through join of *Loan* ⋈ *Account*

Example 2.2-9 (Propositionalization in RelAggs). Let's consider the example Customer Risk database shown in Figure 1.2 again, focusing on the *Loan* and *Account* tables. In the "flattening" process, new features are created in the *Loan* target table to reflect information from the background relation *Account*.

The propositionalization process proceeds as follows. First, these two tables are joined together. As shown in Figure 2.4, through the join between these two tables, each tuple in the target relation *Loan* has one, more than one, or no related tuples in the background relation *Account*. Next, aggregation functions are then applied to the existing related background tuple(s) to summarize information processed by these tuples. Otherwise, NULL value is applied to indicate the absence of related information. For example, for the entity with *loan-id 001*, there are two related tuples in the *Account* table. Thus, a set of aggregate features are created

to summarize these two related entities. When considering the tuple with *loan-id 003* in the target table, there is no related entity in the *Account* relation, so NULL values are used to indicate the absence of related information.

loan-id	...	status	Count(monthly)	Count(weekly)	Min(balance)	Avg(balance)	Sum(balance)	Max(balance)
001		Good	1	1	2000	2760.5	5521	3521
002		Good	1	1	2000	2760.5	5521	3521
003		Bad	NULL	NULL	NULL	NULL	NULL	NULL
004		Bad	NULL	NULL	NULL	NULL	NULL	NULL
005		Good	0	1	10000	10000	10000	10000

Figure 2.5: Final flat file of propositionalization in RelAggs

Figure 2.5 shows the final results of this propositionalization. As shown in the final single table, besides the original attributes (highlighted in light blue) from the target relation *Loan*, newly constructed features, namely *Count(monthly) and Count(weekly)* (created from attribute *new_frequency* presented in Figure 2.4 and highlighted in yellow), and *Min(balance), Avg(balance), Sum(balance)*, and *Max(balance)* (generated from attribute *new_balance* presented in Figure 2.4 and highlighted in green) are added. These newly created features summarize related information from the background relation *Account*. In this way, the converted flat file can be used as input to various propositional learning algorithms.

Other Propositionalization-Related Approaches

The idea of propositionalization has also been applied to some other learning tasks, such as finding association rules and employing distance-based learning. Discovering frequent queries was described by Dehaspe and Toivonen in their algorithm WARMR [61]. This approach is considered as an upgrade of the well-known propositional frequent pattern strategy APRIORI [2]. Following the same line of thought, Emde and Wettschereck presented the so-called RIBL strategy, a relational distance measure for distance-based learning [73]. In addition, Lavrac et al. introduced the RSD algorithm in order to discover subgroups in relational domains [148].

The next section provides a comparative summary of the "Upgrading" and "Flattening" algorithms, showing their strengths and weaknesses.

2.2.3 "Upgrading" versus "Flattening" Strategies

In the previous sections, we introduced well-known algorithms from the two main families of existing multirelational methods, i.e. first-order upgrade and propositionalization. Research has shown that each of these two categories of strategies have its *Pros* and *Cons*. Table 2.7 shows a summary of these two families of approaches.

Table 2.7: Comparison summary of "flattening" and "upgrading" methods

	Propositionalization	First Order Upgrade
Extensive preprocessing of data	Yes	No
-Transform relations into a flat file	Yes	No
-Lose normalized compact structure	Yes	No
-Create statistical skew in the flat file	Yes	No
-Result in a large number of NULL values	Yes	No
-Generate new attributes exponentially	Yes	No
Employing conventional methods	Yes	No
Handling real-world numeric data	Good	Bad
Dealing with noisy values	Good	Bad
Integrating with Databases	Yes	Limited
Expressive power	Limited	good
Learning recursive clause	Inability	Yes
Dealing with large data set	Good	Bad
Requiring end user domain background	No	Some

Firstly, we focus on the data preprocess requirements. As shown in Table 2.7, propositionalization methods need extensive preprocessing of the data. That is, they require considerable time and effort to transform multiple relations into a universal flat file. Recall that, as a consequence of such "flattening" process, we may lose the normalized compact structures of relational databases, cause statistical skew in the resulting data [70, 82], generate a large number of NULL values (missing

values) in the flat file [48], and result in a big table with an exponentially large number of additional attributes [200, 71]. In contrast, first-order upgrade strategies exclude the need of converting multiple relations. The above mentioned drawbacks resulting from the flattening process may therefore be avoided.

Secondly, as indicated in Table 2.7, these two strategies differ in the capability of employing propositional algorithms. In a propositionalization method, multiple relations are converted into a single table, in order to be solved by propositional algorithms. Thus, a wide range of accurate and efficient single-table learning methods can be applied [71]. Instead, first-order upgrade approaches have to "re-invent the wheel" in order to exploit information distributed across multiple relations. In other words, new techniques had to be designed to enable first-order upgrade algorithms to efficiently search for relevant features distributed across interconnected relations [70].

Thirdly, these two methods show varying ability to deal with real-world data—which often contain numeric and noisy values. First of all, the manner of treating numerical values in ILP-based approaches is typically static and limited. In fact, only a small number of ILP systems can handle numeric data well through discretization, compared to discretization strategies employed in the majority of propositional learning algorithms [124]. This difficulty is due to the potentially non-determinate relationships between the target tuples and the entities in the background relations. Such indeterminate connections have challenged existing learning algorithms' handling of numerical values, which exist in both the target table and the background relations [124, 136]. Research has shown that, such facts often require ILP systems to pre-determine numeric thresholds used in the refinement process, which is a crucial step in the construction of a hypothesis [124]. Second, when considering handling noisy data, research has shown that first-order upgrade algorithms often yield unsatisfactory predictive performance, compared to that of aggregate-based propositionalization methods [125, 183].

When considering the capacity of integrating with relational databases, propositionalization approaches such as RelAggs and RollUp are able to take advantage of

the aggregation techniques and computational power of the database management system to enhance the learning process. On the other hand, logic-based upgrade algorithms often face a barrier when it comes to being integrated into relational databases. The reason for this barrier is as follows. Logic-based methods often use logic to represent the learning data, such as in the form of Prolog syntax. In other words, a pre-processing transformation has to be conducted in order to convert the data stored in multiple tables of a relational database into logic syntax forms [19].

As shown in Table 2.7, by virtue of their expression in logic, ILP-based methods are able to describe more complex learning concepts. In contrast, propositionalization approaches are unable to do so, due to the propositional setting. However, the disadvantage of the expressive power of ILP systems is that such power relies on the use of first-order logic clauses. Such presentation often requires the users to set the right trade-off between expressive power of the hypothesis language and efficiency of the learning process (e.g. hypothesis language bias), and to prepare the input data in the right format. These setting and preprocessing requirements may make the ILP system difficult to be used by users who are only familiar with propositional presentations [16, 136].

In addition, as indicated in Table 2.7, these two methods are quite different in the manner of handling large data set. Recall that, logic-based approaches usually suffer from poor scalability when dealing with complex database schemas. ILP techniques tend to be time consuming when building hypotheses. Thus, data sets which dealt with by ILP methods are typically small [21, 22]. On the other hand, many state-of-the-art efficient learning techniques can be "plugged" into propositionalization methods. Also, single-table learning methods tend to be much more efficient in terms of computational cost than relational systems [185].

In practice, the first-order upgrade approach is usually not easy to use. This method often needs some domain knowledge about the data and some logic programming background. For example, efforts to decide on the right representation for the training information (such as training examples and background knowledge) and the right trade-off between expressive power of the hypothesis language

and the efficiency of the learning process (e.g. hypothesis language bias) are often required in order to start the first-order learning process [136].

In summary, the two major families of approaches have been actively investigated in recent years. These algorithms have different preferences about the applied application domains. Nevertheless, research has shown that both of the two categories of strategies, as discussed above, have demonstrated some shortcomings, resulting in certain barriers which have prevented a wider application thereof.

While this research focuses on mining relational databases, other forms of structured data also exist. The next section therefore provides a brief introduction about learning from such other forms of structured data, such as graphs and links.

2.2.4 Learning Structured Data in Other Forms

Besides the surge of interest in strategies on mining relational databases, methods to mine structured data represented by graphs and link data have also become very popular.

Graphs represent a general class of structured data. Mining graph data, where directed or undirected edges are used to represent the relationships among linked objects, have attracted many researchers [43, 44, 119, 97, 143, 157, 236, 237]. Recently, efficient methods have also been developed for a variety of applications, including mining frequent subgraph, indexing graph patterns, labeling subgraphs, and building classification or cluster analysis systems to categorize unknown graphs, amongst others. In a graph-based approach, training examples, background knowledge, and target concept are described as graphs.

Example 2.2-10 (Search for graph). Figure 2.6 depicts a molecule database (as presented by Fischer and Meinl in [78]), where structures of three molecules with similar substructures but different length of the carbon chains (the shadowed regions) are depicted. A typical subgraph pattern learning system may be looking for the common pattern indicated by the shadowed regions of these molecule graphs.

Three main bases for learning from graphs have been actively investigated:

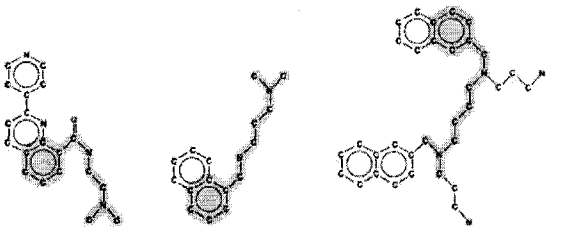


Figure 2.6: Finding frequent subgraphs in a molecule database [78]

mathematical graph theory, greedy (heuristic) search, and kernel functions [119]. The mathematical graph theory based strategy uses a support or frequency measure to evaluate the solution. Well-known methods include the APRIORI-based frequent subgraph mining algorithms, ACM [111] and FSG [143] and the pattern-growth-based graph pattern discovery method, gSpan [236]. On the other hand, greedy search based methods employ heuristic measures to evaluate their findings. This kind of algorithm includes the GBI algorithm [157] which uses the size of the extracted graph as a heuristic measure and the Subdue strategy [43] where Minimum Description Length (MDL) is used as the evaluation heuristics. In contrast, Knodor and Lafferty apply a kernel function, i.e. a diffusion kernel function, to transform the graphs into a higher dimensional space, and then solve the problem in that space [134]. Instead of mining the entire graph patterns which are often very large, some researchers only mine a subset of the entire graph, such as closed graph patterns (a graph pattern is closed if it has no supergraph with the same support as the pattern itself.). Algorithms for mining closed graph patterns have been studied by Yan and Han in [237] and by Yan et al. in [238]. They proposed highly efficient methods for mining this type of data.

Another popular research field that relates to structured data mining is link mining [54, 88, 142, 245, 247]. In this sort of data, links which connect different seemingly uncorrelated objects may demonstrate some useful patterns in mining tasks. Link analysis tends to capture link patterns among objects. Applications such as bibliographic citations, web mining, and XML data mining fall in this field [86]. Using link structures to improve the results of information retrieval

is a widely studied link mining task. One of the famous ones is the PageRank measure proposed by Page et al. in [179], where link structure is used to improve the information query results. The PageRank method provides an objective way to rate the human interest of certain web pages [179]. Basing on the linkages of the web, Kleinberg introduced the hubs and authority scores for the web information retrieval [123]. Some more recent development in this field include strategies of considering other relations. For example, Richardson and Domingos included both the content and linkage information of the web in order to improve the retrieval outcomes [206]. Dean and Henzinger took the co-citation information into account in order to find more related web pages [58]. Following the same trend, some other typical link mining tasks include link-based classification and clustering, link type identification, and link strength and cardinality prediction [102]. For instance, Getoor et al. described methods to deal with link-based classification tasks [87, 155, 214]. In such applications, attribute values of entities and their interlink and correlation information are exploited in order to improve the predictive performance of the classifiers. In [227], Wang and Kitsuregawa presented a link-based clustering method, where common links and co-citation information are used to cluster high relevant pages.

Besides mining from graph and link data, some more specific fields of structured data mining applications have also been investigated. As an example, a social network can be represented by a graph [102]. However, this kind of networks exhibits certain special characteristics. For instance, how the network grows and shrinks shows interesting patterns for researchers. In addition, Viral Marketing Analysis pays more attention on finding patterns to explain or predict the interactions between customers. Also related is newsgroup analysis, which can be considered as a special instance of mining structured data as well [102, 205].

2.2.5 Some Other Studies on Relational Learning

In the past decade, some other studies related to the learning in relational domains have also been introduced.

In order to enable the ILP systems to directly learn in relational databases, Blockeel and De Raedt introduced a framework for employing database queries for hypothesis searching in the ILP systems [19]. This idea was later implemented by Atramentov et al. in [7]. Bockhorst and Ong presented the FOIL-D method [24] to allow the FOIL algorithm to directly operate on relational databases through SQL queries.

Aiming at learning from highly structured data, Gartner et al. presented a method which constructs a kernel following the syntactic structure of the provided data [85]. They used a typed syntax and considered training examples as closed terms in order to collect related information about an example. In this way, a kernel can be defined following the type structure of the individual instances used for the learning. Following the same line of research, Dick and Kersting [63] combined Fisher kernels into relational statistical learners.

Ensembles of relational learning methods have also been investigated [6, 69, 107, 190, 198]. Quinlan introduced the FFOIL algorithm [198], which applies the Boosting strategy [81] to the FOIL system. In order to tackle the problem that boosting standard ILP systems is too slow and may cause a loss in comprehensibility, Hoche and Wrobel [107] described the so-called Constrained Confidence-rated Boosting method. Following the same trend, Dutra et al. [69] presented an algorithm which applies the Bagging strategy [27] to ILP systems. Also, Assche et al. [6] combined the random forest approach [28] with ILP methods. In addition, Preisach and Schmidt-Thieme [190] studied the performance of ensembles of relational learning models.

Under a relational learning setting, some other learning studies have also been introduced. Neville and Jensen [173, 174] investigated the Bias and Variance problem in the relational domains. Chiang and Poole [39] presented a method which

allows dynamic construction of predicates instead of predefined predicates in the ILP systems. Craven and Slattery [47] described a method to combine several Naive Bayes models with FOIL. Srinivasan introduced the Aleph system [217], an implementation of the Progol algorithm in order to learn rules. Yin et al. [243] applied the idea of "virtual join" when building decision trees from multiple interconnected relations, in order to speed up the relational learning process. Approaches which extend the ILP systems' rule construction process to include aggregation predicates were also studied [80, 223].

2.3 Summary

Commercial relational databases abound and multirelational data mining has become a field of strategic importance. The importance lies in the fact that existing mining methods have shown shortcomings, resulting in preventing a wider application of knowledge discovery to relational databases. This chapter introduced related work from both the multi-view learning and multirelational data mining research communities.

Recall that, multi-view learning provides a new technique for constructing models. In this thesis we will show that this learning strategy sheds lights on a new solution for multirelational data mining, as follows. Recall that, in relational databases, closely related features, in terms of semantic meaning, are often grouped in different relations. This characteristic projects a potential bridge between multi-view learning and multirelational data mining. Intuitively, a multi-view learning problem with n views can be seen as n semantically *diverse* sets of features, which are distributed in the multiple relations of a relational database.

This thesis integrates multi-view learning into the multi-relational data mining setting. Unlike existing relational methods, our algorithms learn from multiple sets of features of a relational domain. Specifically, our strategies distinguish themselves from existing multirelational mining approaches by excluding the need to *either* transform multiple relations into a universal single table *or* to devise new techniques

for direct relational learning. That is, existing single-table data mining techniques can be employed within our frameworks.

The next chapter will introduce a multiple view strategy for multirelational classification.

CHAPTER 3
MULTIPLE VIEW MULTIRELATIONAL CLASSIFICATION

Chapter 3

Multiple View Multirelational Classification

Recall from Chapter 2 that, in order to learn from a relational database, existing approaches fall in two categories. The first family of algorithms "upgrade" traditional learning algorithms to deal with relational presentation. The second category of strategies extensively preprocess ("flatten") the multiple relations to a single universal flat file, which is then used as the input to propositional learning methods. In this chapter, the MRC algorithm, which directly learn from a relational database while excluding the need to *either* transform multiple relations into a universal single table *or* to devise new learning algorithms for direct relational learning, is described. The MRC strategy enables learning, using conventional learning techniques, from multiple views of a relational database. Each view is constructed from one of the relations of the presented database. Next, knowledge acquired from individual views is then integrated to augment one another and to form the final classification model. The MRC algorithm offers both predictive performance and efficiency gains over current relational models, when mining diverse relational databases.

This chapter is organized as follows. In Section 3.1, the MRC algorithm is described. This is followed, in Section 3.4, with a comparative evaluation of the

MRC method. Finally, Section 3.5 concludes the chapter.

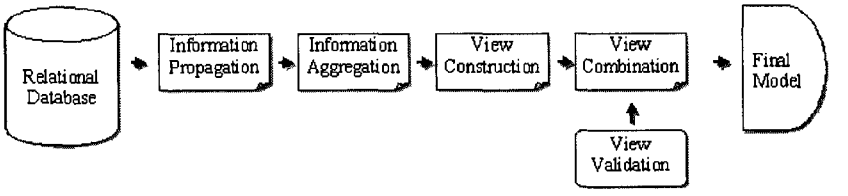


Figure 3.1: The MRC framework

3.1 The Multiple View Approach

The MRC algorithm enables one to classify relational objects by applying single-table data mining methods, while there is no need to flatten multiple relations to a single universal one. Our approach initially employs multiple view learners to separately capture essential information embedded in individual relation. Subsequently, the acquired knowledge is incorporated into a meta-learning mechanism to construct a final classification model. In detail, the MRC framework, as depicted in Figure 3.1 and Algorithm 1, consists of five sequential stages:

Algorithm 1 MRC Algorithm

Input: A DB= $\{T_{target}, T_1, T_2, \cdots, T_n\}$,
View learner $\mathcal{L}$, meta learner $\mathcal{M}$.

Output: Classification model $\mathcal{F}$.

1: Propagate and aggregate information, forming candidate view set $\{V_d^1, \cdots, V_d^n\}$;
2: Select a set $\{\mathcal{V}^i\}_1^{n'}$ from $\{V_d^1, \cdots, V_d^n\}$ (Algorithm 2);
3: Train $\mathcal{L}$ with $\{\mathcal{V}^i\}_1^{n'}$, forming hypothesis set $\{\mathcal{H}^i\}_1^{n'}$;
4: Form final model $\mathcal{F}$ by combining $\{\mathcal{H}^i\}_1^{n'}$, using $\mathcal{M}$;
5: **return** $\mathcal{F}$.

1) Information Propagation Stage: The Information Propagation Stage, first of all, constructs training data sets for use by a number of view learners, using a relational database as input. The Information Propagation Element propagates essential information from the target relation to the background relations, based on the foreign key links. In this way, each resulting relation contains different information, which then enables a propositional learner to learn the target concept.

2) Aggregation Stage: After the *Information Propagation*, the *Aggregation Stage* summarizes information embedded in multiple tuples and squeeze them into one row. This procedure is applied to each of the data sets constructed in the *Information Propagation Stage*. In this stage, aggregation functions are applied to each background relation (to which the essential information from the target relation was propagated). By applying the basic aggregation functions such as *min, max, sum,* and *count* in SQL, new features are created to summarize information stored in multiple tuples. Each newly constructed background relation is then used as training data for a particular view learner.

3) View Learners Construction Stage: In the third phase of the MRC algorithm, the *View Learners Construction Stage* constructs various hypotheses on the target concept, based on the multiple training data sets given by the *Aggregation Stage*. Existing single-table data mining methods (view learners) are used in order to learn the target concept from each view of the database separately. In this stage, a number of view learners, which are separately trained using different relations of the relational database, are constructed.

4) View Validation Stage: All view learners constructed in the *Multiple Views Construction Stage* is then evaluated in the *View Validation Stage*. The trained view learners need to be validated before being used by the meta learner. This processing is needed to ensure that they are sufficiently able to learn the target concept on their respective training sets. In addition, strongly uncorrelated view learners are preferred since disjoint views are encouraged by multi-view learning, in order to achieve better predictive performance.

5) View Combination Stage: In the last step of the MRC strategy, the resulting

multiple view learners (from the *View Validation Stage*) are incorporated into a meta learner to construct the final classification model. The meta learner is called upon to produce a function to control how the view learners work together, to achieve maximum classification accuracy. This function, along with the hypotheses constructed by each of the view learners, constitutes the final model.

The next two sections, i.e. Section 3.2 and Section 3.3, will discuss these five phases, in detail.

3.2 Learning from Individual Views

Recall from Section 2.1 that, in a multi-view learning setting, each view learner is assigned a set of training data with different views (feature sets), from which each should be able to learn the target concept. Recall from Chapter 1 that, often a relational database $\Re$ is designed by domain experts using an Entity Relationship (ER) model, where different relations usually store information with different semantic meaning. To apply the multi-view setting to multirelational classification tasks, we therefore construct a separate view from each relation of the database. Thus, in order to enable each view learner to independently learn to classify the target class from its own relation, we need a propagation process. That is, consider a target variable Y that (only) resides in the target table T_{target}, the multi-view classification algorithm has to first correctly propagate the target variable Y to all other background relations. This process is discussed next, in detail.

3.2.1 Information Propagation

The MRC approach uses foreign key chains to implement this propagation process. The tuple IDs (identifiers for each tuple in the relation) from the target table, which will be used by the aggregation functions, must also be propagated to the background tables. This procedure is formally defined as follows.

Definition 3.2-1 (Directed foreign key chain propagation). Given a database $\Re = \{T_1, \cdots, T_n, T_{target}\}$, where T_{target} is the target table with primary key $T_{target.key}$ and target concept Y. Consider a background table T_i , where $i \in \{1, 2, \cdots n\}$, with an attribute set $A(T_i)$, and a foreign key $T_i.T_{target\_key}$ that links to the target table T_{target}. If a tuple in table T_i has a key value referencing a tuple in the target table through the foreign key $T_i.T_{target\_key}$, we say that this tuple is joinable. All joinable tuples in table T_i are selected to form a new relation T_{i_new}. The attributes of table T_{i_new} are described by the expression $A(T_{i\_new}) = T_{target.key} + Y + A(T_i) - A_{key}(T_i)$, where $A_{key}(T_i)$ are the key attributes of table T_i.

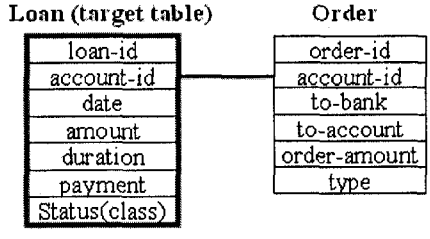


Figure 3.2: A directed foreign key chain

Example 3.2-2 (Directed foreign key chain propagation). Let us use the sample database introduced in Figure 1.3 to demonstrate this definition. The *Loan* target table has attributes *loan-id, account-id, date, amount, duration, payment*, and *status*. Here, the tuple ID is the attribute *loan-id* and the target concept is from the attribute *status*. For the background *Order* table, training data from this view will be created. The background relation *Order* has a reference key linking to the target relation *Loan*, as shown in Figure 3.2. By Definition 3.2-1, we know that this relation follows the *directed foreign key chain propagation* scenario. As such, the training data for the *Order* view consists of all tuples which have an *account-id* linking to the *account-id* in the target table. Each of these tuples has a *loan-id* (tuple ID) and *status* (class label) from the target table, along with all the *descriptive* attributes *to-bank, to-account, account*, and *type* from the *Order* table. This construction procedure can be converted into the following SQL query for a

MySQL database[1]:

```
CREATE TABLE order_new
SELECT L.loan-id, L.status, A.to-bank,
         A.to-account, A.amount, A.type
FROM   Loan L, Order A
WHERE  L.account-id = A.account-id;
```

Definition 3.2-3 (Undirected foreign key chain propagation). Given a database $\Re = \{T_1, \cdots, T_n, T_{target}\}$, where T_{target} is the target table with primary key $T_{target.key}$ and target concept Y. Consider a background table T_i, where $i \in \{1, 2, \cdots n\}$, with a foreign key $T_i.T_{target\_key}$ referencing the target table T_{target}. Next, consider a join path $p = T_a \bowtie T_b \bowtie \cdots \bowtie T_m$ (where $a, b, \cdots, m \neq i$ and $a, b, \cdots, m \in \{1, 2, \cdots n\}$) and another background table T_k , where $k \neq i$ and $k \in \{1, 2, \cdots n\}$ with attribute set $A(T_k)$ and foreign key $T_k.T_{i\_key}$ that links to table T_i through the join path p. If a tuple in table T_k has a key value linking to a tuple in table T_i, and this tuple has a link to a tuple in the target table T_{target}, we say that the tuple from T_k is joinable to the target table T_{target}. All joinable tuples in table T_k are selected to form a new relation T_{k_new}. We define the set of attributes for table T_{k_new} as $A(T_{k\_new}) = T_{target.key} + Y + A(T_k) - A_{key}(T_k)$, where $A_{key}(T_k)$ are the key attributes of table T_k.

Example 3.2-4 (Undirected foreign key chain propagation). Consider now the situation presented in Figure 3.3, which is taken from Figure 1.3. For the background *Client* table, the training data for this view is created as follows. First we notice that the *Client* relation has no directed foreign key referencing to the target relation *Loan*. However, it does have a reference key linked to relation *Disposition*, which in turn has a foreign key referencing the target relation *Loan*. By Definition 3.2-3,

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

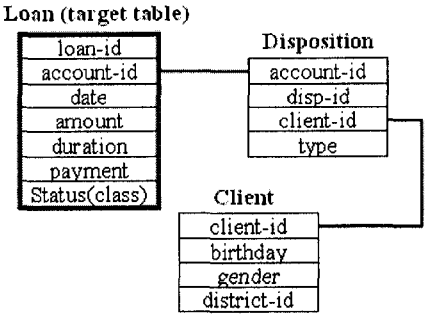


Figure 3.3: An undirected foreign key chain

this relation follows the *undirected foreign key chain propagation* scenario. Following this definition, the tuples from table *Client* which have *client-id* linked to the *client-id* attribute in table *Disposition* and the tuples from table *Disposition* with *account-id* linked to the target table are selected to construct the training data. Since the *Client* table has attributes *client-id, birthday, gender*, and *district-id*, the final training data consists of four attributes, namely *birthday* and *gender* from relation *Client* and *loan-id* and *status* from the target relation *Loan*. The above construction procedure can also be converted into the following SQL query for a MySQL database:

```
CREATE TABLE client_new
SELECT L.loan-id, L.status, C.birthdate, C.gender
FROM   Loan L, Disposition D, Client C
WHERE  L.account-id = D.account-id and
       D.client-id  = C.client-id
```

The MRC strategy builds views from relations which contain at least one *descriptive attribute* in the database. Except the view built directly from the target table, each background relations in a database will construct one view using the *shortest directed* or *undirected* foreign key chain, i.e. the foreign key chain with

less involved foreign key joins. The use of shortest foreign key chain is based on the following observations. First, shorter join paths can save storage and computational cost, compared to longer ones. In addition, join paths involving many relations have more chance to decrease the number of entities related to the target tuples, potentially providing less information about the target tuples. Finally, the semantic link with too many joins usually becomes very weak in a relational database [242]. Such weak links, therefore, tend to offer insufficient knowledge for learning the target concept. Basing on these observations, the MRC algorithm discourages long join paths.

One special case must be given more attention when using the shortest foreign key chains to implement the information propagation process. That is, in the case where one background relation has two different foreign key chains that have the same length (chains with the same number of involved foreign joins), the chain with the highest number of related target tuples is chosen. In other words, the chain resulting in a larger number of related tuples in the target relation is selected in order to provide more information for learning.

After the propagation, aggregation functions are then applied to the newly generated tables T_{i_new}, to finish constructing training data sets for the view learners. This aggregation procedure is described next in Section 3.2.2.

3.2.2 Aggregation-based Feature Generation

Relational data mining approaches have to handle the difficulties inherent with one-to-many and many-to-many associations between relations [71, 223]. The relations T_{i_new} obtained in Section 3.2.1 consist of one-to-many relationships with respect to the primary key value $T_{target.key}$ (i.e. the tuple ID) as propagated from the target relation T_{target}. This kind of indeterminate relationship has a significant impact on the predictive performance and has been studied by several researchers [125, 139, 184]. To address this problem, the MRC method applies aggregation functions taken from the field of relational databases.

Aggregation functions are originally used to statistically describe the properties of samples of populations [103]. For example, *average* and *standard deviation* are two widely used functions in this field. In commercial database systems, aggregate functions, which map a set of values to a value, are an important feature of practical database query languages [31, 225]. For example, the standard SQL language includes aggregation functions for the computation of *average, count, max, min,* and *sum.* Aggregation functions such as *sum* and *average* are also typical computation for statistical analysis of the data in data warehousing and OLAP applications [110]. When considering relational learning strategies, aggregations are often applied to summarize attribute values from related entities in one-to-many relationships associated with any particular tuple [140]. Over the past years, methods for generating aggregation features have been intensively studied in the relational learning community [18, 124, 147, 182, 184].

Aggregation has been widely used to summarize the properties of a set of related objects in order to "propositionalize" relational data for modeling [140, 175]. Knobbe [124] applied aggregate functions such as *min, max,* and *count* for propositionalization in the RollUp relational learning system. Neville et al. [175] used aggregation functions such as *average, mode,* and *count* for the relational probability tree system. Assche et al. [6] deployed aggregate features in first order random forests. Also, aggregate functions were used to summarize information from multiple tuples in the multi-relational decision tree induction [7, 126]. Recently, Reutemann et al. [204] developed a propositionalization toolbox called Proper in the Weka system [232]. The Proper Toolbox implements an extended version of the RelAggs algorithm designed by Krogel [140] for propositionalization, in order to allow single-table learning methods such as methods in the Weka package to learn from relational databases. In this RelAggs strategy, the *sum, average, min, max, stddev,* and *count* functions are employed for *numeric* attributes, and *count* function is applied for the *nominal* features from multiple tuples. Following the same research trend, the MRC algorithm deploys aggregation functions to squeeze multiple related tuples into a single row, grouping using the primary key from the

target relation. Each new table T_{i_new}, therefore, is able to be aggregated to form a view training set T_{i_view}. Each T_{i_view} is then used to train separate view learners.

Example 3.2-5 (Aggregation and propagation). Let us revisit the example we discussed earlier, namely the *Order_new* example presented in example 3.2-2, where tuple IDs and class labels from the target relation have been propagated. Also, suppose the MRC method applies the same aggregation functions deployed by the RelAggs algorithm in Weka. The *Order_new* table has nominal attributes *to-bank, to-account*, and *type* and numeric attribute *amount*. Three aggregation attributes are generated to replace the nominal attributes *to-bank*, *to-account* and *type*, respectively. They are *count(to-bank), count(to-account)* and *count(type)*. Six new aggregation attributes are created to replace the numeric attribute *amount*. They are *sum(amount)*, *avg(amount)*, *min(amount)*, *max(amount)*, *stddev(amount)*, and *count(amount)*. The aggregation procedure of the MRC method, along with the propagation procedure of Section 3.2 will result in the following SQL query:

```
CREATE TABLE order_view
SELECT L.loan-id, L.status, COUNT(A.to-bank),
       COUNT(A.to-account), SUM(A.amount),
       AVG(A.amount), MIN(A.amount), MAX(A.amount),
       STDDEV(A.amount), COUNT(A.amount), COUNT(A.type)
FROM   Loan L, Order A
WHERE L.account-id = A.account-id;
GROUP BY L.loan-id
```

Note that, the above SQL query combines the information propagation and aggregation steps of the MRC algorithm, and creates the *Order_view* view from the *Order* relation. The tuples stored in the *Order_view* table will be used to train a view learner (to be discussed in the next section). In other words, the process

which generates the table *Order_new* (where the temporarily intermediate results of the propagation process are stored) is no longer need.

In order to better understand the above query, we present the next example, which will show how data are retrieved and processed in the database for this query.

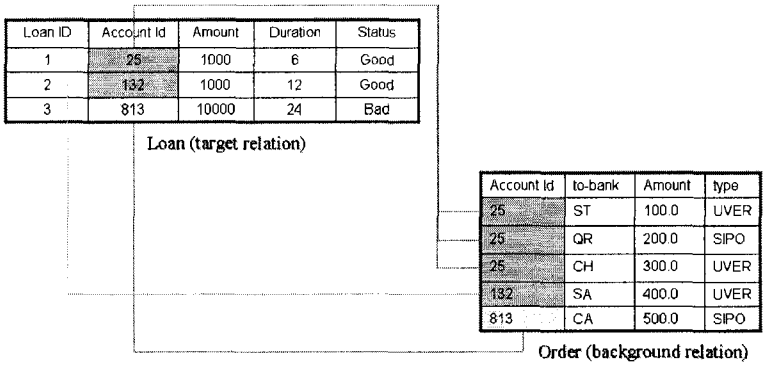
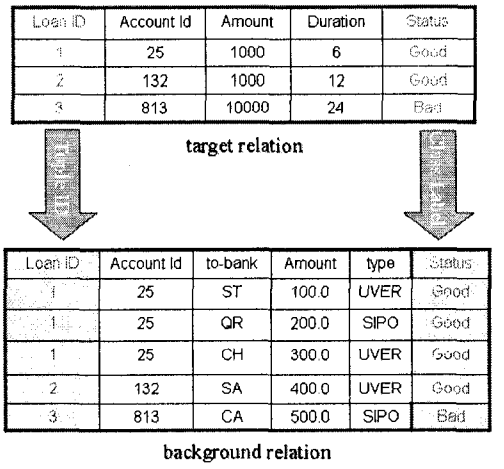


Figure 3.4: Example tuples of the sample database

Example 3.2-6 (A running example of the view construction process). Let's consider the example database (shown in Figure 1.3), where the *Loan* table is the target table and the target variable is the *Status* attribute. This target variable indicates whether a tuple in the target table is *good* or *bad.* In this database, there are seven background tables, i.e. tables *Account, Order, Transaction, Credit Card, Disposition, Client,* and *Demographic.* All background relations are associated with the target relation by foreign key joins. Following the view construction procedure described previously, the MRC method constructs eight (8) views, namely views constructed from tables *Loan, Account, Order, Transaction, Credit Card, Disposition, Client,* and *Demographic,* respectively. Here let's take a close look at how one of the views, e.g. *Order* view, is constructed, as follows.

Recall that, the relationship between the background table *Order* with the target relation *Loan* is presented in Figure 3.2. As we have previously discussed in Example 3.2-2, this foreign key join follows a *directed foreign key chain propagation scenario.* Consider parts of the data are provided in Figure 3.4, where join



Loan ID	Account Id	Amount	Duration	Status
1	25	1000	6	Good
2	132	1000	12	Good
3	813	10000	24	Bad

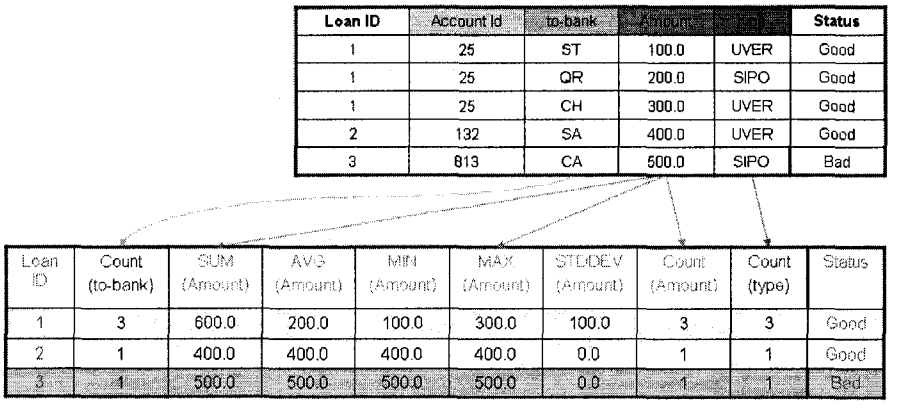
target relation

Loan ID	Account Id	to-bank	Amount	type	Status
1	25	ST	100.0	UVER	Good
1	25	QR	200.0	SIPO	Good
1	25	CH	300.0	UVER	Good
2	132	SA	400.0	UVER	Good
3	813	CA	500.0	SIPO	Bad

background relation

Figure 3.5: Propagation of essential information

path between target relation *Loan* and background relation *Order* exists through the key attribute *account-id*. Following the MRC algorithm described previously, attributes *Loan ID* (tuple ID) and *Status* (class label) are propagated to the *Order* relation (following the join path *account-id*). The resultant *Order* relation (after the propagation) is shown in Figure 3.5, where information about the tuple ID and class label exists.



Loan ID	Account Id	to-bank	Amount		Status
1	25	ST	100.0	UVER	Good
1	25	QR	200.0	SIPO	Good
1	25	CH	300.0	UVER	Good
2	132	SA	400.0	UVER	Good
3	813	CA	500.0	SIPO	Bad

Loan ID	Count (to-bank)	SUM (Amount)	AVG (Amount)	MIN (Amount)	MAX (Amount)	STDDEV (Amount)	Count (Amount)	Count (type)	Status
1	3	600.0	200.0	100.0	300.0	100.0	3	3	Good
2	1	400.0	400.0	400.0	400.0	0.0	1	1	Good
3	1	500.0	500.0	500.0	500.0	0.0	1	1	Bad

Figure 3.6: Aggregation of the *Order* relation

From Figure 3.5, we can also see that an one-to-many relationship (such as tuples with *loan-id* of 1) exists in the *Order* relation after the propagation. According

to the MRC algorithm, aggregation functions are applied to squeeze multiple related tuples into one. Consider we follow the same aggregation functions employed in the RelAggs algorithm as implemented in Weka, we obtain the resultant *Order* relation as shown in Figure 3.6. The aggregation process proceeds as follows. Since the original *Order* table has nominal attribute *to-bank* and *type* and numeric attribute *amount*. Two aggregation attributes (the second and ninth columns of Figure 3.6) are generated to replace the nominal attribute *to-bank* and *type*, respectively. Also, six new aggregation attributes (columns from third to eighth) are created to replace the numeric attribute *amount*. In this way, each tuple ID in the *Order* relation consists of only one tuple. Thus, this resultant relation can be used as training data for a single-table learning method to construct a view learner. The view learner construction process is discussed next.

3.2.3 View Learner Construction

After constructing the multi-view training data sets, the MRC algorithm calls upon the view learners to learn the target concept from each of these sets. Each learner constructs a different hypothesis based on the data it is given. Many traditional single-table learning algorithms, such as Decision Trees, SVMs, or Neural Networks, can be applied [71]. In this way, all view learners make different observations on the target concept based on their perspective. The results from the learners will be validated and combined to construct the final classification model. This is discussed in the next section, namely Section 3.3.

3.3 Multiple View Combination

3.3.1 View Validation

The view learners trained in Section 3.2.3 need to be validated before being used by the meta learner. This step is needed to ensure that they are sufficiently able to learn the target concept on their respective training set. In addition, recall from

Section 2.1 that strongly uncorrelated view learners are preferred since disjoint views are encouraged by multi-view learning in order to achieve superior predictive performance. These two issues are discussed next.

View Efficiency

The view validation has to be able to differentiate the strong learners from the weak ones [49]. In other words, in order to produce a combined classifier which is superior to the individual view learners, we need to ensure that each inducted view learner has sufficient knowledge on the target concepts to be learned. Consider a special case of a MRC learning setting, where the available database forms three disjoint data sets V_1, V_2, and V_3. In this case, a tuple x in the target relation T_{target} is viewed as

$$\vec{x} = \langle x^1, x^2, x^3, y \rangle$$

where x^1, x^2, and x^3 are instances in the data set V_1, V_2, and V_3, respectively. The variable y denotes the class label. Three view learners f^1, f^2, and f^3 are constructed by training using data sets V_1, V_2, and V_3, respectively. In this way, we have decision functions

$$f(x^1, x^2, x^3, y) = f^1(x^1, y) \cup f^2(x^2, y) \cup f^3(x^3, y)$$

where $\cup$ denotes a model combination scheme. Often, a common way to combine the three learners is a majority vote [112]. That is let

$$F(X) = \texttt{mode}\{f^1, f^2, f^3\}$$

where the *mode* operation chooses the classification which is predominantly predicted by the three view learners. In other words, x is assigned the class that receives the largest number of votes. However, poor predictors may be transferred into a poorly performing final model. For example, say view learner f^1 is very good at predicting the class but view learners f^2 and f^3 both have little knowledge

on the target concepts. Here, we may still obtain a combined classifier with poor predictive performance on the target concepts. We, therefore, need to evaluate the validation (in terms of sufficient knowledge on the classes to be learned) of the view learners f^1, f^2, and f^3, in order to obtained a better combined classifier.

In the MRC method, we use the error rate in order to evaluate the predictability quality of a view learner, since the MRC method aims to maximize the accuracy of the final induced model. A classifier which randomly guesses each instance's class has an error rate of 50% on binary problems [81]. In other words, a learner with error rate greater than such value means that the learner fails to learn the target concept. This number has been adopted by AdaBoost [81] to measure a hypothesis's learning ability over random guess. Following this line of thought, we here remove view learners which perform worse than random guessing, since they failed to learn the target concept. In other words, only learners with predictive performance better than random guessing are used to construct the final model.

View Correlation

Besides high efficiency, view learners prefer to be as *uncorrelated* to each other as possible. The next four sections will discuss this issue in detail.

A) Assumption of Uncorrelated Views

Recall from Chapter 1 that the theoretical foundations of multi-view learning are based on the assumption that the views are *independent* [23]. However, studies conducted by Muslea and Abney [1, 171] have shown that in real-world domains, the ideal assumption of multiple strictly independent views is often violated in the data. They have also demonstrated that when the independence assumption is violated, namely in applications with correlated views (such as views are weakly dependent), the multi-view approach can still perform well. In addition, research has also illustrated that disjoint views are preferred by multi-view learning in order to achieve better predictive performance of the final induced model.

Based on the above observation, the MRC method aims to identify a set of strongly uncorrelated views. We use the concept of *correlation* [235] to measure the dependence relationship of views. In statistics, correlation was first scientifically studied by Karl Pearson [1857 - 1936], and it refers to the departure of two variables from independence [235]. For example, if we say two random variables x and y are correlated, it means knowing x helps in predicting y; if knowing x tells us nothing about y, we say they are uncorrelated or their correlation coefficient should be zero. The term correlation in the thesis refers to the degree of dependence or predictability of one view with another. In other words, a view is said to be useful for building the classifier if it is correlated with the class, namely the view helps predicting the class; otherwise it is of little relevant. In addition, a view is redundant if one or more of the other views are highly correlated with it. That is, the redundant view can be predicted or implied by other views. This redundant view provides similar information with others (provides no extra predictive capacity). We, therefore, can evaluate a view's *quality*, in terms of helping predicting the class and its predictivity of other views, through the calculation of the correlation score. In this sense, a given view's quality here is similar to that of a feature in a supervised learning problem. In a classification task, a good feature is one that helps predicting the class but contains less common information with others [101].

The MRC method aims to construct a set of views which can better predict the class of the target tuples. Thus, on the one hand, we prefer views which are strongly correlated with the class to be learned, since our goal is to maximize the predictive accuracy of the induced model. On the other hand, we discourage views which are correlated with each other, since if a given view's predictive capability is covered by another then it is safe to remove this view. That is to say, we need to identify a set of views which are strongly correlated with the class, while they are uncorrelated to one another.

Next, we will discuss the selection of such a set of high quality views, in detail.

B) Heuristic Measurements

The MRC algorithm introduces a *Correlation-based View Validation (CVV)* algorithm in order to select a subset of views which are highly correlated with the target concept, but irrelevant to one another. We here employ two heuristic measures in order to identify such a set of views.

First, we need to be able to measure the "goodness" of a *subset* of views. Methods for selection of a subset of variables have been studied by many researchers [3, 4, 93, 101, 108, 121, 131, 133, 154, 244]. Such approaches aim to identify a subset of attributes for machine learning algorithms in order to improve the efficiency of the learning process. For example, Almunllim and Dieterich [3, 4] introduced a "min-features bias" method, which exhaustively searches the feature space and then finds the minimum combination of attributes that can better divide the training examples into pure classes. At each state of the feature searching process, the most promising attribute, such as the one that has the most discriminating power, is added to the current feature subset. However, a problem for this method is that the searching process may continue to add features to the final subset in order to attain consistency on the training data [101].

Following the same trend, in [154], Liu and Setiono presented a LVF algorithm for identifying a subset of attributes with high "quality". This strategy iteratively explores the entire feature space. At each exploration, the method generates a random subset of features from the entire feature space. The "quality" of the newly created feature set is then compared with that of the current best subset, in terms of a measurement called *inconsistency rate*. This measure is calculated based on the number of "matching" instances of the current feature subset and the number of examples with the most frequent class value in the "matching" instances. In addition, similar ideas which are based on rough sets theory [162, 180] have also been proposed. Nevertheless, one of the major drawbacks of both the LVF and rough set algorithms is that they are bias to attributes with many values [101].

Along the same line of research, Koller and Sahami [133] described a strategy for selecting a subset features, which can help better predicting the class. Their

approach aims to choose the subset which remains the probability distributions (over the class values) as close to that of the original features as possible. The strategy starts with all features available and then keep removing less promising attributes. That is, the algorithm tends to remove features that causes the least change between the two distributions. The algorithm requires the user to specify the number of the final attribute subset.

Besides methods which directly identify a subset of attributes, strategies which weight the usefulness of the features have also been investigated [121, 131]. In this type of approaches, each feature's "quality" is marked by a weight and then a user-specified threshold is enforced to select the final subset of features. For example, Kira and Rendell [121] proposed an algorithm which ranks the usefulness of features based on their ability to distinguish the classes.

In addition to research on the feature selection field, studies on identifying redundant graphs have also been introduced. Pearl [181] described the d-separation algorithm to compute the conditional independence relations entailed in a directed graph. Using a causal graph as input, this method can identify the dependency or partial dependency of a set of variables. In this way, the d-separation approach can be used to prune dependent variables in the graph, based on their correlation information. However, in order to apply the d-separation strategy, we need to have a probabilistic graphical models such as a Bayesian Networks [106] to map the variables into a directed graphical model and to correctly capture the probability distributions among these variables; this work is not trivial [105].

In summary, the above strategies focus on either identifying the correlation between features and class values or evaluating the dependency between features. That is, the above methods cannot fulfill our view validation requirement. In other words, they are unable to compare the "goodness" of different subsets of variables. The "goodness" measure of such a subset should be able to have a good trade-off between two potentially *contradictive* requirements. That is, on the one hand, this subset of attributes has to be strongly correlated with the class, so that it can help better predicting the class labels. Consider this correlation information (denoted

as $\overline{R_{cf}}$) is calculated by averaging correlation scores of all feature-to-class pairs. In this sense, keeping expanding the number of features in the subset increases the value of $\overline{R_{cf}}$. On the other hand, the correlation score between attributes within this subset is required to be as low as possible, so that they contain diverse knowledge. Suppose we calculate this score (denoted as $\overline{R_{ff}}$) by averaging the correlation information between all feature-to-feature pairs. In such scenario, the value of $\overline{R_{ff}}$ will be decreasing if we keep removing features from the subset.

In this thesis, in order to measure the level of such "goodness" of a given subset of views, we adapt a heuristic principle from the test theory [93]:

$$\mathcal{C} = \frac{K\overline{R_{cf}}}{\sqrt{K + K(K-1)\overline{R_{ff}}}} \tag{3.1}$$

This formula has previously been applied in test theory to estimate an external variable of interest [93, 108, 244]. In addition, Hall has adapted it into the CFS feature selection strategy [101], where this measurement aims to discover a subset of features which are highly correlated to the class but irrelevant to one another. In this heuristic equation, $\mathcal{C}$ is the heuristic "goodness" of a selected component (feature) subset; K is the number of features in the selected subset; $\overline{R_{cf}}$ calculates the average feature-to-class correlation, and $\overline{R_{ff}}$ stands for the average feature-to-feature dependence.

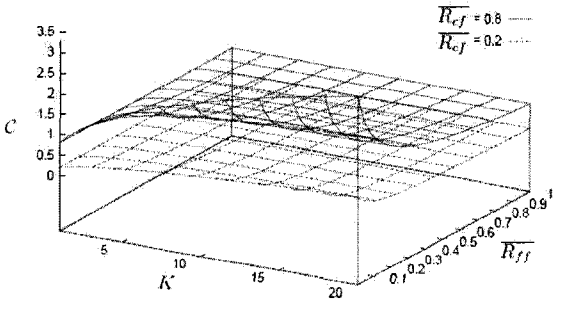


Figure 3.7: The effects among $\overline{R_{cf}}$, $\overline{R_{ff}}$, and K of the "goodness" formula $\mathcal{C}$ [101]

The reason for adapting this measurement in our thesis is based on the follow-

ing observations and conclusions, as presented in [101]. The relationship between $\mathcal{C}$, $\overline{R_{cf}}$, $\overline{R_{ff}}$, and K are illustratively plotted in Figure 3.7 [101]. In this figure, two illustrative values were assigned to $\overline{R_{cf}}$ of the Equation 3.1, and then the values of $\overline{R_{ff}}$ and K were varied in order to observe their impact on the values of $\mathcal{C}$. As plotted in Figure 3.7, the heuristic principle has the following three characteristics [101]: 1) the higher the correlation score between the components (features) and the class, the higher the score of the $\mathcal{C}$ value; 2) the lower the correlations among features, the higher the $\mathcal{C}$ value; and 3) as the number of features increases, the higher the $\mathcal{C}$ score. However, research has shown that it is unlikely that a set of features which have high correlated scores with the class while features within this set have low correlation measure with one another [93, 101]. These three characteristics state the exact goal of our view correlation evaluation procedure. Recall that we aim to identify a subset of views which are highly diverse, but strongly correlated with the class to be learned. Consider we use views as the components in the above heuristic equation. In this way, the $\overline{R_{cf}}$ will represent the average correlation information between the views and the class, and $\overline{R_{ff}}$ will calculate the average view-to-view dependence. In other words, the heuristic measurement $\mathcal{C}$ can provide us with a principle to evaluate the "goodness" level of different view sets. We, therefore, adapt this heuristic principle to identify a set of views which contain less common information and possess sufficient knowledge about the class.

Second, in order to measure the correlations between features and the class, and between features, namely $\overline{R_{cf}}$ and $\overline{R_{ff}}$ in Equation 3.1, we adopt the *Symmetrical Uncertainty ($\mathcal{U}$)* [191]. This measure is a modified *information gain* (*InfoGain*) measure [196]. It compensates for *InfoGain*'s bias toward attributes with more values. The reason for choosing this metric is as follows. Quality measures for features have been extensively studied [101, 121, 135, 191, 195, 208, 231]. The fact that some measures (such as *InfoGain* [195]) are bias toward attributes with more values has also been highlighted [135, 195, 231]. Kononenko [135] and Hall [101] have extensively examined the bias property of widely used measures for calculating the quality of features, and identified the three most acceptable biases with respect to

the number of values of the features: Symmetrical Uncertainty ($\mathcal{U}$) [191], Minimum Description Length (MDL) [208], and *Relief* [121]. Also, research in the field of feature selection [101] has shown that although the *Symmetrical Uncertainty, MDL,* and *Relief* measures provide similar behavior and results, the *Symmetrical Uncertainty* performs slightly better overall. We, therefore, can use the *Symmetrical Uncertainty* metric for calculating the correlation scores of views in this thesis.

The *Symmetrical uncertainty* is defined as follows:

Given features X and Y,

$$\mathcal{U} = 2.0 \times \left[\frac{InfoGain}{H(Y) + H(X)} \right].$$

where

$$H(Y) = -\sum_{y \in Y} p(y) \lg(p(y)),$$

and

$$\begin{aligned} InfoGain = \; & - \sum_{y \in Y} p(y) \lg(p(y))) \\ & + \sum_{x \in X} p(x) \sum_{y \in Y} p(y|x) \lg(p(y|x)) \end{aligned} \tag{3.2}$$

Note that, in classification tasks, features and classes may involve different data types such as continuous and nominal. In order to apply the $\mathcal{U}$ measurement to calculate the correlation scores between features, and between features and class, in a uniform way, it is desirable to have discretization methods to convert continuous data into nominal values. Feature discretization methods have been widely investigated in the machine learning community, since many learning algorithms only focus on learning from nominal features [67]. Popular approaches include *Equal Width Interval Binning, 1RD* [109], and *Recursive Minimal Entropy Partitioning (RMEP)* [77]. We here employ the supervised discretization method *RMEP* proposed by Fayyad and Irani [77]. This strategy uses a minimum entropy heuristic to discretize numeric features and has demonstrated its superior performance when

pre-processing data for many machine learning strategies [67, 101, 132, 220]. In the *RMEP* approach, the class entropy of candidate partitions is first calculated. Next, the one that minimizes the class entropy is then selected as the cut-off point for discretization. The algorithm recursively finds cut-off points for the two intervals of the previous split until the stopping criteria is met. In this way, the algorithm is able to find the optimal cut-off points to divide the range of observed continuous value into a set of bins (intervals) [77].

After having chosen the heuristic measures, the view validation process needs one more element: *view features*. In the CVV method, the correlation score between two *view learners* is obtained through calculating the correlation information between their corresponding *view features*. We will discuss this element next.

C) View Feature

The CVV strategy first uses view features to represent their corresponding view learners. Next, it calculates the correlation scores between these view features, in order to approximate the correlation information between the view learners. The idea here was inspired by the success of meta-learning algorithms [33, 95]. In a meta-learning setting such as Stacked Generalization [33, 159, 221, 233], knowledge of the base learners is conveyed through their predictions in the meta level. Basing on this observation, the CVV approach uses class probabilistic predictions generated by a given view learner as its view features. Here the class probabilities serve as the *confidence measure* [221] for the prediction made by a given view learner. This is described next in detail.

Definition 3.3-1 (View feature). Let $\{V^1, \cdots, V^n\}$ be n views to be validated. Given a *view validation data set* $\mathcal{T}_v$ with m labels $\{y_1, \cdots, y_m\}$. For each instance t (with label L) in $\mathcal{T}_v$, each view learner is called upon to produce predictions $\{f_{V^i}^{y_k}(t)\}$ ($i \in \{1, \cdots, n\}$ and $k \in \{1, \cdots, m\}$) for it. Here, $f_{V^i}^{y_k}(t)$ denotes the probability that instance t belongs to class y_k, as predicted by view learner over view V^i. In this way, a *view validation example* which consists of a set of $P_{V^i}^{y_k}(t)$

(each describes the hypothesis knowledge possessed by each view), along with the original class labels y_k, is constructed.

That is, V^1, for example, is described and represented by features $\{f_{V^1}^{y_k}(t)\}$ ($k \in \{1, \cdots, m\}$) in a *view validation example*. We call $\{f_{V^1}^{y_i}(t)\}$ *View Features* of view V^1. By doing so, a set of view validation examples $\mathcal{T}_v$ are created. Each instance t consists of a set of *View Features* to describe the corresponding view, along with a class label of the instance.

View features generated by view **LOAN**		View features generated by view **ORDER**		Class label
0.930073	0.069927	0.212184	0.787816	Bad
0.661307	0.338693	0.313489	0.686511	Good
0.898148	0.101852	0.809461	0.190539	Good
0.232177	0.767823	0.469726	0.530274	Bad

Figure 3.8: Sample view feature set generated by the *Loan* and *Order* views

Example 3.3-2 (An example of generating view features). Let's continue the running example described in Example 3.2-6. In that example, we have constructed two views, namely *Loan* and *Order* from the example database (shown in Figure 1.3). Note that, this example database has seven background relations and one target relation. Eight views, therefore, should have been formed, each is separately constructed from one of the eight tables of the database. For simplicity, we here only presented two views, i.e. *Loan* and *Order*. After finishing constructing the two views, the MRC algorithm separately builds two view learners from these two views (denoted as V^{loan} and V^{order}). In this learning task, the target variable *Status* has two values, namely *good* and *bad*. Hence, for each tuple t in the evaluation data set, view learner V^{loan} generates two probabilistic predictions f_{good}^{loan} and f_{bad}^{loan} ($f_{good}^{loan} + f_{bad}^{loan} = 1$). Similarly, two view features f_{good}^{order} and f_{bad}^{order} are generated for *Order* view as well. Figure 3.8 shows four (4) examples generated by views *Loan* and *Order*, where view features generated by the former are in the green cells and those created by the latter in the blue cells. Each of the four examples also

contains the original class labels of the evaluation data.

In this way, a subset of view features can then be selected based on measure C. Next, we will discuss this selection process in detail.

D) The CVV Approach

After finishing constructing the view feature set, the CVV algorithm subsequently ranks *view feature* subsets according to the correlation-based heuristic evaluation measure C. It searches all possible view feature subsets, and constructs a ranking on them. The best ranking subset will be selected, i.e. the subset with the highest value of C.

To search the feature space, two widely used heuristic approaches, namely *hill climbing* and *best first*, are often employed [101, 128, 129]. Hill climbing is a greedy method. In each step of the search, it considers the heuristic value of the feature subset by adding one more feature (forward selection) or removing one feature from the current set (backward elimination) [161]. In this kind of greedy search, the algorithm never back-tracks the visited paths. Like the hill climbing strategy, the best first method explores the feature space by adding or removing features from the current feature subset. However, the best first allows the search to back-track to a more promising subset (and then continue the search from there) if the current search path is determined less promising. The best first search strategy [94, 210] has demonstrated its successes in many feature selection approaches [96, 101, 129]. Also, Hall [101] has experimentally studied the performance of the best first and hill climbing in the CFS feature selection algorithm and observed that the best first performed slightly better than the hill climbing. We, therefore, use the best first search strategy in the CVV method.

The best search strategy initiates with an empty set of features, and keeps expanding with one more feature. In each round of the expansion, the best feature subset, namely the subset with the highest "goodness" value C will be chosen to keep expanding in the same way. Additionally, the best search traversal keeps a

ranking of all its visited subsets. If the current expansion results in no improvement in terms of "goodness" value, the search can go back to the next best path and use it to expand its feature set.

Often, a stopping criteria is imposed to a best first search in order to stop it from exploring the entire search space. The CVV algorithm will terminate the search if a number of consecutive non-improvement expansions occur. Hall heuristically set this number to five (5) in the CFS feature selection method and has achieved very good results [101]. Also, this number is used as a default value for the wrapper feature selector implemented in the MLC++ machine learning algorithms library [130] and the feature selection method in the Weka data mining softwares collection [232]. We here use the same number for our view validation strategy.

Note that, the best first view search here requires $(m^2 - m)/2$ (m is the number of view features) computational operations in the worst case (an exhaustive search). However, the use of the heuristic stopping criterion can significantly reduce the probability of the searching of the entire feature space [101]. Moreover, the number of view features is often small, which makes the search computation affordable even for a complex database schema. The number of view features is a *linear* function with respect to the number of relations in the database. For instance, for a relational database with n tables, the number of view features for a k-class problem will be $m = kn$. Also, our experimental studies, which will be presented in the experimental section (i.e. Section 3.4.3), have shown that this validation process is very fast, in terms of computational time needed.

In the CVV method, views are selected based on the final best subset of view features, which are considered highly correlate to the target concept to be learned. If a view has no view features to be considered strongly correlate to the class to be learned, it means that knowledge possessed by this view is not important for the learning. Thus, it makes sense to ignore this view. Therefore, the CVV algorithm selects a view *if* and *only if* any of its *view features* appears in the final best ranking subset of the *view feature* selection procedure.

Example 3.3-3 (View Validation). Consider a final view feature subset $\{ f_{V^1}^{y_1}(t), f_{V^1}^{y_1}(t), f_{V^4}^{y_2}(t) \}$. This subset of view features means only knowledge from views V^1 and V^4 really contributes to build the final model. Thus views V^1 and V^4 are selected by the *Correlation-based View Validation* method. Views other than V^1 and V^4 are ignored because they are considered weakly relevant to the target concept to be learned.

Algorithm 2 Correlation-based View Validation

Input: Candidate view set $\{V_d^1, \cdots, V_d^m\}$,
View validation data set $\mathcal{T}_v$.

Output: View set $\{V^1, \cdots, V^k\}$ $(k \leq m)$.

1: Let View Set $V = \emptyset$;
2: Remove candidate views with view learners' accuracy less than 50% from $\{V_d^1, \cdots, V_d^m\}$, forming view set V';
3: Generate a validation data set $\mathcal{T}_v'$, using $\mathcal{T}_v$ and V';
4: Select a view feature set $\mathcal{A}'$ from $\mathcal{T}_v'$;
5: **for each** view V^i $(V^i \in V')$ which has at least one attribute in $\mathcal{A}'$ **do**
6: V.add(V^i);
7: **end for**
8: **return** V.

In summary, the CVV method is shown in Algorithm 2. As presented in Algorithm 2, the *View Validation* element first removes view learners with accuracy less than 50%. Next, it generates a *view validation data set*, where hypothesis knowledge possessed by multiple views are represented by *view features*. Thirdly, based on the heuristic correlation evaluation measure as described in Equation 3.1, the best first search strategy is employed to select the best subset of view features. Finally, views are filtered out if none of its view features appear in the final view feature subset.

3.3.2 View Combination

The last step of the MRC approach is to construct the final classification model, using the trained view learners.

Strategies for combining models have been investigated by many researchers. The most popular are those such as bagging [27], boosting [81] and stacking [234]. Voting approaches usually only make sense if the learners perform comparably well, otherwise poor predictors may be transferred into a poorly performing model [112, 232]. View learners in the MRC strategy learn only from their own feature set. In addition, each view learner may be trained by instances which relate to different subset of the target tuples. We, therefore, have no reason to strongly believe that the resultant view learners perform comparably well. In other words, we cannot guarantee preventing some over-confidence voters from misleading the final model. Consider the following situation. Although some view learners have high predictive performance, but they are trained by a data set which is only related to a small part of the target tuples in the target relation. Thus, we need to prevent these view learners from becoming over-confidence and thus misleading the final model since they may have very little knowledge on the entire target concept. Furthermore, we also need to avoid some high accurate view learners from being "blocked" by majority "weak" learners. For example, in the MRC algorithm, we may have a view learner which consistently predicts the target tuples correctly from its own view, while this view only relates to a specific part of the target tuples. In this scenario, consider that a voting strategy is applied. The prediction capability of this view can be shadowed by other views if these views have no knowledge on this subset of target tuples.

As an example, suppose we have three view learners, each learns from a separate region of the target relations, as shown in Figure 3.9. In other words, view learners f^1, f^2, and f^3 separately learn from regions *region1, region2, and region3* of the target relation, respectively. From Figure 3.9, one can see that each view learner learns from a different part of the target tuples and features. In such case, a

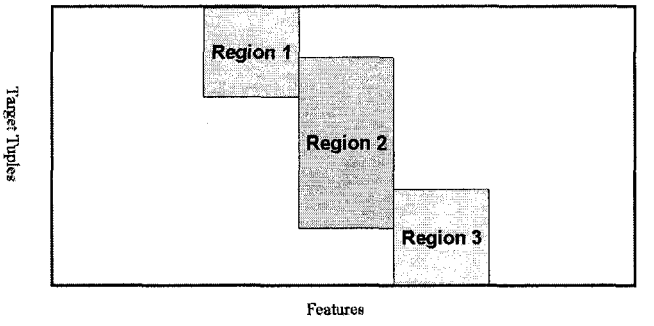


Figure 3.9: Various learning regions in the target relation

majority voting schema to combine these three view learners will become difficult. For instance, view learner f^2 may consistently classify all tuples from *region2*, while the other two view learners have little information about this region. In such case, a majority voting does not make sense. As a real-world example, Figure 3.10 visualizes the data from the *Client*, *District*, and *Credit Card* views of the banking database as introduced in Figure 1.3. From this example, one can see that, unlike the *District* view, the views *Client* and *Credit Card* have little knowledge about the target tuples with *Tuple ID* larger than 6000. In this scenario, applying majority voting to the region with *Tuple ID* larger than 6000 is prohibitive to the predictivity capability of the *District* view.

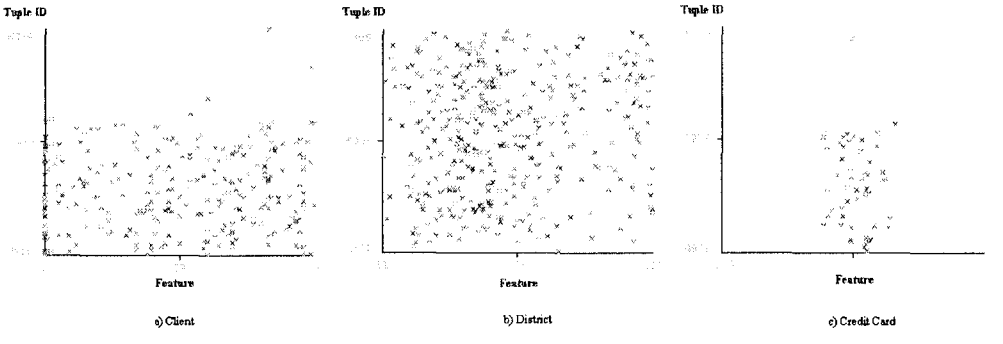


Figure 3.10: Real data sets from the sample database

On the other hand, a meta learner method [34, 95, 224, 234] is designed to learn from knowledge generated by base classifiers, using another learning method

(called *meta-learner* or *combiner*) [33]. The meta-learner aims to learn the *relationship* between predictions from base classifiers, and to learn which classifiers are the reliable ones [232]. For example, a meta-learner can make good use of a situation where a base classifier consistently make accurate predictions regardless the disagreements from other base classifiers [33]. Consider the above example as presented in Figure 3.9 again. A simple meta-learning function which relies on the view learner constructed from *region2* is able to correctly classify a major part of all target tuples. Based on these observations, and the fact that we use multiple different relations to form multiple views, the meta-learning method is applied to the multi-view learning framework, as discussed next.

In meta-learning schemes, a learning algorithm is usually used to construct the function that combines the predictions of the individual learners. This combination process contains two steps. Firstly, a meta training data set is generated. Each instance of the meta training data consists of the class label predictions (denote the probabilities that the instance belongs to different classes), made by the individual learner on a specific training example, along with the original class label for that example. Secondly, a meta-learner is trained using the meta data constructed to achieve a strong predictive performance, as the final classification model.

In the last stage of the learning, the MRC approach returns a set of view learners, along with a meta learner which knows how to combine these multiple learners to achieve a strong predictive performance.

3.3.3 Summary

In summary, the MRC strategy (Shown in Algorithm 1) works as follows. The MRC method initially propagates and aggregates information to form multiple views from a relational databases. Subsequently, Algorithm 2 is called upon to identify a subset of uncorrelated views. After doing so, the MRC algorithm initiates the view learners to learn the target concept from each of these uncorrelated views. In this way, each view learner constructs a different hypothesis based on the data it

is given. Finally, multiple view learners are then combined to form a final model.

The next section presents our empirical evaluations.

3.4 Experimental Study

This section provides the results obtained for the MRC algorithm on benchmark real-world databases. These results are presented in comparison, in terms of predictive performance achieved and running time needed, with four other well known multirelational data mining systems, namely the FOIL rule learner [199], CrossMine method [242], TILDE first-order logical trees [20], and RelAggs algorithm [136], along with a "baseline" "flattening" approach (denoted as SimFlat).

Recall from Chapter 2 that, these logic-based "upgrading" methods and "flattening" algorithms employ different strategies to deal with structured data. The FOIL algorithm is commonly used when benchmarking the performance of relational data mining methods. The CrossMine method is a recent addition. It extends the FOIL approach and has demonstrated its high scalability and accuracy when mining relational databases [242]. In contrast to the general-to-specific searching approach as employed by FOIL and CrossMine algorithms, the divide-and-conquer searching technique is applied in the TILDE method [20] to build a logical decision tree. On the other hand, the RelAggs algorithm is a state-of-the-art "flattening" approach, where aggregate operators are used to transform multiple relations into a single table in order to be able to use propositional algorithms for the learning.

In order to provide a "baseline" approach, we introduce the SimFlat strategy. In contrast to considering advanced aggregation techniques (such as function dependency among attributes) in the RelAggs strategy, the SimFlat method employs the same aggregation calculations as that of the MRC algorithm to convert multiple relations into a universal single table. That is, the SimFlat strategy uses the same aggregate functions and follows the same join chains used by the MRC method to "flatten" relations. The goal of the SimFlat strategy is therefore to provide a

universal flat file which consists of all features used by individual views in the MRC strategy. In this way, the performance achieved by learning from multiple feature sets (by the MRC algorithm) can be compared to that of learning from all features available.

3.4.1 Methodology

Six learning tasks, derived from four standard real-world databases, were used to evaluate our algorithm. The four benchmark databases, namely the Financial, Mutagenesis, Thrombosis, and Warehouse databases, come from different application domains, have variant relational structures, consist of different numbers of tuples in the entire database and in the target relation, and present varying degree of class distribution in the target relation. In addition, in order to test how different numbers of tuples in the target relation (with the same database schema) affect the performance of the MRC algorithm, we derived three learning tasks from the Financial database. Each of these three tasks has a different number of target tuples but shares the same background relations.

We implemented the MRC algorithm using Weka [232]. Also, our experiments applied C4.5 decision trees [196] and Naive Bayes probabilistic classifiers [115] as view learners and meta learners of the MRC algorithm. The C4.5 decision tree learner was used due to its de facto standard for empirical comparisons. In addition, Naive Bayes classifiers were chosen because of their sensitivity to the changes of the input attributes [65]. The default settings of these two learning methods were used. Each of these experiments produces accuracy results using ten-fold cross validation. The MRC algorithm, SimFlat strategy, TILDE approach, RelAggs algorithm, and CrossMine method were run on a 3 GHz Pentium 4 PC with 1 GByte of RAM running Windows XP and MySQL. The FOIL approach was run on a Sun4u machine with 4 CPUs. For each data set and system we report the average running time of each fold. The "flattening" preprocessing of the SimFlat method, using SQL, takes considerable manual time and effort and assumes an expert level

of database programming skills. We therefore do not provide the running time for this method. For comparison purpose, our experiments here applied the same aggregation functions used by the RelAggs method as implemented in Weka.

Table 3.1: Summary of the data sets used

Data Set	#tuples in the target	#related relations	target class distribution	# tuples in task
MUT188	188	3	125:63	15,218
F682AC	682	7	606:76	76,264
F400AC	400	7	324:76	75,982
F234AC	234	7	203:31	75,816
ECML98	7,329	8	3705:3624	197,478
THROM	770	4	695:75	4,780

3.4.2 The Data Sets Used for Experimental Comparison

Mutagenesis Database

Our first experiment (denoted as MUT188) was conducted against the Mutagenesis data set [218]. This benchmark data set is composed of the structural descriptions of 188 Regression Friendly molecules that are to be classified as mutagenic or not (Of the 188 instances 125 tuples are positive and 63 are negative). The background relations of this learning problem consist of descriptions about the atoms and bonds that make up the molecules, which include 4893 atoms and 5244 bonds. The *Atom* and *Bond* relations link to the target relation *Molecule* through the *Molecule_Atom* relation which only contains key attributes. A summary of the characteristics for the learning data set is given in Table 3.1. For this database, 3 views, namely *molecule, bond*, and *atom* are constructed by the MRC algorithm.

Financial Database

Our second experiment was conducted against the previously introduced Financial database, which was used in the PKDD 1999 discovery challenge [13]. The database was offered by a Czech bank and contains typical business data. The schema of

the Financial database is shown in Figure 1.3. Recall that the original database is composed of eight tables. The target table, i.e. the Loan table consists of 682 records, including a class attribute status which indicates the status of the loan, i.e. A (finished and good), B (finished but bad), C (good but not finished), or D (bad and not finished). The background information for each loan is stored in the relations *Account, Client, Order, Transaction, Credit Card, Disposition* and *Demographic*. All background relations relate to the target table through directed or undirected foreign key chains, as shown in Figure 1.3. This database provides us with three different learning problems. Our first learning task is to learn if a loan is good or bad from the 234 finished tuples. The second learning problem attempts to classify if the loan is good or bad from the 682 instances, regardless of whether the loan is finished or not. Our third experimental task uses the Financial database as prepared in [242], which has only 400 examples in the target table. The authors sampled the *Transaction* relation, since it contains an extremely large number of instances and discarded some positive examples from the target relation to make the learning problem more balanced. A summary of the three learning data sets is also presented in Table 3.1, where F234AC, F682AC and F400AC denote the first, second, and third learning tasks respectively. For this database, eight views, namely, the views *loan, account, client, order, transaction, credit card, disposition*, and *demographic* were constructed for each task. Note that for comparison purpose, all these three learning tasks use the same background relations as prepared in [242].

Thrombosis Database

Our next experiment used a database derived from the Thrombosis database used for the PKDD 2001 Discovery Challenge [46]. This database is originally organized using seven relations. We used the *Antibody_exam* relation as the target table, and *Thrombosis* as the target concept. This concept describes the different degrees for Thrombosis, i.e. None, Most Severe, Severe, and Mild. The target table has 770 records describing the results of special laboratory examinations performed on pa-

tients. Our task here is to determine whether or not a patient is thrombosis free. For this task, we include four relations for our background knowledge, namely *Patient_info, Diagnosis, Ana_pattern*, and *Thrombosis*. All four background relations are linked to the target table by foreign keys. Therefore, for this data set, five different views are constructed by the MRC algorithm. A summary of this data set is shown as part of Table 3.1 (identified by Throm).

ECML98 Database

Our last experiment used the database for the ECML 1998 Sisyphus Workshop [140]. This database was extracted from a customer data warehouse of a Swiss insurance company [120]. The task of this learning problem is to classify 7,329 households of class 1 or 2. Eight background relations are provided for the learning task. They are stored in tables *Eadr, Hhold, Padr, Parrol, Part, Tfkomp, Tfrol* and *Vvert*, respectively. In this experiment, we used the new star schemes prepared in [140]. A summary of the this learning data set is also presented in Table 3.1 (denoted as ECML98).

3.4.3 Experimental Results

In this section, we conducted three experiments to examine the performance of the MRC method, in terms of both the accuracy obtained and the running time needed. These three experiments used different propositional learning algorithms as the view learners and meta learners for the MRC algorithms in order to evaluate how different propositional algorithms impact the performance of the MRC algorithm.

Experiment #1: Using Decision Trees as propositional learners, view learners, and meta learners

In the first experiment, we examine the performance of the MRC algorithms in terms of accuracy obtained and running time needed. In this experiment, C4.5 decision trees were used by the RelAggs and SimFlat approaches as the propositional

Table 3.2: Accuracies obtained using C4.5 as view learners and meta learners (%)

Data Set	RelAggs	SimFlat	FOIL	CrossMine	TILDE	MRC (with VV)	MRC (without VV)
MUT188	85.1 (-1.6)	**88.3** (+1.6)	85.7 (-1.0)	85.7 (-1.0)	85.6 (-1.1)	86.7	86.7 (±0.0)
F682AC	92.1 (-1.3)	92.4 (-1.0)	83.9 (-9.5)	90.3 (-3.1)	88.9 (-4.5)	93.4	**93.7** (+0.3)
F400AC	**89.0** (+1.7)	87.5 (+0.2)	72.8 (-14.5)	85.8 (-1.5)	81.0 (-6.3)	87.3	87.8 (+0.5)
F234AC	**90.2** (+0.5)	89.7 (+0.0)	74.4 (-15.3)	88.0 (-1.7)	86.8 (-2.9)	89.7	88.0 (-1.7)
ECML98	**88.0** (+0.4)	87.9 (+0.3)	61.5 (-26.1)	85.3 (-2.3)	53.7 (-33.9)	87.6	87.3 (-0.3)
THROM	**100.0** (±0.0)	90.4 (-9.6)	**100.0** (±0.0)	90.0 (-10.0)	90.4 (-9.6)	**100.0**	**100.0** (±0.0)

Table 3.3: Running time required using C4.5 as view learners and meta learners (in seconds)

Data Set	RelAggs	FOIL	CrossMine	TILDE	MRC (with VV)	MRC (without VV)
MUT188	12.80	2.30	**1.00**	1.40	3.26	3.52
F682AC	89.54	14173.20	11.60	1051.90	**5.59**	5.60
F400AC	60.00	8454.80	8.10	650.00	**2.83**	**2.83**
F234AC	40.80	4675.80	5.00	568.30	**1.60**	1.64
ECML98	1703.58	2456.80	570.90	1108.60	**418.37**	424.43
THROM	0.72	**0.70**	0.90	75.70	1.03	1.03

learners and by the MRC algorithm as the view learners and meta learners.

We present the predictive accuracy obtained for each of the six learning tasks in Table 3.2, where *MRC with VV* and *MRC without VV* stand for the MRC approaches with view validation applied and without view validation, respectively. For each data set in Table 3.2, the highest results are highlighted in *bold*. In addition, in the parentheses of this table, we provide the accuracy *gains (denoted by "+")* or *loss (denoted by "-")* of each approach, compared to that of the MRC algorithm with view validation applied. To evaluate the performance of the MRC strategy in terms of run time, we also provide the running time needed (in seconds) for each learning tasks in Table 3.3, where the best results for each data set are also highlighted in *bold*.

Discussion of Experiment #1

The predictive performance results, as presented in Table 3.2, show that the MRC algorithm appears to consistently reduce the error rate for almost all of the data

sets, when compared to the logic-based FOIL, CrossMine, and TILDE methods. The only exception is that the MRC and the FOIL algorithm achieved the same predictive performance against the THROM database. In addition, our results, as shown in Table 3.2, also indicate that, in many cases the error rate reduction achieved by the MRC approaches is large. For example, the MRC approaches reduced the error rates by at least 9% against 1) the F682AC, F400AC, F234AC, and ECML98 data sets, compared to the FOIL method; 2) the THROM data set, in comparison with the CrossMine algorithm; and 3) the ECML98 and THROM data sets when comparing with the TILDE approach.

When considering the comparison with the "flattening" based RelAggs and SimFlat strategies, the MRC algorithm conducted comparable predictive performance. As shown in Table 3.2, the results indicate that the RelAggs method performed a bit better in three of the six data sets, but was less accurate in two of the six cases, when compared with the MRC approach. Also, the predictive performance of the MRC approach was similar with that of the SimFlat algorithm. However, the experimental results showed that the differences of accuracy achieved by these three algorithms are less than 2%, except for the THROM data set. In the THROM data set, the SimFlat approach yielded a 9.6% of accuracy loss, compared to that of the MRC algorithm. Our further analysis suggests that, this significant performance lost was due to the fact that, the flattening process of the SimFlat approach resulted in a large number of NULL values in the converted single table, and these NULL values then confused the propositional learners.

In terms of running time needed, one can see from the experimental results (shown in Table 3.3) that the MRC methods achieved very promising outcomes, when compared to the other four well-known algorithms. The MRC algorithm meaningfully reduced the running time needed for most of the cases. For example, against four of the six data sets, namely the F682AC, F400AC, F234AC, and ECML98 data sets, the MRC methods were at least 25 times faster than the FOIL and TILDE algorithms. Comparing to the CrossMine strategy, the MRC approaches were at least twice as fast as the CrossMine method when learning against

the F682AC, F400AC, and F234AC data sets. Not surprisingly, the RelAggs approaches were time consuming in many cases due to the flattening process, when comparing with the MRC strategy. For example, against the F234AC, F400AC, F682AC, ECML98, and MUT188 data sets, the MRC algorithm was 25, 21, 16, 4, and 3.9 times faster than the RelAggs strategy, respectively. In short, the results against the six data sets suggest that 1) the logic-based CrossMine method was efficient to learn from multiple relations, compared to the FOIL and TILDE strategies; 2) the MRC algorithm, in general, was much faster than the three well-know relational learning methods, namely the RelAggs, FOIL, and TILDE, and required very comparable running time when compared with the CrossMine algorithm.

When considering the impact of the view validation process in the MRC algorithm, the experimental results, as shown in Tables 3.2 and 3.3 imply that the MRC algorithms, with and without the view validation applied, resulted in little difference in terms of accuracy achieved and running time needed when C4.5 decision trees were used as the meta learners.

Experiment #2: Using Decision Trees as propositional learners, view learners, and Naive Bayes as meta learners

The second experiment aims to investigate the impact of the meta learning algorithms on the MRC approach. In this experiment, we used C4.5 decision trees as the propositional learners of the RelAggs and SimFlat approaches and as view learners of the MRC algorithm (as we did in experiment #1). However, we applied Naive Bayes as the meta learners of the MRC algorithm in this setting, instead of using C4.5 decision trees.

We present the predictive accuracy obtained for each of the six learning tasks in Table 3.4 and running time needed (in seconds) in Table 3.5. Note that in this experiment, the RelAggs, SimFlat, FOIL, TILDE, and CrossMine methods have the same performance as that in the experiment #1, since the change of the meta learners did not affect them.

Table 3.4: Accuracies obtained using C4.5 as view learners and Naive Bayes meta learners (%)

Data Set	RelAggs	SimFlat	FOIL	CrossMine	TILDE	MRC (with VV)	MRC (without VV)
MUT188	85.1 (-3.8)	88.3 (-0.6)	85.7 (-3.2)	85.7 (-3.2)	85.6 (-3.3)	88.9	**89.4** (+0.5)
F682AC	92.1 (-1.3)	92.4 (-1.0)	83.9 (-9.5)	90.3 (-3.1)	88.9 (-4.5)	**93.4**	91.5 (-1.9)
F400AC	**89.0** (+2.1)	87.5 (+0.6)	72.8 (-14.1)	85.8 (-1.1)	81.0 (-5.9)	86.9	84.3 (-2.6)
F234AC	**90.2** (+0.9)	89.7 (+0.4)	74.4 (-14.9)	88.0 (-1.3)	86.8 (-2.5)	89.3	82.9 (-6.4)
ECML98	**88.0** (+0.8)	87.9 (+0.7)	61.5 (-25.7)	85.3 (-1.9)	53.7 (-33.5)	87.2	86.4 (-0.8)
THROM	**100.0** (±0.0)	90.4 (-9.6)	**100.0** (±0.0)	90.0 (-10.0)	90.4 -(9.6)	**100.0**	**100.0** (±0.0)

Table 3.5: Running time required using C4.5 as view learners and Naive Bayes meta learners (in seconds)

Data Set	RelAggs	FOIL	CrossMine	TILDE	MRC (with VV)	MRC (without VV)
MUT188	12.80	2.30	**1.00**	1.40	3.31	3.13
F682AC	89.54	14173.20	11.60	1051.90	5.74	**5.32**
F400AC	60.00	8454.80	8.10	650.00	2.88	**2.77**
F234AC	40.80	4675.80	5.00	568.30	1.64	**1.62**
ECML98	1703.58	2456.80	570.90	1108.60	**449.37**	450.37
THROM	0.72	**0.70**	0.90	75.70	1.00	0.97

Discussion of Experiment #2

The experimental results, as presented in Tables 3.4 and 3.5, show that the MRC algorithms had consistent performance in terms of predictive accuracy obtained and running time needed, regardless the meta learning algorithms used, namely no matter if the C4.5 decision trees or Naive Bayes were applied as the meta learners.

In addition, the experimental results, as shown in the last two column of Table 3.4, implied that the view validation process improved the accuracy achieved by the MRC algorithms when Naive Bayes were used as meta learners. For example, in five of the six data sets, the view validation assisted MRC algorithm to improve or achieve equal accuracies. As shown in Table 3.4, against the F234AC and F400AC data sets, the view validation helped the MRC algorithms to achieve accuracy gains by 6.4% and 2.6%, respectively. Only against the MUT188 data set, the MRC algorithm with view validation obtained slightly lower accuracy than that of the MRC algorithm without the view validation (lower by only 0.5%).

In terms of running time needed, as shown in Table 3.5, the MRC algorithms with and without the view validation resulted in little difference. These results suggest that the view validation processes of the MRC algorithm were very fast.

In summary, these results indicate that when using Naive Bayes, which is unstable in regard to changes in the features used, as meta learning algorithms, the view validation component was able to improve the predictive performance of the MRC algorithm. This outcome suggests that the view validation process was able to successfully remove the uncorrelated views. Theoretically, the basic assumption made by Naive Bayes is of feature independence [102]. Research has shown that correlations between features degrade the predictive performance of the Naive Bayes methods [66, 146]. Our experimental results here imply that after successfully identifying and removing correlated views, the MRC framework was able to generate meta data which are less correlated, thus resulting in improving the predictive performance of the final model induced by the Naive Bayes methods. This implication also justifies the experimental results observed in Experiment#1, where C4.5 decision trees were used for the meta learners. That is the use of decision trees as meta learners had little impact on the performance of the MRC methods, in terms of accuracy obtained. The C4.5 decision tree [196] aims to find the best attribute for each tree split, and thus it is less sensitive to the dependency between the features, compared to the Naive Bayes approaches.

Experiment #3: Using Naive Bayes as propositional learners, view learners, and Decision Trees as meta learners

In the last experiment, we aim to evaluate the performance impact of the MRC algorithm when different propositional learning methods are applied as view learners. Contrary to the settings in the experiments #1 and 2, we here used Naive Bayes as the propositional learners of the RelAggs and SimFlat approaches, and as the multiple view learners of the MRC algorithm (where the C4.5 decision trees were employed as meta learners).

Table 3.6: Accuracies obtained using Naive Bayes as view learners and C4.5 as meta learners (%)

Data Set	RelAggs	SimFlat	FOIL	CrossMine	TILDE	MRC (with VV)	MRC (without VV)
MUT188	85.1 (-2.2)	86.2 (-1.1)	85.7 (-1.6)	85.7 (-1.6)	85.6 (-1.7)	**87.3**	85.8 (-1.5)
F682AC	73.8 (-19.2)	71.4 (-21.6)	83.9 (-9.1)	90.3 (-2.7)	88.9 (-4.1)	**93.0**	92.4 (-0.6)
F400AC	73.3 (-15.7)	69.8 (-19.2)	72.8 (-16.2)	85.8 (-3.2)	81.0 (-8.0)	89.0	**89.8** (+0.8)
F234AC	85.9 (-5.6)	75.6 (-15.9)	74.4 (-17.1)	88.0 (-3.5)	86.8 (-4.7)	**91.5**	91.0 (-0.5)
ECML98	81.4 (-1.7)	50.5 (-32.6)	61.5 (-21.6)	**85.3** (+2.2)	53.7 (-29.4)	83.1	83.0 (-0.1)
THROM	98.4 (-1.6)	86.0 (-14.0)	**100.0** (±0.0)	90.0 (-10.0)	90.4 (-9.6)	**100.0**	**100.0** (±0.0)

Table 3.7: Running time required using Naive Bayes as view learners and C4.5 as meta learners (in seconds)

Data Set	RelAggs	FOIL	CrossMine	TILDE	MRC (with VV)	MRC (without VV)
MUT188	12.80	2.30	**1.00**	1.40	3.14	3.12
F682AC	90.50	14173.20	11.60	1051.90	4.64	**4.52**
F400AC	60.00	8454.80	8.10	650.00	2.47	**2.42**
F234AC	40.80	4675.80	5.00	568.30	1.38	**1.30**
ECML98	1704.70	2456.80	570.90	1108.60	**300.29**	352.37
THROM	0.72	**0.70**	0.90	75.70	0.95	0.99

We present the predictive accuracy obtained for each of the six learning tasks in Table 3.6 and running time needed (in seconds) in Table 3.7. Note that in this experiment, the FOIL, TILDE, and CrossMine have the same performance as that of the experiments #1 and 2 since they do not dependent on the use of the propositional and meta learners.

Discussion of Experiment #3

Our results, as presented in Table 3.6, surprisingly show that RelAggs and SimFlat approaches appears to performance very poorly in terms of accuracy obtained, compared to that of experiments #1 and 2, where C4.5 decision trees were used as propositional learners. For example, against the F682AC, F400AC, ECML98, and F234AC data sets, the accuracy lost of the RelAggs algorithm with Naive Bayes as propositional learners were 18.3%, 15.7%, 6.6%, and 4.3%, respectively, compared to that of the RelAggs approach with C4.5 decision trees as propositional learners (presented in experiments #1 and 2). In addition, the "flattening" approach Sim-

Flat also experienced large accuracy lost against almost all the data sets, due to the use of Naive Bayes as propositional learners (instead of C4.5 decision trees). For example, the accuracy lost were at least 14% against four of the six learning data sets. Our further analysis of the experimental results implies that the significantly predictive performance lost of the "flattening" based approaches were due to the large numbers of newly created aggregation attributes in the converted flat files. In addition, a close look at the THROM data set show us that, there were a large number of NULL values in the resulting flat file, which further confused the Naive Bayes learning algorithms.

Contrary to the large loss of predictive performance conducted by the RelAggs and SimFlat methods, the MRC algorithm resulted in very comparable predictive accuracy regardless of the propositional learners used. Against the THROM, F400AC, and F234AC data sets, the MRC algorithm obtained equal accuracies or improved the predictive performance, compared to the MRC methods which used decision trees as view learners. Only against the MUT188, F682AC, and ECML98 data sets, the Naive Bayes learners brought small performance lost to the MRC algorithms. Further analysis of the results suggests that such consistent performance benefits from combining multiple views.

Importantly, the results of this experiment (shown in Table 3.6) indicate that the MRC algorithm outperformed the two "flattening" based and three logic-based approaches, in terms of accuracy obtained, against almost all data sets. The results show that the MRC strategy achieved the highest accuracy in 28 out of the 30 cases. One exception is against the ECML98 data set, where slightly lower accuracy (compared to that of the CrossMine method) was obtained by the MRC methods (lower by only 2.2%); another exception is against the THROM data set, where the FOIL method achieved the same accuracy of 100% as that of the MRC algorithms.

Promisingly, the experimental results as presented in Table 3.6 show that the MRC algorithms significantly benefited from the framework of learning from multiple view. Recall that, the flat file conducted by the SimFlat approach consists of attributes from all views of the MRC strategy. That is to say, in the MRC

algorithm, a set of feature set (each feature set corresponds to a view in the MRC strategy) were separately employed to learn the target concepts. On the other hand, the SimFlat approach combined all sets of feature set into a flat file, and then learned from all these features available. When comparing the accuracy resulted from the SimFlat and MRC algorithms, the results presented in Table 3.6 show that the MRC approach meaningfully outperformed the SimFlat method against almost all six data sets. For example, against the ECML98, F682, F400AC, F234AC, and THROM data sets, the MRC algorithm with view validation improved the accuracy over the SimFlat method by 32.6%, 21.6%, 19.2%, 15.9%, and 14.0%, respectively. These results imply that, in this experiment, the learning of using cooperation of multiple feature sets can significantly improve the predictive performance gains over that of using all features available.

In terms of running time needed, the experimental result (in Table 3.7) show that the propositional learning methods do not have a large impact on the RelAggs, SimFlat, and MRC algorithms. In addition, the MRC algorithms resulted in little difference regardless the use of different view learners and meta learners as well.

Summary of the three experiments

In summary, the three experimental results (shown in Tables 3.2, 3.3, 3.4, 3.5, 3.6, and 3.7) indicate that the MRC approach achieves promising results in comparison with two flattening-based and three logic-based relational learning techniques, when evaluated in terms of overall accuracy obtained and run time needed. Our results also imply that the MRC strategy benefits from learning from multiple feature sets, being able to directly apply state-of-the-art single table learning algorithms, and excluding the need of "propositionalization".

More specifically, results from experiments #1 and #2 imply that the view validation process embedded in the MRC method was able to successfully remove the highly correlated views. The removal may result in better predictive performance of the final model when the meta learners are sensitive to the redundant views. In

addition, the results also suggest that the view validation process was very fast in terms of running time required.

When considering results from experiments #1 and #3, one may conclude that the large number of features, the dependency between the aggregation features, and the large amounts of NULL values in the converted flat file confused some single-table learning methods. In particular, a large impact was witnessed when employing learning methods such as Naive Bayes, which are sensitive to the NULL values and correlation values in the feature set, leading to the loss of predictive performance. On the other hand, the MRC framework yielded robust models, in terms of predictive performance, regardless of the view learners employed. These results suggest that the MRC method has benefited from excluding the "flattening" process.

Importantly, results from experiments #1, #2, and #3 indicate that the MRC strategy outperformed other relational learning approaches, in terms of accuracy obtained and running time needed. Our further analysis of these results implies the following reasons for the improvement of the MRC algorithm over other compared methods. When compared to logic-based methods, the MRC strategy first takes advantage of efficient, accurate, and state-of-the-art propositional learning algorithms and techniques. Also, the application of aggregation provides more useful attributes for the MRC strategy. On the other hand, when comparing with flattening-based approaches, the MRC algorithm avoids putting all features together. In a data set with a large number of attributes, the learning method may ignore some second-best feature sets or get confused by the large number of features available. Another reason is that, the MRC algorithm does not create additional NULL values, which may also cause problems for propositionalization methods, as being observed in our experiments.

Another important implication of the three sets experimental results is that converting multiple relations into a single flat table may be a counterproductive process. Our analysis of the results suggests that such counterproductivity may be caused by the following fact. That is, each relation in a database often groups a set

of features with very close semantic meaning. On the one hand, the propositionalization process may introduce too much complexity when connecting different relations into the resultant flat file. These include a large number of NULL values, statistical skew, and large amounts of redundant features. On the other hand, when separately learning from each relation of a database, the MRC strategy constructs a very diverse set of view learners. Each is an expert from its own view. The diversity may not only benefit the predictive performance and stability of the final model, but enables the MRC approach to make good use of "second-class" feature groups. In other words, the multiple view techniques applied here benefits the performance of the MRC algorithm.

Next, we conclude this chapter.

3.5 Conclusions

Most of today's structured data are stored in relational databases. To deal with such relational repositories, existing approaches either "upgrade" single-table learning strategies or "flatten" interconnected tables into a single relation. This chapter introduces a new multiple view approach—where neither "upgrading" nor "flattening" is required—to bridge the gap between propositional learning algorithms and relational databases. Our approach learns from multiple views of a relational databases, and information acquired by view learners is then integrated to construct a final model. Our empirical studies show that the method outperforms the classifiers induced by a number of other multirelational systems in terms of accuracy obtained and running time needed.

The next chapter presents a multiple view approach for dealing with imbalanced multirelational data.

CHAPTER 4

Learning from Imbalanced Multirelational Data

Chapter 4

Learning from Imbalanced Multirelational Data

For the things we have to learn before we can do them, we learn by doing them.

–Aristotle (384-322 BCE)

The previous chapter introduced the MRC multi-view method for multirelational classification tasks. In this chapter, we present the MRC-IM algorithm, which deals with skewed-class multirelational data. Here, the number of examples of one class in the target relation (the so-called majority class) is much higher than the other (minority classes) and correctly classifying the underrepresented examples is of importance. The importance of this research lies in the fact that, when learning from such imbalanced data, data mining algorithms often concentrate on correctly classifying the overwhelming examples from the majority class, thus performing poorly, in terms of predictive accuracy, on the minority classes.

Similar to the MRC method, the MRC-IM algorithm first learns from multiple feature sets of a relational database, resulting in a set of view learners where each has different awareness of the imbalanced problem. Next, in contrast to the

MRC strategy introduced in Chapter 3, where views are validated based on their accuracy, the MRC-IM method evaluates each view's knowledge against *both* the majority and minority classes. Finally, while a meta-learning approach is employed by the MRC algorithm, the MRC-IM method adopts the voting combination technique to integrate knowledge acquired by the view learners. By doing so, the MRC-IM algorithm is able to find a model which can better predict examples, not only from the majority class, but also from the minority classes. Our strategy performs well when measured in terms of ROC curve and AUC value achieved, especially when the class imbalance is very high.

We start this chapter with an introduction of mining imbalanced multirelational data. Next, a detailed discussion of the MRC-IM algorithm is presented. This is followed by our experimental studies against the MRC-IM algorithm. Finally, we conclude the chapter.

4.1 Introduction

There have been extensive studies [32, 35, 65, 98, 113, 141, 151, 152, 153] of imbalanced class learning applied to single table (flat file) data sets. For two-class problems, such data sets consist of a large number of examples from one class (majority class) while only a few from another class (minority class). Data mining algorithms may be biased towards the majority class (negative examples), thus producing poor predictive accuracy over the minority class (positive examples). The result is that these biased classifiers have high predictive accuracy over the majority class, but poor predictive accuracy over the minority class [113, 152, 156]. This is due to the fact that there are much more examples of the majority class than of the minority class. In these cases, learning from the naturally occurring class distribution often produces classifiers that perform poorly on minority-class examples.

In contrast to the extensive studies in the conventional machine learning community, the imbalanced class problem has seldom been investigated in the mul-

tirelational data mining research field. Recall from Chapter 1 that multirelational data mining aims to discover useful patterns across multiple interlinked relations in a relational database [71]. In many practical database applications, such as credit card fraud detection and disease diagnosis, the target tuples are highly imbalanced. That is, the number of examples of one class in the target relation is much higher than the others. Also, correctly classifying the minority examples is often of importance. Since multiple relations and indeterminate relationships between relations are involved, learning skewed classes in multirelational data becomes more complicated than when learning from flat file data sets. In addition, it is difficult to apply traditional imbalanced learning techniques to multirelational data, because of the involvement of multiple interconnected relations.

In this chapter, we present a strategy to deal with imbalanced multirelational data. The algorithm first learns from multiple views (feature sets) of a relational database, resulting in a set of view learners where each has different awareness of the imbalanced problem. Next, it combines the knowledge acquired by the view learners in order to find a model which can better predict both the majority and minority classes. Similar to the MRC algorithm, the MRC-IM method learns from multiple feature sets of the database. However, in order to cope with imbalanced class frequencies in the target relation, the MRC-IM strategy adopts different learning techniques for its *view validation* and *view combination* components, as follows. Recall from Chapter 3 that, the MRC framework deploys predictive accuracy as the criterion for its view validation. Also, a meta learning strategy is used to combine multiple view learners. In contrast, the MRC-IM approach measures the AUC values achieved by view learners in order to evaluate whether they have sufficient knowledge on the imbalanced problem settings. Furthermore, the MRC-IM method uses a majority voting strategy to integrate the multiple view learners.

Our approach was motivated by the following observations. First, using different feature subsets of a relational domain, one can potentially build a set of diverse classifiers, in terms of their knowledge of the skewed class. Recall from Chapter 1 that multirelational data are often defined by a large number of attributes dis-

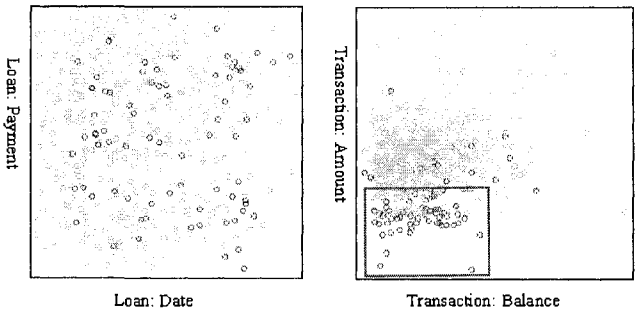


Figure 4.1: Different dimensional projections of feature space from various relations present various awareness of the imbalanced classes

tributed in multiple interconnected relations, where each groups a set of features with closely related semantic meaning. Among the large feature space in a relational domain, we have observed that different feature subsets may have different awareness of the skewed classes. That is, classifiers built using different feature sets contained in various relations of a relational database may have different capability of classifying the minority class of the presented data.

As an example, Figure 4.1 visualizes the data from the *Loan* and *Transaction* relations of the banking database as introduced in Figure 1.3. In this example, table *Loan* describes attributes about each loan application entity; instead, the *Transaction* table contains transaction entities related to certain accounts in the *Loan* table. The target tuples of this learning task consist of 324 negative examples and only 76 positive instances. Figure 4.1 shows the distribution of the data, when considered in terms of the two different tables. Here, the minority class is depicted using red circles, and the majority class is shown in green. When one considers the class distribution from the perspective of the *Loan* table, using the *date* and *payment* attributes (left hand side of the figure), it follows that there is no clear division. However, when utilizing the *Transactions* table, with attributes *amount* and *balance*, as shown in the right hand side of the figure, most of the positive examples (red circles) were gathering at the left-bottom corner. In this case, a simple decision region indicated by the solid-line rectangle can has a good

prediction against the positive examples. This example suggests that, classifiers separately built using attributes from the *Transaction* and *Loan* relations of the relational domain may result in very diverse hypotheses, in terms of knowledge about the skewed classes.

While different subsets of features in a relational domain can potentially be used to construct diverse classification models, research has shown that in many cases, ensembles of classifiers can be effective in producing a classifier that is superior to any of the individual classifiers, especially when the combined classifiers are diverse and accurate [64, 178, 197]. For example, widely used ensemble methods such as Bagging [27], Boosting [81], and Stacking [233] have shown their power of improving the predictive performance of individual classifiers [232]. Following the same line of thought, Lazarevic and Kumar [150] improved a system for detecting outliers by bagging features. Bryll et al. [29] used attribute bagging to construct ensembles of classifiers to improve the classifier's accuracy.

The above observations indicate that ensemble-based classifiers, built using features in different relations of a relational database, may potentially improve the classification of the underrepresented examples in the skewed target relation. The research presented in this chapter follows this line of thought.

The next section introduces related work on mining imbalanced class distribution data.

4.1.1 Related Work

Imbalanced learning problems under single relation presentation setting have been studied by many researchers [32, 35, 65, 98, 113, 141, 152, 156, 192, 229]. For two-class problems, the class imbalance problem corresponds to domains for which one class is represented by a large number of examples while the other is represented by only a few [113]. Research has shown that learning from imbalanced data sets presents an important challenge to the machine learning community. This is due to the fact that traditional machine learning algorithms tend to ignore small classes

while concentrating on classifying the large ones accurately (the so-called learning bias), thus producing poor predictive accuracy over the minority class [35, 113, 156, 192, 229].

In their experiments, Weiss and Provost have shown that examples belonging to the minority class are misclassified more often than examples belonging to the majority class [229]. Also, classifiers tend to perform worse on the minority class than on the majority class. The authors reported that, in their opinion, this drawback is due to two reasons. The first one is that the class "priors" in the natural training distribution are biased strongly in favor of the majority class. Some learners, such as decision trees, explicitly factor in these class priors, which biases the learning process by causing the majority class to be predicted more often than it otherwise would have been. This results in improved performance on the majority-class test examples, but degraded performance on the minority-class test examples [37]. The examples the authors gave is that, if a decision tree simply adopts the strategy of labeling the leaf with the majority class, the performance of the classifier on the majority will improve, but at the expense of the minority class. The second reason is that a classifier is less likely to fully flesh out the margins of the "minority" concept in the concept space because there are fewer examples of that class to learn from. Thus, some minority-class test examples may be classified as belonging to the majority class [229]. In the work of Japkowicz et al., the authors further investigated whether or not the class imbalances are truly responsible for the degradation of the standard classifiers when learning from imbalanced data sets [114]. The authors suggested that the significant losses of performance in standard classifiers are not directly caused by class imbalances, but caused by the small disjuncts which may be yielded by the class imbalances [114].

There have been several proposals for coping with imbalanced data sets with single relation presentation [114]. Typical approaches to dealing with imbalanced data sets include under-sampling negative examples [141], over-sampling positive instances [152], penalizing misclassification of minority classes (cost-sensitive learning) [65, 151, 194, 215], weighting examples in an effort to bias the learning toward

the minority class [32], applying boosting algorithms [98], and creating synthetic data to balance the examples of the classes [35]. Unfortunately, these imbalanced learning techniques are difficult to apply to multirelational data, because of the involvement of multiple interconnected relations in relational domains.

Few attention has been paid to the learning of skewed class relational data. To our best knowledge, one such approach was proposed by Sen and Getoor to address the cost-sensitive learning problem [213]. They introduce the so-called "structured" cost functions to deal with this problem. That is, misclassification costs assignment depends on not only the cost of individual example, but also the correction of related instances. The advantage of this method is its capability of dealing with cost-sensitive classification tasks with non-IID (independent and identically distributed) data. The motivation example the authors gave is that, in a banking system for classifying loan applications the cost of a set of account holders which associated with a particular joint account should be considered together. That is the cost of each of these holders is not independent, but related. In other words, the method can calculate the structured (or relational) cost among related instances of the data, through the use of a cost matrix. A cost matrix is formulated as $Cost(y_c, \tilde{y}_c)$. It encodes the cost of a set of related entities c (with label set y_c) of being labeled with label set $\tilde{y}_c$. However, a drawback of the above approach is that it requires relational classifiers be able to capture the correlations of the examples and to output conditional probabilities associated with the target variables. In their paper [213], they encode the "structured" cost functions into such a classifier, i.e. Conditional Markov Networks. The goal of their method is thus to minimize the expected cost of misclassification of the classifiers.

This chapter presents a strategy to learn from imbalanced target tuples in relational databases. In contrast to the method proposed by Sen and Getoor, our strategy enables traditional single-table classifiers to learn from skewed class data. That is to say many accurate, efficient traditional learning methods such as decision trees and support vector machines can be applied. In addition, the goal of our algorithm is to produce a classifier which can better predict both the minority

and majority classes in a relational database.

Table 4.1: Confusion Matrix

	Predicted Negative	Predicted Positive
Actual Negative	True Negative(TN)	False Positive(FP)
Actual Positive	False Negative(FN)	True Positive(TP)

4.1.2 Measurement Metrics

Imbalanced classification algorithms are often evaluated using the ROC curves and AUC values [35, 98, 141, 150, 156, 193, 230]. This is due to the ROC curve's insensitivity to changes in the class distribution of the data set [76]. The ROC and AUC metrics are based on the definition of the confusion matrix, as shown in Table 4.1. Two important statistics are the True Positive rate (TP rate) and False Positive Rate (FP rate). These two measures are calculated as follows

$$TP\ rate = \frac{TP}{TP + FP}$$

$$TP\ rate = \frac{FP}{FP + TN}$$

A ROC curve, as shown in Figure 4.2, represents the trade-offs between the TP rate and FP rate over a range of decision thresholds. Although it can summarize a classifier's performance over a range, the ROC curve is a two-dimensional technique. The area under the ROC curve (AUC), therefore, is often calculated to reduce the measurements of an ROC curve to a single value. This value provides an average expected performance of a classifier.

The next section presents the MRC-IM method in detail.

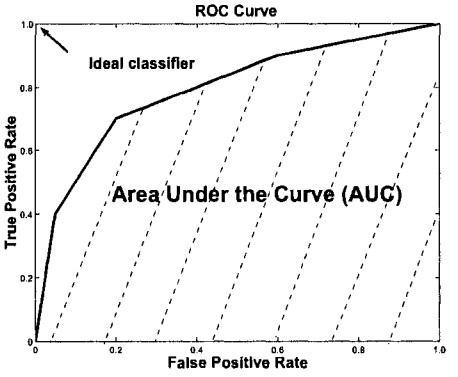


Figure 4.2: ROC curve and AUC

4.2 The MRC-IM Approach

Our method learns from multiple views (feature sets) of a relational database, in order to construct view learners which possess different awareness of the skewed classes of the target relation. By doing so, the combination of such a variety of view learners can form a model which can better predict both the majority and minority classes in imbalanced multirelational data.

The so-called MRC-IM strategy (as described in Algorithm 3) extends the MRC framework as presented in Chapter 3. Recall that this framework aims to enable propositional algorithms to explore multiple views of a relational domain directly. It consists of five sequential stages: *information propagation, aggregation, view learners construction, view validation,* and *view combination.* We here extend the MRC framework in order to handle skewed classes in multirelational data. Two major extensions, namely view validation and view combination, have been introduced in the MRC-IM strategy and are discussed next.

4.2.1 View Validation

Recall from Section 3.3.1 of Chapter 3 that, in order to produce a combined classifier which is superior to the individual view learners, we need to ensure that each inducted view learner has sufficient knowledge on the target concepts to be

Algorithm 3 The MRC-IM Algorithm

Input: A DB= $\{T_{target}, T_1, T_2, \cdots, T_n\}$,

Target tuples $\{\langle x_i, y_i \rangle\} \in T_{target}$, where $x_i \in X$,

$y_i \in Y$ (X is some instance space and Y is a label set)

View learner $\mathcal{L}$.

Output: Classification model F.

1: Propagate information from T_{target} to each $\{T_t\}_1^n$, forming $\{T'_t\}_1^n$;

2: Aggregate multiple tuples in $\{T'_t\}_1^n$, along with relation T_{target}, forming view set $\{\mathcal{V}^t\}_0^n$ from each relation in the DB;

3: Train each $\mathcal{L}$ separately with $\mathcal{V}^t \in \{\mathcal{V}^t\}_0^n$, forming hypothesis set $\{f^t\}_0^n$, where $f^t : X \rightarrow Y$, with AUC value of α^t;

4: Remove f^t with $\alpha^t \leq 0.5$, forming hypothesis set $\{f^t\}_0^{n'}$;

5: Output the final hypothesis:

$$F(X) = \underset{c}{\operatorname{argmax}} \sum_{t=0}^{n'} \hat{p_{ct}}$$

where $\hat{p_{ct}}$ is the probability estimate from the t^{th} hypothesis f^t for the c^{th} class.

learned. When considering the MRC-IM approach, we need to validate that each view learner has sufficient knowledge on *both* the majority and minority classes, in order to obtain a better combined classifier. The following is an illustrative example.

As an example, Figure 4.3 describes four view learners constructed from the example database as presented in Figure 1.3, using C4.5 decision trees as single-table learning methods. They are view learners constructed from the views *Order, Demographic, Account*, and *Loan*, respectively. This database has 606 negative examples with *class A* and 76 positive instances in *class B*. As shown in Figure 4.3, for each view learner, we provide its accuracy, AUC value, and confusion matrix.

ORDER view

Correctly Classified Instances 597 87.5367 %
Incorrectly Classified Instances 85 12.4633 %

=== Detailed Accuracy By Class ===

TP Rate	FP Rate	F-Measure	ROC Area	Class
0.982	0.974	0.933	0.604	A
0.026	0.018	0.045	0.604	B

=== Confusion Matrix ===

a	b	<-- classified as
595	11	a = A
74	2	b = B

DEMOGRAPHIC view

Correctly Classified Instances 606 88.8563 %
Incorrectly Classified Instances 76 11.1437 %

=== Detailed Accuracy By Class ===

TP Rate	FP Rate	F-Measure	ROC Area	Class
1	1	0.941	0.494	A
0	0	0	0.494	B

=== Confusion Matrix ===

a	b	<-- classified as
606	0	a = A
76	0	b = B

ACCOUNT view

Correctly Classified Instances 606 88.8563 %
Incorrectly Classified Instances 76 11.1437 %

=== Detailed Accuracy By Class ===

TP Rate	FP Rate	F-Measure	ROC Area	Class
1	1	0.941	0.494	A
0	0	0	0.494	B

=== Confusion Matrix ===

a	b	<-- classified as
606	0	a = A
76	0	b = B

LOAN view

Correctly Classified Instances 605 88.7097 %
Incorrectly Classified Instances 77 11.2903 %

=== Detailed Accuracy By Class ===

TP Rate	FP Rate	F-Measure	ROC Area	Class
0.99	0.934	0.94	0.576	A
0.066	0.01	0.115	0.576	B

=== Confusion Matrix ===

a	b	<-- classified as
600	6	a = A
71	5	b = B

Figure 4.3: Detailed performance information of the four view learners

From this figure, one can see that if one ranks all these four view learners based on their accuracy, the *Account* and *Demographic* view learners (with the same accuracy of 88.8563%) are the best two of these four. However, the confusion matrixes of these two view learners indicate that they misclassified all the 76 positive instances. As shown in Figure 4.3, the AUC values of these two view learners are 0.494, which is smaller than that of a random guessing classifier. On the other hand, when considering ranking the four view learners based on their AUC values, the best two view learners are the two built from the *Order* and *Loan* views. These two view learners obtained the AUC values of 0.604 and 0.576, respectively. The information in their confusion matrixes shows that the two view learners correctly classified some positive examples. In other words, the AUC value obtained by a view learner contains the view learner's knowledge on both the majority and minority classes. This example indicates that using AUC value as evaluation criterion for an imbalanced class problem makes more sense than using the accuracy.

Unlike the view validation criterion error rate used by the MRC approach, the MRC-IM strategy uses the AUC value to evaluate the *quality* of the multiple view learners. It removes view learners with a low AUC value. The reason for this choice is as follows. In imbalanced class applications, predictive accuracy is inappropriate for evaluating learning methods. As we have observed from the above example, a classifier which always predicts the majority class in a highly skewed class problem can get a very high resultant accuracy. In addition, accuracy metric is sensitive to the change of the class distribution in the data. AUC values, on the other hand, can provide an average performance of a classifier over a range of trade-offs between TP rate and FP rate, as well as over different decision thresholds [76, 156, 193]. We, therefore, employ the AUC value to evaluate the performance of view learners of the MRC-IM algorithm.

In the MRC-IM strategy, view learners with AUC value less than 0.5 are discarded. In the ROC space, random guessing algorithms produce the diagonal line between the points (0,0) and (1,1), thus obtaining an AUC value of 0.5. That is, following the line of thought discussed in Section 3.3.1, only view learners with average performance better than random guessing are used in the MRC-IM method. As discussed in Section 3.3.1, view learners which perform worse than random guessing should be considered failing to learn the target concept from its own view. As an illustrative case, let's revisit the above example. Although the *Demographic* and *Account* view learners yielded the highest overall accuracy, they obtained a AUC value which is lower than 0.5. As shown in Figure 4.3, these two classifiers just simply categorized all examples as negative instances. Thus, when using AUC value as the view validation criterion, the *Demographic* and *Account* view learners are not qualified for membership of the view combination component, since they don't have sufficient knowledge on the minority class. When considering the *Order* and *Loan* view learners, each of them achieved a AUC value larger than 0.5, so these two view learners will be used to form the final classification model. The combination process will be discussed next.

4.2.2 View Combination

In contrast to the meta-learning used in the MRC method, the voting combination technique is employed in the MRC-IM strategy in order to obtain better knowledge on both the majority and minority classes.

Recall from Section 3.3.2 of Chapter 3 that the MRC method applies meta-learning to combine multiple view learners. That is, a meta-learner is called upon to produce a function to control how the multiple view learners work together, in order to achieve maximum classification accuracy. Recall that, in the meta learning process, view features are first created by view learners, and a single-table learning method such as a C4.5 decision tree or a Naive Bayes classifier is then applied to construct a function from these view features. Here, the task of the single-table learning algorithm is to learn a function to maximize the predictive performance in terms of overall accuracy obtained [33]. However, in an imbalanced data set, the meta learning process can end up with a function which is able to correctly predict all examples from the majority class, but misclassify all negative instances. This is due to the fact that most of these learning algorithms use the error rate as their objective function [45]. As an illustrative case, consider the example presented in Figure 3.9. As discussed in Section 3.3.2, a meta learner which heavily relies on the view learner constructed from *region2* can form a final model with very high overall accuracy. However, suppose here the *region2* is composed of only negative examples. In such scenario, the final classification model constructed by the meta learner will be strongly biased towards the negative instances. In other words, the combination process of the MRC method aims to yield a better accuracy, regardless of the predictions on the minority class. Thus, such a combination technique tends to end up with very poor predictive performance on the underrepresented positive examples in a highly imbalanced data set. This observation indicates that a meta-learning combination technique is inappropriate for learning the minority class in a skewed class data set.

In contrast to the meta-learning method, another widely used model combina-

tion technique is the *voting* principle (as introduced in Chapter 3), which is used in Boosting [81] and Bagging [27] approaches. Boosting applies a *weighted voting* strategy. In this method, each individual model has a different weight affecting the final results. Often, the more confident individual learners have more impact on the final results [112]. In a *majority voting* combination method, such as the one used in *Bagging*, each individual model outputs one score, e.g. a probability. This score is either assigned to one class label as a whole or divided into several class labels. Next, the class label with the largest score, for example, is taken as the final result of the combined model.

In the Bagging and Boosting strategies, individual classifiers are built from overlapped instance subspaces [27, 234]. Following the same line of thought, the MRC-IM method encourages view learners with overlapped information, in terms of the target tuple space. Recall from Section 3.3.2 that the voting strategy is not suitable for the MRC algorithm, because the MRC method is prone to result in a *small* number of views with disjoint portions of the feature space and the target tuple space. Recall that, the MRC algorithm applies a correlation based view validation component, i.e. the CVV approach, to select a set of diverse views, in order to improve the efficiency and accuracy of the meta learning component. This CVV process tends to result in a small number of view learners where each has knowledge only on a disjoint region of the target space (as presented in the Figure 4.4 (b)). Such a resultant situation is good for a meta learning strategy, since meta learners often prefer diverse features with less redundancy and noise. However, such a case may proof to be disaster for a voting schema. Recall from Section 3.3.2 that, in this scenario, each voter has almost disjoint knowledge on the target tuples to be voted. These voters, therefore, do not have comparable knowledge. Thus, a voting strategy is inappropriate [232].

In order to compensate the above negative effects of the voting schema, the MRC-IM algorithm does not deploy the CVV strategy. Another reason for not applying the CVV method is that the objective of the CVV approach is to identify a set of diverse views, regardless their knowledge on the minority class. As a

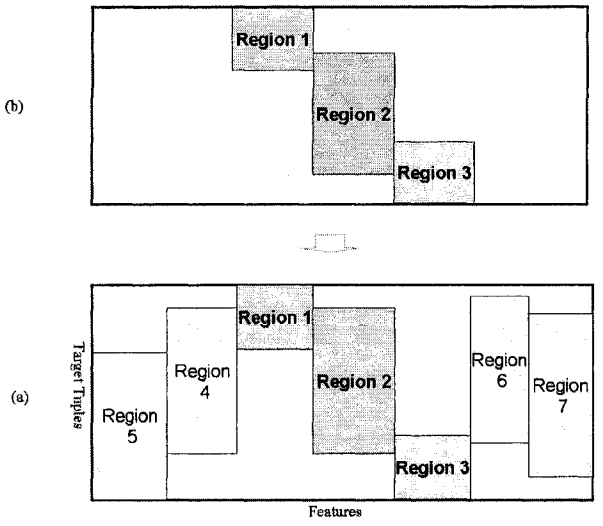


Figure 4.4: Learning regions in the target relation

consequence of such an aim, the CVV approach may return a set of view learners which have little knowledge on the underrepresented target instances. The MRC-IM strategy excludes the process of obtaining a small set of disjoint views, as used by the CVV technique. Thus, overlapped views exist in the view combination process of the MRC-IM algorithm. As an example, Figure 4.4 (a) pictures a set of views with overlapped target tuples. As shown in the figure, views in the MRC-IM algorithm overlap within the tuple space of the provided relational domain. In this sense, employing a voting strategy to combine the views, therefore, is inclined to result in a more reliable combined model, compared to the scenario presented in Figure 4.4 (b).

In the MRC-IM algorithm, we employ a majority voting method rather than a weighted voting strategy. This is due to the following observation. View learners in the MRC-IM strategy learn only from their own feature set. In addition, each view learner may be trained by different subset of the target tuples. We, therefore, have no reason to strongly believe that certain resultant learners are "better" than the others. Furthermore, unlike the weighted voting strategy, a majority voting scheme can prevent over-confidence in voters.

The majority voting strategy in the MRC-IM algorithm, as described in Algorithm 3, works as follows. For each test example x, each view learner f^t first outputs a probability score for assigning each class c of the class set Y, i.e. $\hat{p_{ct}}$. Subsequently, these scores are summed up according to each class in the learning task. Finally, the class with the largest probability score is assigned to the test instance as the final result. That is, the final classifier decision of a test example x is based on the sums of the probability outputs [112]

$$F(X) = \underset{c}{\operatorname{argmax}} \sum_{t=0}^{n'} \hat{p_{ct}}$$

Here, the *argmax* stands for the argument of the maximum. In other words, each value of X classify to the class that receives the largest score.

Prediction from View learner1		Prediction from View learner2		Prediction from View learners3		Sum of the probability output of each class		Predicted class
class1	class2	class1	class2	class1	class2	Class1	class2	
0.93	0.07	0.21	0.79	0.73	0.27	1.87	1.13	class1
0.66	0.34	0.32	0.68	0.46	0.54	1.44	1.56	class2

Figure 4.5: A majority voting example

As an example, Figure 4.5 illustrates how the class labels of two test instances are decided by three view learners, using a majority voting strategy. As shown in Figure 4.5, for each test instance, each classifier first outputs its probabilistic prediction for each of the potential classes, namely *class1* and *class2*, respectively. Next, for each class, the probabilistic predictions from all the three classifiers are summed up. Finally, the test instance is classified to the class with the largest sum score. That is, the first instance is given a class label of *class1*, since the sum probabilities (for this test example) for the *class1* and *class2* are *1.87* and *1.13*, respectively. Similarly, the second test example is classified to *class2* because its sum probabilities for the two potential classes are *1.44* and *1.56*, respectively.

The above two sections present our extensions to the MRC framework. Now

we simply describe the entire process of the MRC-IM strategy (Algorithm 3). As shown in Algorithm 3, the method, first, constructs multiple views from the relational database through information propagation and aggregation. Next, it trains multiple learners, each learning from a different view constructed. Subsequently, view learners with AUC value less than 0.5 are removed. Finally, the trained view learners are combined using majority voting.

The next section presents our empirical studies.

4.3 Experimental Study

This section provides the results obtained for the MRC-IM algorithm on benchmarking databases. These results are presented in comparison, in terms of AUC and ROC achieved, with three other state-of-the-art multirelational data mining systems, namely TILDE, RelAggs, and CrossMine, along with the "baseline" "flattening" approach SimFlat and the original MRC method (as presented in Chapter 3).

4.3.1 Methodology

We derived six learning tasks from the databases used in Chapter 3. These learning tasks present varying degree of class distribution in the target relation. That is, the percentage of the positive examples in these six learning problems ranges from 4% to 34%. Thus, these learning tasks provide us a diverse test bed.

We implemented the MRC-IM algorithm using Weka [232]. Our experiments applied the same single-table learners as used in Chapter 3, namely the C4.5 decision trees [196] and Naive Bayes probabilistic classifiers [115] for the learning in the MRC-IM, RelAggs, MRC, and SimFlat approaches. The default settings of these two learning methods were used. Each of these experiments produces ROC curves and AUC results using ten-fold cross validation. In order to generate the average ROC curve of the 10 folds, all test instances of the 10 folds were put together and

sorted according to their probabilities of belonging to the classes; the AUC values were then calculated using the Wilcoxon-Mann-Whitney statistic [232].

4.3.2 Databases Used

Table 4.2: Summary of the data sets used

Data Set	#target tuples	#related relations	target class distribution	percentage of positive examples	# tuples in task
MUT188	188	3	125:63	34%	15,218
F400AC	400	7	324:76	19%	75,982
F234AC	234	7	203:31	13%	75,816
F682AC	682	7	606:76	11%	76,264
ECML_I	4,065	8	3705: 360	8%	194,214
ECML_II	7,329	8	6969: 360	4%	197,478

Our first experiment (denoted as MUT188) was conducted against the Mutagenesis database (as presented in Chapter 3). In this data set, 34% of the target tuples belong to the minority class. Our second experiment was conducted against the Financial database. Recall from Section 3.4.2 that this database provides us with three different learning problems. There are 11%, 13%, and 19% of minority class instances in the target tables of the three learning tasks, respectively. Our last experiment used the database for the ECML 1998 Sisyphus Workshop. The original database consists of balanced target class distributions. In order to obtain highly imbalanced data sets, we artificially derived two learning tasks from this database. In the first task (denoted as ECML_I), we removed tuples in class 2 (minority class) with id smaller than 14700; in the second task (denoted as ECML_II), we reversed the class of these removed tuples and then added them back to the target relation in order to obtain a highly imbalanced data set. There are 8% and 4% of minority class instances in the target tables of the two learning tasks, respectively.

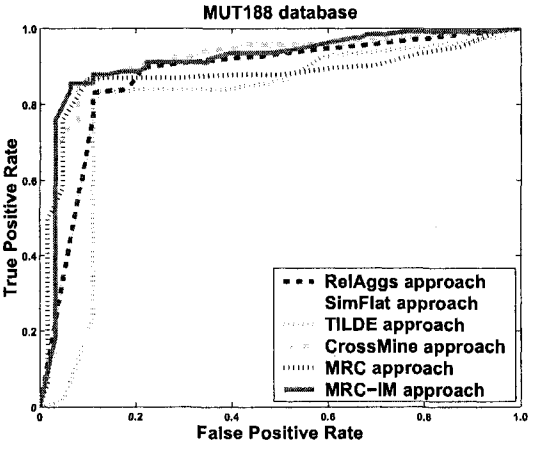


Figure 4.6: ROC curves for the Mutagenesis Database (with C4.5 algorithms applied in the MRC-IM, RelAggs, CrossMine, MRC, and SimFlat methods)

A summary of the learning data sets used is given in Table 4.2, where class distribution, percentage of positive examples, and the number of background relations and target tuples are provided.

4.3.3 Experimental Results

In order to evaluate the performance of the MRC-IM method and to determine how different applied conventional learning algorithms impact the performance of the tested algorithms, we conducted two experiments, with C4.5 and Naive Bayes as learning methods, respectively.

Using C4.5 decision trees

In the first experiment, C4.5 decision trees were used by the MRC-IM, RelAggs, and SimFlat approaches. Also, C4.5 decision trees were used as the view learners and meta learners of the MRC method. We present the ROCs and AUCs obtained for each of the six learning tasks in Figures 4.6 to 4.11 and Table 4.3, respectively. In Table 4.3, the best results for each data set were highlighted in *bold*.

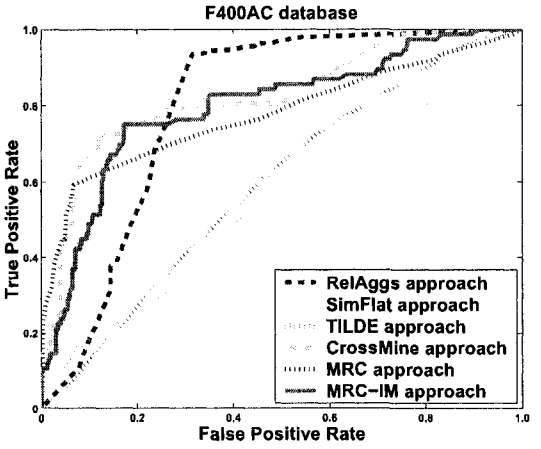


Figure 4.7: ROC curves for the F400AC Database (with C4.5 algorithms applied in the MRC-IM, RelAggs, CrossMine, MRC, and SimFlat methods)

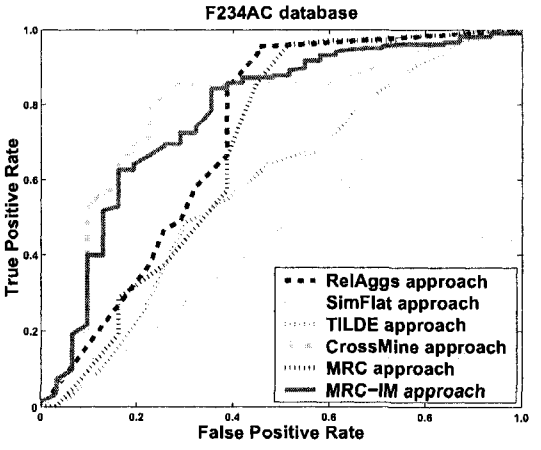


Figure 4.8: ROC curves for the F234AC Database (with C4.5 algorithms applied in the MRC-IM, RelAggs, CrossMine, MRC, and SimFlat methods)

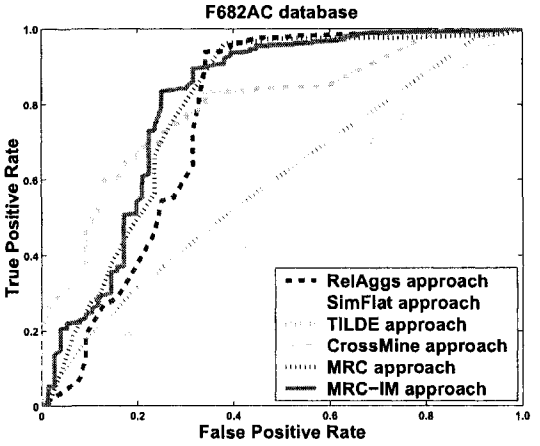


Figure 4.9: ROC curves for the F682AC Database (with C4.5 algorithms applied in the MRC-IM, RelAggs, CrossMine, MRC, and SimFlat methods)

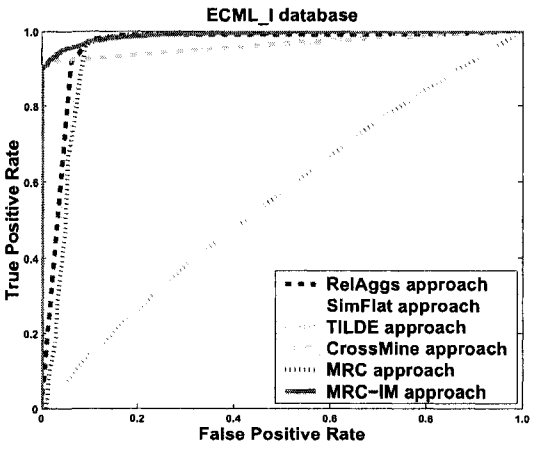


Figure 4.10: ROC curves for the ECML_I Database (with C4.5 algorithms applied in the MRC-IM, RelAggs, CrossMine, MRC, and SimFlat methods)

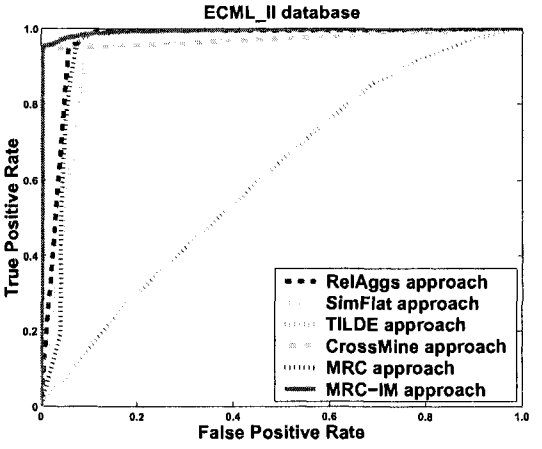


Figure 4.11: ROC curves for the ECML_II Database (with C4.5 algorithms applied in the MRC-IM, RelAggs, CrossMine, MRC, and SimFlat methods)

AUC analysis: The AUC values listed in Table 4.3 show that for almost all datasets the MRC-IM method was able to improve over the RelAggs, TILDE, CrossMine, and SimFlat algorithms, when C4.5 decision trees were used. Exceptions were against the F400AC and F234AC data sets when compared with the CrossMine algorithm. In other words, our MRC-IM strategy can provide an improvement in classifying underrepresented classes of the data against almost all data sets, regardless of the skewed rate of the datasets. Promisingly, the MRC-IM strategy conducted almost perfect AUC values against the two most highly imbalanced data sets, namely ECML_I and ECML_II, with AUC values of 0.990 and 0.993, respectively. These values outperformed that of all other five tested systems, namely TILDE, RelAggs, CrossMine, MRC, and SimFlat approaches.

Furthermore, the experimental results as presented in Table 4.3 show that the MRC-IM algorithms significantly benefited from the framework of learning from multiple views. Recall that, the flat file conducted by the SimFlat approach consists of attributes from all views of the MRC-IM strategy. That is, in the MRC-IM algorithm, a set of feature sets (each feature set corresponds to a view in the MRC-IM strategy) were separately employed to learn the skewed classes. On the

Table 4.3: AUCs of the RelAggs, SimFlat, TILDE, CrossMine, MRC, and MRC-IM approaches (using C4.5)

Data set	RelAggs	SimFlat	TILDE	CrossMine	MRC	MRC-IM
MUT188	0.876	0.888	0.807	0.912	0.874	**0.914**
F400AC	0.797	0.582	0.591	**0.824**	0.777	0.798
F234AC	0.722	0.48	0.591	**0.785**	0.687	0.775
F682AC	0.762	0.506	0.590	0.793	0.792	**0.805**
ECML_I	0.957	0.945	0.553	0.961	0.946	**0.990**
ECML_II	0.965	0.949	0.608	0.971	0.952	**0.993**

other hand, the SimFlat approach combined all sets of feature set into a flat file, and then learned from all these features available. When comparing the AUC values resulted from the SimFlat and MRC-IM algorithms, the results presented in Table 4.3 show that the MRC-IM approach meaningfully outperformed the SimFlat method against almost all six data sets. For example, against the F682, F234AC, F400AC, ECML_I, and ECML_II data sets, the MRC-IM algorithm improved the AUC values over the SimFlat method by 29.9%, 29.5%, 21.6%, 4.5%, and 4.4%, respectively.

In addition, further study of the experimental results show that the MRC-IM algorithm benefited from the use of the majority voting strategy, when compared with the MRC method, where a meta learning approach is employed for view combination. The results in Table 4.3 show that the MRC-IM approach outperformed the MRC method against all tested cases, in terms of AUC obtained.

Our results also suggest that the CrossMine method generally performed better than the RelAggs, SimFlat, and TILDE approaches, and yielded good results when the data sets were not much imbalanced, when compared with the MRC-IM strategy. The possible reason is that, the CrossMine algorithm employed a "sampling" technique (as will be discussed next) to ensure a balanced distribution in the target tuples when each rule was constructed, thus, decreasing the skewed classes problem in the data. The *sampling* technique employed in the CrossMine algorithm works as

follows. Recall from Chapter 2 that the CrossMine strategy was proposed by Yin et al. [242]. Recall that learning techniques such as the virtual joins, look-one-ahead, and sampling strategies are employed in this algorithm. The sampling technique in the CrossMine method aims at speeding up the process of building a good rule. The CrossMine algorithm builds a set of logic rules to distinguish negative examples from positive ones. Before building a rule, the CrossMine method samples the examples in order to balance the classes of the tuples in the training data set, if the number of positive examples is greater than NEG_POS_RATIO times of the negative examples. The default value of NEG_POS_RATIO is 1, which was used in our experiments. That is, before each rule was built in the CrossMine method, the "undersampling" technique may be applied to ensure the balance of the negative and positive tuples in target relations. In other words, the observation here implies that the employed "sampling" technique benefited the performance of the CrossMine method.

Our further analysis of the experimental results also shows that the RelAggs algorithm performed slightly better than the SimFlat and TILDE methods. In addition, our results indicate that the TILDE algorithm yielded good results when the data is less skewed (such as against data set MUT188), but obtained unsatisfactory performance when the data sets are more imbalanced, such as against the other five data sets.

ROC analysis: The ROC graphs in Figures 4.6 to 4.11 again demonstrate that, in general, the MRC-IM method potentially performs better against both the minority and majority classes, as compared with the RelAggs, TILDE, and SimFlat approaches. The results show that, when against the F400AC, F234AC, and F682AC data sets, the ROC curves of the MRC-IM strategies dominated that of the TILDE and SimFlat approaches. In these three cases, the ROC curves of the MRC-IM method were overlapped with that of the RelAggs and CrossMine methods in the ROC space. Promisingly, against the two most imbalanced data sets ECML_I and ECML_II, the ROC curves of the MRC-IM method dominated that

Table 4.4: AUCs of the RelAggs, SimFlat, TILDE, CrossMine, MRC, and MRC-IM approaches (using Naive Bayes)

Data set	RelAggs	SimFlat	TILDE	CrossMine	MRC	MRC-IM
MUT188	0.899	0.909	0.807	0.912	0.848	**0.947**
F400AC	0.650	0.685	0.591	**0.824**	0.823	0.751
F234AC	0.630	0.591	0.591	0.785	0.685	**0.812**
F682AC	0.686	0.692	0.590	0.793	**0.845**	0.769
ECML_I	0.956	0.929	0.553	0.961	**0.980**	**0.980**
ECML_II	0.964	0.919	0.608	0.971	0.967	**0.989**

of all other tested systems. In other words, in these two highly skewed-class tasks, the MRC-IM strategy performed superior to the other five tested systems over all possible decision thresholds.

In addition, these ROC graphs confirmed that the MRC-IM algorithms significantly benefited from learning from multiple views of a relational database. For example, against almost all the tested data sets, the ROC curves conducted by the MRC-IM strategies dominated that of the SimFlat approaches. One exception is against the less skewed MUT188 database, in which the MRC-IM's ROC curve was slightly overlapped with the ROC curve yielded by the SimFlat method (at the ROC space with high TP rates).

Using Naive Bayes probabilistic learners

In the second experiment, Naive Bayes was used by the MRC-IM, RelAggs, and SimFlat approaches. We present the ROCs and AUCs obtained for each of the six learning tasks in Figures 4.12 to 4.17 and Table 4.4, respectively. In Table 4.4, the best results for each data set were also highlighted in *bold*.

AUC analysis: The AUCs listed in Table 4.4 again show that for almost all learning tasks the MRC-IM method was able to improve over all other tested systems, namely the RelAggs, TILDE, CrossMine, and SimFlat algorithms, when Naive

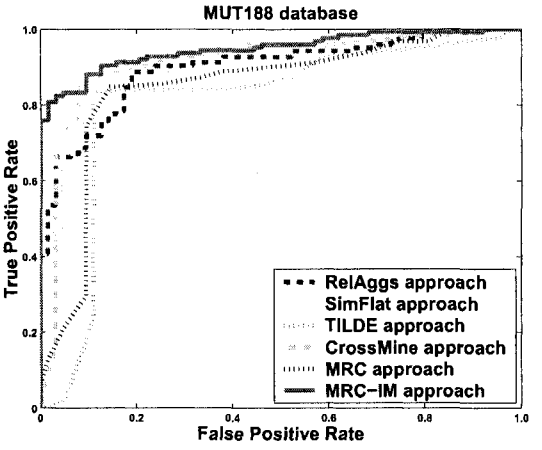


Figure 4.12: ROC curves for the Mutagenesis Database (with Naive Bayes algorithms applied in the MRC-IM, RelAggs, CrossMine, MRC, and SimFlat methods)

Bayes learners were used. Exceptions were against the F400AC and F682AC data sets when compared with the CrossMine algorithm. In other words, our MRC-IM strategy provides an improvement of classifying underrepresented classes of the data against almost all data sets, regardless of the skewed ratio of the datasets, when Naive Bayes was used as view learners as well. Again, the MRC-IM method produced the best results for very highly imbalanced data sets.

When comparing results obtained by using C4.5 as learning methods, the MRC-IM strategy yielded consistent results in terms of AUCs obtained regardless of the use of different learning methods. However, the RelAggs methods performed somewhat worse in five of the six learning tasks, as compared to that of using the C4.5 as learners. Experimental results also indicate that the MRC method performed somewhat better than when using C4.5 decision trees as view learners. In addition, the SimFlat method had AUC improvement in half of the learning problems, but yielded AUC loss against another half of the learning tasks, compared to results of using C4.5 as mining strategy. Note that the applied propositional learning algorithm did not affect the performance of the TILDE and CrossMine methods.

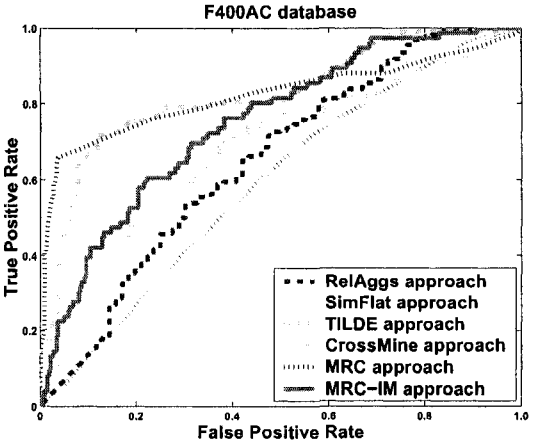


Figure 4.13: ROC curves for the F400AC Database (with Naive Bayes algorithms applied in the MRC-IM, RelAggs, CrossMine, MRC, and SimFlat methods)

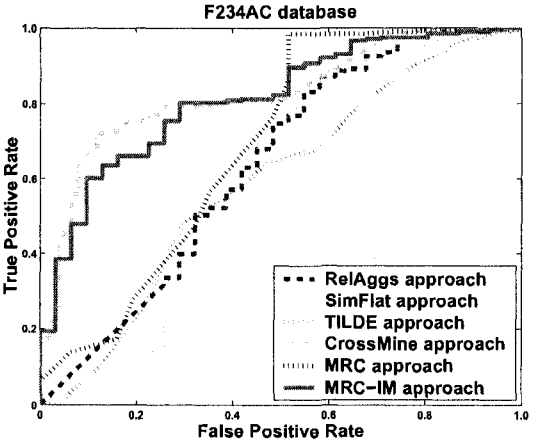


Figure 4.14: ROC curves for the F234AC Database (with Naive Bayes algorithms applied in the MRC-IM, RelAggs, CrossMine, MRC, and SimFlat methods)

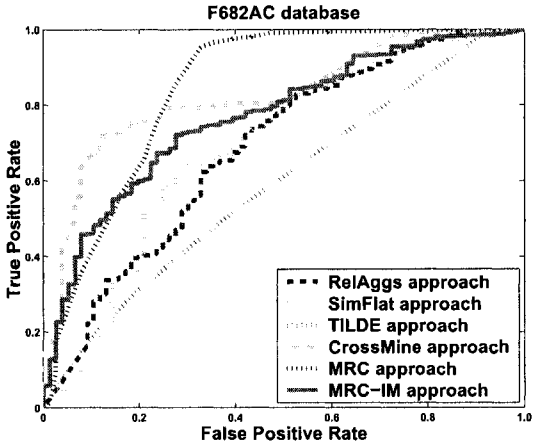


Figure 4.15: ROC curves for the F682AC Database (with Naive Bayes algorithms applied in the MRC-IM, RelAggs, CrossMine, MRC, and SimFlat methods)

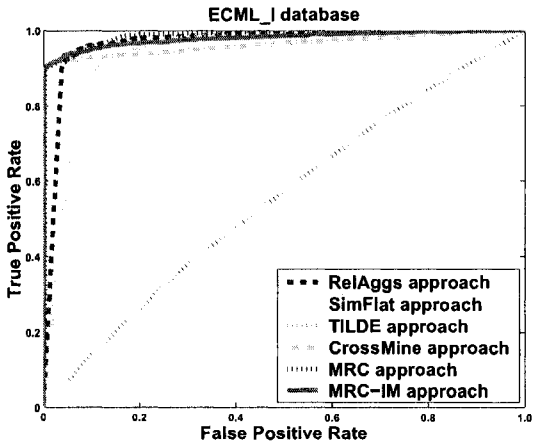


Figure 4.16: ROC curves for the ECML_I Database (with Naive Bayes algorithms applied in the MRC-IM, RelAggs, CrossMine, MRC, and SimFlat methods)

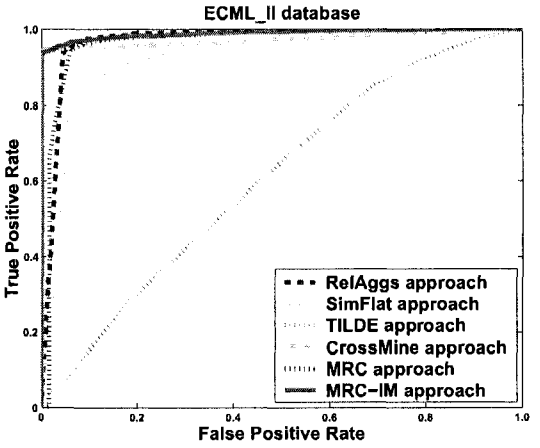


Figure 4.17: ROC curves for the ECML_II Database (with Naive Bayes algorithms applied in the MRC-IM, RelAggs, CrossMine, MRC, and SimFlat methods)

ROC analysis: Analysis on ROC graphs as shown in Figures 4.12 to 4.17 confirmed that the MRC-IM method was superior as compared to the RelAggs, TILDE, and SimFlat approaches. The ROC curves of the MRC-IM methods dominated over ROC curves produced by the other three approaches. That is, the MRC-IM algorithms were able to more correctly classify both the majority and minority classes in the data regardless of the class skew and decision thresholds. When compared with the CrossMine algorithm where the "under-sampling" technique was employed, the ROC curves produced by the MRC-IM strategy dominated that of the CrossMine method against the MUT188, ECML_I, and ECML_II data sets. Against the other three tested data sets, the curves obtained by these two methods overlapped in the ROC space. In addition, these graphs also confirmed that the MRC-IM strategy yielded consistent results in terms of ROC curves obtained regardless of the application of different single-table learning methods.

In summary, the two experimental results show that, against the six imbalanced learning problems, the MRC-IM strategy outperformed the other five tested relational learning systems both in terms of AUC values and ROC graphs obtained, against most of the tested cases. Promisingly, our method produced the best re-

sults for very highly imbalanced data sets. In addition, these results imply that the superior performance of the MRC-IM strategy significantly benefited from the learning of multiple views. Note that another advantage of the MRC-IM approach is that, in contrast to the extensive preprocessing effort of converting multiple relations into a single table in the RelAggs and SimFlat approaches, the MRC-IM algorithm directly learns from a relational database without preprocessing the data into a universal table. Also, when compared with the CrossMine and TILDE methods, the MRC-IM strategy can apply different algorithms chosen from a wide range of traditional learning methods.

4.3.4 Discussion

This chapter studied forming diverse classifiers, using different feature subsets of the learning domain, to cope with imbalanced class data in a database context. While voting and meta learning techniques for combining multiple individual classifiers work well in many domains, it would be beneficial to search new techniques for model combination for imbalanced learning applications. This is especially true when each individual learner learns only from a disjoint subspace of the entire learning domain. One possible direction for the combination is to look at the "hyper-ROC" model concept. In such scenario, a ROC curve may be constructed from a few view learners. The results of these view learners as subsequently combined by only considering their "area of expertise". Some view learners distinguish better at the positive end, others are good at the negative end, and the others are better in the middle of the ROC spectrum. When learning from an imbalanced data set, this type of combination technique will be able to dynamically select the best area of expertise of each view learner when classifying new instances.

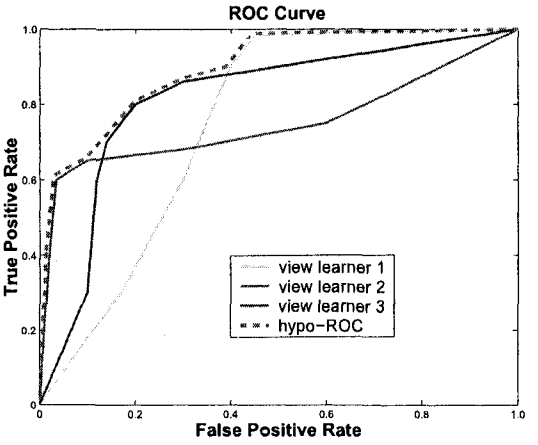


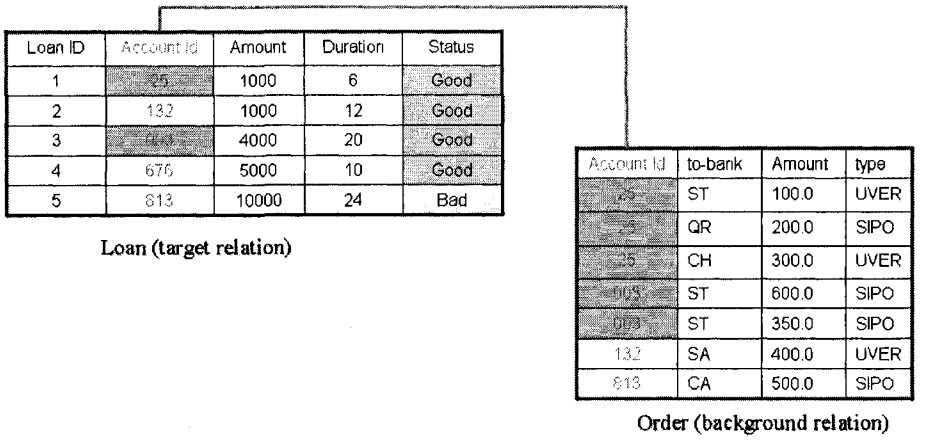
Figure 4.18: ROC which would be achieved by hypo-ROC model combination

As an example, Figure 4.18 pictures three view learners where each performs better in various areas of the ROC spectrum. That is, view learner1 and view learner2 perform better at the two ends of the ROC space, respectively, but view learner3 possesses strong "expertise" in the middle of the ROC spectrum. In this example, a hypo-ROC combination scenario would achieve a combined model with ROC curve as depicted in the red-dot line of Figure 4.18. In other words, the ROC curve yielded by the hypo-ROC strategy can optimize the AUC value by using the best area of expertise of the three individual learners when combining them.

In order to achieve the above goal, further studies on developing sophisticated view validation strategies for imbalanced class learning would also be very useful. Such validation methods should identify a set of view learners where each has *diverse*, but sufficient knowledge on *both* the majority and minority classes of the target relation.

In addition, it would be very interesting to investigate how to upgrade widely used techniques for dealing with imbalanced single-table data into the relational learning setting. For example, it would be beneficial to research how to implement a sampling approach (such as the over-sampling or under-sampling strategy) in relational settings and to study how such technique affects the learning in imbalanced relational problems. As an example, when applying down-sampling to balance the

class frequency in a relational database, it would make sense to not only balance the class distribution of tuples in the target relation, but also consider how the sampling affects the related information in interconnected background tables.



Loan ID	Account Id	Amount	Duration	Status
1	25	1000	6	Good
2	132	1000	12	Good
3	003	4000	20	Good
4	676	5000	10	Good
5	813	10000	24	Bad

Loan (target relation)

Account Id	to-bank	Amount	type
25	ST	100.0	UVER
25	QR	200.0	SIPO
25	CH	300.0	UVER
003	ST	600.0	SIPO
003	ST	350.0	SIPO
132	SA	400.0	UVER
813	CA	500.0	SIPO

Order (background relation)

Figure 4.19: Sampling in related relations

Consider the sample database as provided in Figure 4.19, where the target table *Loan* consists of four negative examples (with the *Status* value of *Good*) and one positive instance (with the *Status* value of *Bad*), along with a background relation *Order* related to the target relation through the foreign key join *Account-Id*. In this scenario, it makes sense to consider how tuples in the background relation *Order* relate to target tuples in the target relation *Loan*, when sampling the target entities. For example, suppose the sampling process removes all target tuples with the *Loan-ID* value of *25* or *003* (highlighted with *green* in the table). This sampling procedure, therefore, results in less skewed-class data, where two negative examples and one positive instance are left. However, as a consequence of such sampling process, the resultant *Order* background relation has only two tuples, namely the tuples with *Account-Id* values of *813* and *132*, related to the resultant target entities. In this way, one can conclude that such sampling process, which does not pay attention to the related relations, will lost useful information from related relations. For example, the above sampling process lost information

contained in the five tuples as highlighted with *green* in the *Order* background relation. Following the same line of thought, it would also be worthwhile to study approaches which can upgrade single-table cost-sensitive techniques to cope with cost-sensitive multirelational data.

4.4 Summary

Knowledge discovery applications of commercial databases have often been hindered by the lack of skewed-class learning algorithms. This chapter presents a strategy to deal with imbalanced multirelational data. The method learns from multiple views (feature sets) of relational data in order to construct view learners with different awareness of the imbalanced problem to be learned. These various observations possessed by multiple view learners are then combined in order to yield a model which has better knowledge on both the majority and minority classes in a relational database. Our algorithm takes advantages of large numbers of features available in multirelational data and was motivated by the observation that different feature subsets have different sensitivity to, or awareness of, the skewed class distribution of the presented data.

The algorithm presented here achieved promising results when compared with five other multirelational learning methods against six imbalanced learning problems, in terms of AUCs and ROC curves obtained. These data sets contained different skew ratios, different relational schemas, and different tuples both in the target relation and background relations, thus providing a diverse test bed. The experimental results also indicate that regardless of the skew rate in the data, our method can improve the accuracy of classifying positive examples. Furthermore, an important result indicates that the MRC-IM method is superior when the class imbalanced is very high.

The next chapter will present a multi-view approach which enables view construction process to search and collect relevant features from multiple related relations.

CHAPTER 5

Constructing Views across Relations

Chapter 5

Constructing Views across Relations

The previous two chapters introduced the MRC framework for multirelational classification tasks and an extended algorithm for dealing with imbalanced data. While the MRC framework allows constructing each individual view only from a sole relation, most of today's relational learning systems enable the search for useful attributes across multiple related tables. Will the MRC framework gain further performance improvement if its view construction component is capable of collecting good features across related relations in the database? This chapter describes another extension of the MRC framework, the so-called MRC-Cross extension, which aims to answer the above question.

This chapter starts with the motivation of this extension. Next, it introduces two preliminary concepts for the extension, namely *join path* and *relational feature*. A detailed discussion of the extended algorithm is then presented. Finally, this chapter is concluded with a presentation of the experimental studies.

5.1 Motivation

Recall from Chapter 3 that, when constructing an individual view in the MRC approach, we impose some restrictions. That is, the MRC algorithm considers features obtained from only a sole relation in the database. Such constraint raises a question. That is whether or not the MRC method will performance better if it is allowed to search for useful features across multiple tables. Most relational learning systems are capable of doing so. For example, recall from Chapter 2 that the FOIL, CrossMine, and TILDE algorithms search for good predicates from all related relations available; RelAggs and RollUp strategies collect features from all tables into the target relation. In order to answer such a question, we extend the MRC method to explore related attributes across multiple relations. In this way, through the development of the extension, we can evaluate the performance impact of an alternative view construction strategy on the multiple view learning framework. The major goal of the so-called MRC-Cross strategy is, therefore, to enable an individual view to include relevant features across interconnected tables in a relational database.

5.2 Preliminary Concept

To achieve the above goal, we introduce two preliminary concepts for the MRC-Cross algorithm: *join path* and *relational feature*. A join path describes a set of related relations in a relational database. Relational features gather information from relevant relations defined by a join path. As will be discussed in detail in Section 5.3, these two elements enable a learning method to pull essential information from multiple relevant relations while constructing classification hypotheses. Next, we will present these two concepts in detail.

Join Path

Recall from Section 1.3.1 that in a relational database $\Re$ which describes a set of tables $\Re = \{T_1, \cdots, T_n\}$, foreign key attributes of one table link to primary key attributes of other tables. This link specifies a *join* between two tables T_1 and T_2, through key attributes a and b from T_1 and T_2, respectively (denoted as $T_1 \frac{\bowtie}{a=b} T_2$). A set of joins with n tables $T_1 \bowtie \cdots \bowtie T_n$ describes a join path, denoted as $\bowtie_n^c = T_1 \times \cdots \times T_n$ (c is the join conditions). For each entity t in the first relation T_1 of the join path $\bowtie_n^c$, each other table T_i in this join path will have a *set* of tuples related to t. In this thesis, we let $\mathcal{T}_t(T_i) = \{t_1, \ldots, t_j\}$ denote the *set* of tuples in table T_i, corresponding to tuple t in table T_1. To help understanding the join path concept, consider the following example.

Loan ID	Account Id	Amount	Duration	Status
1	25	1000	6	Good
2	132	1000	12	Good
3	813	10000	24	Bad

Loan (target relation)

Account Id	to-bank	Amount	type
25	ST	100.0	UVER
25	QR	200.0	SIPO
25	CH	300.0	UVER
132	SA	400.0	UVER
813	CA	500.0	SIPO

Order (background relation)

Figure 5.1: An example join path

Example 5.2-1 (Join Path). Let's revisit examples 3.2-2 and 3.2-4 (as presented in Section 3.2). In fact, these two foreign key chains define two join paths, i.e. *Loan* $\bowtie$ *Order* and *Loan* $\bowtie$ *Disposition* $\bowtie$ *Client*, respectively, from the example database as presented Figure 1.3. The first example describes a join path with two relations, i.e. the *Loan* target table and the *Order* background table, through a reference key *account-id*. In this way, each tuple in the *Loan* table relates to a set of entities in the *Order* background table. As shown in Figure 5.1, these two tables have different sets of attributes, and they are linked through the foreign key join *Account-id*. In addition, the target entity with *loan-id 1*, for example, has three related tuples in the *Account* table. Similarly, examples 3.2-4 described a join

path with three involved tables, namely the target relation *Loan* and background relations *Disposition* and *Client*.

Relational Feature

Based on relations defined by join paths, relational features are constructed using tuple attributes and the standard relational algebra operation π (projection) and aggregation operator ϕ (e.g., average, min, max). Relational features of tuple t, denoted as $\varphi(t)$, is formally defined as follows:

If there exist a join path

$$\exists \bowtie_n^c = T_t \times T_{b_1} \times \cdots \times T_{b_n},$$

for

$$\forall t \in T_t, \forall T_{b_i}, \tau \in \{\pi, \phi\},$$

then the *relational feature* set is:

$$\varphi(t) = \{\tau(t), \tau(T_t(T_{b_i}))\} \tag{5.1}$$

In this way, relational features constructed from a join path possess information from all relations in the join path.

Example 5.2-2 (Relational Feature). Consider we have two relations T_t and T_1. Suppose $T_t \frac{\bowtie}{a=b} T_1$ and T_t has one descriptive attribute *age* and T_1 has a descriptive attribute *income*. The relational features for each tuple t in T_t are: $\{age, \tau(income)\}$. Here τ (*income*) transforms information from multiple tuples in T_1, which related to tuple t of T_t by join $\frac{\bowtie}{a=b}$, into a single entity (being grouped by the *tuple identifier* of tuple t). For a practical example, let's revisit the first join path described in the previous example, i.e. example 5.2-1, where the join path includes relations *Loan* and *Order*. Consider here we use the same aggregation functions deployed

by the RelAggs algorithm in Weka. Fourteen relational features are created. They are original attributes *loan-id* (tuple ID), *status* (class label), *amount*, and *duration* from the target table *Loan* and aggregate features *count(to-bank)*, *count(to-account)*, *count(type)*, *sum(amount)*, *avg(amount)*, *min(amount)*, *max(amount)*, *stedev(amount)*, and *count(amount)* from the *Order* table. The generation of these relational features can be obtained through the following SQL query:

```
SELECT L.loan-id, L.status, L.date, L.amount, L.duration,
       COUNT(A.to-bank), COUNT(A.to-account), SUM(A.amount),
       AVG(A.amount), MIN(A.amount), MAX(A.amount),
       STDDEV(A.amount), COUNT(A.amount), COUNT(A.type)
 FROM  Loan L, Order A
 WHERE L.account-id = A.account-id;
 GROUP BY L.loan-id
```

Relational Classification using Relational Features

Through constructing relational features, a multi-view relational classification problem refers to using these created attributes to form a set of n views $\{V^1, \ldots, V^n\}$. These views are then used to model a target function in order to approximate the target concept to be learned.

Recall from Chapter 3 that, in a multi-view learning setting with two views, i.e. V^i and V^j, a tuple t in the target relation T_t is viewed as:

$$\vec{t} = \langle \varphi^i(t), \varphi^j(t), y \rangle \tag{5.2}$$

where $\varphi^i(t)$ and $\varphi^j(t)$ are instances in the views V^i and V^j, respectively. The variable y denotes the class label.

Recall that a relational classification task is to find a function $F(x)$ which maps

each target tuple x to the category Y:

$$Y = F(x, T_{1\cdots n}, T_{target}), x \in T_{target} \tag{5.3}$$

Combining Equation 5.3 and Equation 5.2, a multi-view relational classification task becomes

$$y = F(x, \varphi^1(x), \ldots, \varphi^n(x)) \tag{5.4}$$

where

$$x \in T_{target}, \varphi^1(x) \in V^1, \ldots, \varphi^n(x) \in V^n$$

Given a relational database as input, the task of the MRC-Cross algorithm is, therefore, to obtain a function as formulated in Equation 5.4. Next, we will present this strategy in detail.

5.3 The MRC-Cross Algorithm

The MRC-Cross method extends the view construction process of the original MRC method introduced in Chapter 3. Recall from Chapter 3 that the MRC method consists of five sequential stages, namely *information propagation, aggregation, view learners construction, view validation*, and *view combination*. The view construction procedure of the MRC approach aims to construct a set of attribute-based training data sets, each based on a single table. This construction process is implemented by the first two phases of the MRC framework, i.e. *information propagation* and *aggregation* stages. In contrast to the view construction of the MRC method, where features in each view comes from the same relation of the relational database, the view construction process of the MRC-Cross algorithm searches and collects relevant features from multiple relations. Next, we will discuss this process in detail.

Algorithm 4 MRC-Cross Algorithm

Input: A DB= $\{T_{target}, T_1, T_2, \cdots, T_n\}$,

View learner $\mathcal{L}$, meta learner $\mathcal{M}$.

Output: Classification model $\mathcal{F}$.

1: Convert the DB into a relational domain graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ (Algorithm 5);
2: Construct subgraph set $\{\mathcal{G}_{s_1}, \cdots, \mathcal{G}_{s_n}\}$ from $\mathcal{G}(\mathcal{V}, \mathcal{E})$ (Algorithm 6);
3: Generate relational features for subgraph set $\{\mathcal{G}_{s_1}, \cdots, \mathcal{G}_{s_n}\}$, forming candidate view set $\{V_d^1, \cdots, V_d^n\}$;
4: Select a set $\{\mathcal{V}^i\}_1^{n'}$ from $\{V_d^1, \cdots, V_d^n\}$;
5: Train $\mathcal{L}$ with $\{\mathcal{V}^i\}_1^{n'}$, forming hypothesis set $\{\mathcal{H}^i\}_1^{n'}$;
6: Form final model F by combining $\{\mathcal{H}^i\}_1^{n'}$, using $\mathcal{M}$;
7: **return** F.

5.3.1 View Construction of the MRC-Cross Algorithm

Using a relational database as input, the *View Construction* process of the MRC-Cross method constructs multiple *views*, each contains relevant features not only from the target relation but also from related background relations. This construction procedure consists of the following three stages. Firstly, the MRC-Cross method builds a undirected graph (relational domain graph) to describe the relevancies among relations in the relational database. Subsequently, the MRC-Cross strategy traverses the relational domain graph, using a heuristic search strategy, to extract a set of subgraphs. Each corresponds to a join path with multiple relations. Finally, *join operations* and *aggregation functions* are applied to each subgraph to convert a set of related tuples into a single entity. This results in forming *relational features* for each *candidate view*. Each contains knowledge on the target concept, not only from the target relation, but also from related background relations. Next, we discuss these three steps in detail.

Constructing Relational Domain Graph

In the first step of the view construction procedure, the MRC-Cross algorithm builds a relational domain graph to explicitly describe the join relationships between relations in a relational domain. As will be discussed next, through maintaining a graph representation, heuristic graph traversal techniques can be employed to collect related information distributed in relations.

Algorithm 5 Construct Relational Domain Graph

Input: Relational Database $\Re = T_t \times T_{b_1} \times \cdots \times T_{b_n}$.

Output: A Relational Domain Graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$.

1: $\mathcal{V} = \emptyset$, $\mathcal{E} = \emptyset$;
2: **for each** node $T_i \in \Re$ **do**
3: $\quad \mathcal{V}$.add(T_i)
4: $\quad$ **for each** $T_j \in \Re$ and $T_j \neq T_i$ **do**
5: $\quad\quad$ **for all** Join path $T_i \frac{\bowtie}{a=b} T_j$ **do**
6: $\quad\quad\quad$ **if** $\mathcal{E}(T_i \frac{\bowtie}{a=b} T_j) \notin \mathcal{E}$ **then**
7: $\quad\quad\quad\quad \mathcal{E}$.add($T_i \frac{\bowtie}{a=b} T_j$)
8: $\quad\quad\quad$ **end if**
9: $\quad\quad$ **end for**
10: $\quad$ **end for**
11: **end for**
12: **return** $\mathcal{G}(\mathcal{V}, \mathcal{E})$

A *Relational Domain Graph (RDG)* corresponds to an *undirected graphical model.* Let $\Re$ be a database with a target tables T_t, a set of background tables $T_{b_1}, \cdots, T_{b_n}$, and a set of joins. A *RDG* $\mathcal{G}(\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is a set of nodes and $\mathcal{E}$ a set of undirected edges ($\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$) corresponds to a relational database $\Re$. Each node in graph $\mathcal{G}$ is in one-to-one correspondence with a table in database $\Re$ (Here we denote the node which associates with the target table as the *target node* $\mathcal{V}_t$,

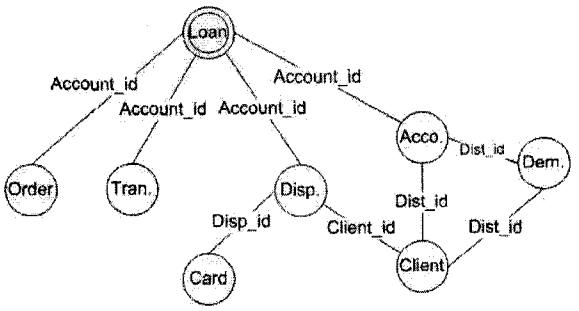


Figure 5.2: An example undirected graph

nodes associate with background tables are *view nodes* $\mathcal{V}_{b_1}, \cdots, \mathcal{V}_{b_n}$). Each edge in graph $\mathcal{G}$ is associated with a foreign key join $T_i \bowtie T_j$ between two tables, where $i \neq j$ and $T_i, T_j \in \Re$.

Note that the MRC-Cross algorithm allows the *RDG* to have multiple edges between two nodes, because different joins between two tables represent different semantic relationships between these two tables.

To build a *RDG* to capture how relations relate to one another in a database, the MRC-Cross algorithm, as shown in Algorithm 5, traverses all possible joins between any pair tables in a relational database and collects join information visited. In this way, the *RDG* explicitly describes the join relationships in the relational domain. Through traversing the *RDG*, the MRC-Cross algorithm therefore can explore the related relations of the domain.

Example 5.3-1 (Undirected Graph). In the example database of Figure 1.3, the join relationships of the database can be represented by the graph shown in Figure 5.2. The graph has eight nodes. Each corresponds to one of the eight tables in the database. Also, each edge corresponds to a foreign key join between two tables. The texts on each edges of the graph capture the join attributes between two relations.

Exploring Relational Domain Graph

After constructing a relational domain graph, the *View Construction* component traverses the graph to extract a set of subgraphs.

Definition 5.3-2 (Subgraph). Given a relational domain graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, a subgraph $\mathcal{G}_s(\mathcal{V}', \mathcal{E}')$ of $\mathcal{G}(\mathcal{V}, \mathcal{E})$ has $\mathcal{V}' \subseteq \mathcal{V}$ and $\mathcal{E}' \subseteq \mathcal{E}$. Each subgraph $\mathcal{G}_s$ is defined on a sequence of nodes.

Each subgraph $\mathcal{G}_s$ corresponds to a join path from the database $\Re$. The sequence of joins of a join path contains exactly the relations and joins that must be followed to calculate relational features for a specific view. Also, for each entity t in the target relation, a sequence of joins results in a set of tuples $T_t(T_i) = \{t_1, \ldots, t_j\}$ for each relation T_i within the join path.

The *View Construction* component starts to traverse the *RDG* from the *target node*, using a breath-first strategy. This search strategy visits all adjacent edges before going deeper. Thus, it fits the *View Construction* component's traversal preference, i.e. visiting shorter join paths first. Recall from Section 3.2.1 that the multiple views strategy discourages long join paths since the semantic link with these join paths usually becomes very weak in a relational database. The search approach, as described in Algorithm 6, consists of a number of rounds. Each round visits an unique sequence of nodes, starting from the *target node* $\mathcal{V}_t$ and stopping in a *view nodes* $\mathcal{V}_{b_i}$, visiting nodes following the edges. When reaching a *view nodes* $\mathcal{V}_{b_i}$ and the sequence of visited paths are unique, this round of search stops. The sequence of nodes visited is then collected as a subgraph $\mathcal{G}_s$. As described in Algorithm 6, the algorithm firstly visits and collects join paths with two tables, namely the number of joins is *one*. Then it search deeper by appending the collected subgraphs to one more relation, namely, join paths with three relations. It keeps expanding the collected subgraphs in the same way.

Example 5.3-3 (Travel the Graph). Let's continue the example 5.3-1, where the example database of Figure 1.3 has been mapped to a undirected graph (as presented in the left hand side of Figure 5.3). In this way, the MRC-Cross algorithm

can traverse the graph, start from the *Loan* table. Each round collects a unique join path from the graph. In the right hand side of Figure 5.3, we present three constructed subgraphs from the example database. There are join paths *Loan* ⋈ *Order*, *Loan* ⋈ *Disposition* ⋈ *Card*, and *Loan* ⋈ *Disposition* ⋈ *Client* ⋈ *Dem*, respectively.

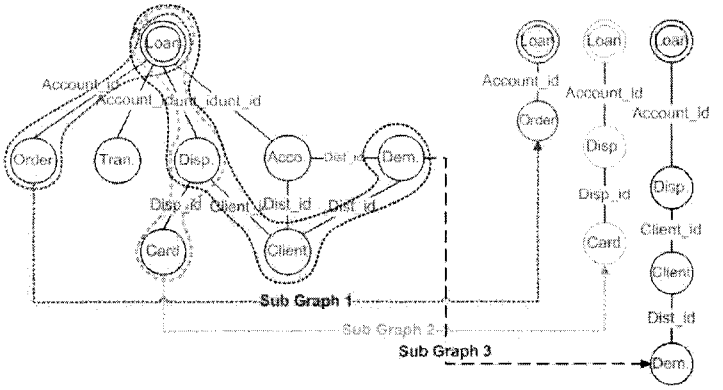


Figure 5.3: Travel the undirected graph

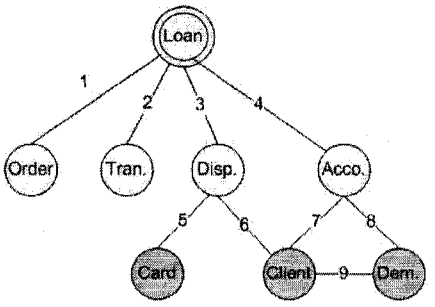
One constraint is imposed on each round of the graph traversal. That is, it is not allowed to revisit a node. This heuristic guarantees that each edge is visited once in each round of the search. Visiting cycles in the graph is also avoided. The intuition reason for this strategy is as follows. On the one hand, the search space in a relational domain with many relations is usually very large. It is unaffordable to exhaustively search the entire hypothesis space [242]. On the other hand, join paths with many relations and edges may decrease the number of entities related to the target tuples. Recall from Section 3.2.1 that the semantic link with too many joins usually becomes very weak in a relational database. Such weak links, therefore, tend to offer insufficient knowledge for learning the target concept. Therefore, to trade off between the efficiency and accuracy of the MRC-Cross algorithm, this restriction is introduced.

For the same above reason, given the complex structure in a graph, a stopping criterion is introduced to prevent the heuristic graph traversal from exploring very

long paths. The *View Construction* component sets a maximum number of joins as stopping criteria. When this predefined search depth is reached, the entire traversal stops, and the subgraphs collected are returned.

Finally, a *Redundancy Checking* method is applied to remove subgraphs which contain the same join path. Using the above traversal strategies, it can be seen that this process may contain redundant join paths. For example, the two join paths $T_i \underset{a=a}{\bowtie} T_j \underset{a=a}{\bowtie} T_k$ and $T_i \underset{a=a}{\bowtie} T_k \underset{a=a}{\bowtie} T_j$ contain the same information. Thus, when finishing collecting all subgraphs, a *Redundancy Checking* process is initiated to remove duplicate join paths. In addition, join paths ending with a relation which *only* contains non-descriptive attributes, i.e. all attributes are key attributes, are also removed, since these join paths add no more information to the learning.

Example 5.3-4 (Constructing All Join Paths). Consider continuing the example 5.3-3, where we have shown how three subgraphs are constructed. In the right hand side of Figure 5.4, we present all the constructed subgraphs from the example database. In this figure, each digital number corresponds to a foreign key join in the example database. The subgraphs marked with an x are redundant and will be removed by the *Redundancy Checking* process after the search is complete.



one join: 1,2,3,4

two joins: 3-5, 3-6, 4-7, 4-8

three joins: 3-6-7, 3-6-9, 4-7-6, 4-7-9(x), 4-8-9

four joins: 3-6-7-8, 3-6-9-8, 4-7-6-5, 4-8-9-6

five joins: 4-8-9-6-5

Figure 5.4: All subgraphs constructed from the example database

The next step describes the process of constructing *relational feature sets* from the subgraphs that have been collected.

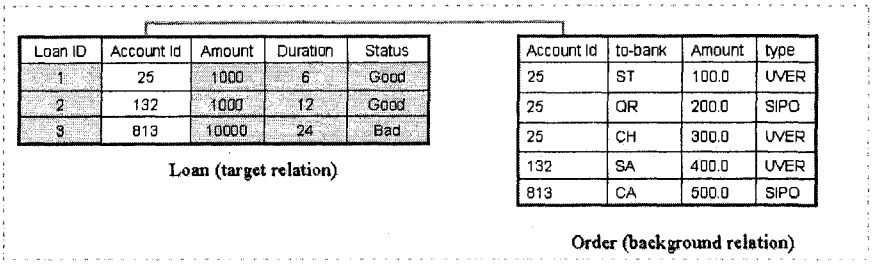
Constructing Relational Features

In this step, the *View Construction* component constructs relational feature sets for each view using the set of join path $\bowtie_n^c$ obtained. This process is formally defined as follows.

Assuming we have a join path $\bowtie_n^c = T_t \times T_{b_1} \times \cdots \times T_{b_n}$. For each tuple $t \in T_t$, the relational feature is:

$$\varphi(t) = \{\pi(t), \phi(\pi(\mathcal{T}_t(T_{b_i})))\}, \forall T_{b_i} \in \{T_{b_1}, \cdots, T_{b_n}\}$$

For each join path $\bowtie_n^c$, a set of tuple is formed, denoted as $\mathcal{S}_{\bowtie_n^c}$. Aggregation functions are then employed to generate relational features. In this way, each $\mathcal{S}_{\bowtie_n^c}$ separately creates a set of tuples with relational features $\varphi(t)$, forming a specific view for the MRC-Cross method. Also, the *View Construction* component uses all descriptive features from the target table to form a special view.



Loan ID	Account Id	Amount	Duration	Status
1	25	1000	6	Good
2	132	1000	12	Good
3	813	10000	24	Bad

Loan (target relation)

Account Id	to-bank	Amount	type
25	ST	100.0	UVER
25	QR	200.0	SIPO
25	CH	300.0	UVER
132	SA	400.0	UVER
813	CA	500.0	SIPO

Order (background relation)

a) Join Path 1

Loan ID	Amount	Duration	Status	Count (to-bank)	SUM (Amount)	AVG (Amount)	MIN (Amount)	MAX (Amount)	STDDEV (Amount)	Count (Amount)	Count (type)
1	1000	6	Good	3	600.0	200.0	100.0	300.0	100.0	3	3
2	1000	12	Good	1	400.0	400.0	400.0	400.0	0.0	1	1
3	10000	24	Bad	1	500.0	500.0	500.0	500.0	0.0	1	1

b) The view corresponding to Join Path 1 after generating relational features

Figure 5.5: Sample data after generating relational features for Join Path 1

Example 5.3-5 (Constructing Relational Feature). Let's continue the previous example, namely example 5.3-3, where we have created three subgraphs, i.e. join

paths 1, 2, and 3 (depicted in Figure 5.3). According to the MRC-Cross strategy, each of these join paths then generates a specific view through the use of aggregation function, namely through creating relational features. Here let's take a close look at how the view corresponding to join path 1 is created. As presented in example 5.2-2, this join path involves two relations, i.e. *Loan* and *Order*, and the SQL query for generating the relational features has been presented there. We here provide some sample instances of this example in Figure 5.5, where a) presents sample instances of both the *Loan* and *Order* relations from join path 1 and b) provides the sample data after creating relational features. From Figure 5.5 b), one can see that fourteen features exist in the view which corresponding to the join path 1. Among the fourteen, four of them are from the target table *Loan*, and the rest of them are aggregate features generated based on the *Order* background relation. In other words, the view collects all descriptive features from all involved relations in the join path, forming an attribute-based instance set which is able to be used as input for a single-table learning method.

The above section provides a detailed discussion of the view construction strategy of the MRC-Cross algorithm. Now, we simply describe the entire process of the MRC-Cross approach. As shown in Algorithm 4, the MRC-Cross algorithm initiates the *View Construction* procedure to explore the relational domain to construct multiple views, each consists of a set of attribute-based training instances with relational features. Subsequently, as employed by the MRC method, a *View Validation* component is launched to select a set of strongly uncorrelated views which are appropriate for multi-view relational learning. Finally, the *View Combination* mechanism combines the view learners to form a final classification model.

The next section presents our empirical evaluations performed against the MRC-Cross algorithm.

Algorithm 6 Search Graph

Input: Graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, Target Node $\mathcal{V}_t$ and View Nodes $\{\mathcal{V}_{b_1}, \cdots, \mathcal{V}_{b_n}\}$, Maximum number of joins $MaxJ$.

Output: Subgraph set $\{\mathcal{G}_{s_1}, \cdots, \mathcal{G}_{s_n}\}$.

Let $\mathcal{G}_s$.add($\mathcal{V}_t$), current number of joins ℓ= 0, current subgraph set $\mathcal{W}$=$\{\mathcal{V}_t\}$;
repeat
    ℓ++; Current Search Subgraph Set $\mathcal{S}$=$\mathcal{W}$;
    $\mathcal{W}$=$\emptyset$;
    for each subgraph $s \in \mathcal{S}$, $s = \{v_t \times \cdots \times v_{b_k}\}$ **do**
        for each edge e in $\mathcal{E}$ **do**
            if $e = \{v_{b_k} \times v_{b_n}\}$ and $v_{b_n} \notin s$ **then**
                append e to s, add s to subgraph $\mathcal{W}$
            end if
        end for
    end for
    for each $w \in \mathcal{W}$ **do**
        $\mathcal{G}_s$.add(w),
    end for
until $\ell \leq MaxJ$ or $\mathcal{W}$=$\emptyset$
return $\mathcal{G}_s$

5.4 Experimental Study

In this section, we present the performance studies of the MRC-Cross method against both benchmarking databases and synthetic databases.

5.4.1 Experiments against Benchmarking Databases

We first provide the performance results obtained for the MRC-Cross algorithm against various benchmarking databases. These results are presented in comparison with the MRC strategy in order to understand the benefits of constructing

individual views across multiple relations.

We implemented the MRC-Cross algorithm using Weka [232], and used the same experimental settings as that of Section 3.4. In other words, we here utilized the same data sets, namely databases MTU188, F682AC, F400AC, F234AC, ECML98, and THROM, in order to compare the performance of the two tested methods. We also ran the experiments on the same systems as being used in Section 3.4. In these experiments, the longest length of join path which the MRC-Cross algorithm traveled is *five (5)*.

Experimental Results

In this section we conduct three sets of experiments with varying settings in order to examine the performance of the MRC-Cross method, both in terms of the accuracy obtained and run time needed. These three sets of experiments used different propositional learning algorithms as the view learners and meta learners for the MRC and MRC-Cross approaches, in order to evaluate how different propositional strategies impact the performance of the two methods.

In the first and second experiments, C4.5 decision trees were used as propositional learners. In order to examine the impact of the meta learners on the MRC-Cross methods, these two experiments applied different meta learners: the C4.5 decision trees were used by the former, and Naive Bayes classifiers were used in the latter. To evaluate the effort of different propositional learning methods, the third experiment applied the Naive Bayes classifiers as view learners and C4.5 decision trees as meta learners of the two tested systems.

Experiment #1: Using Decision Trees as view learners as well as meta learners

In the first experiment, we aim to evaluate the performance of the MRC-Cross algorithms in terms of accuracy obtained and running time needed. In this experiment, C4.5 decision trees were used as the view learners and meta learners for the

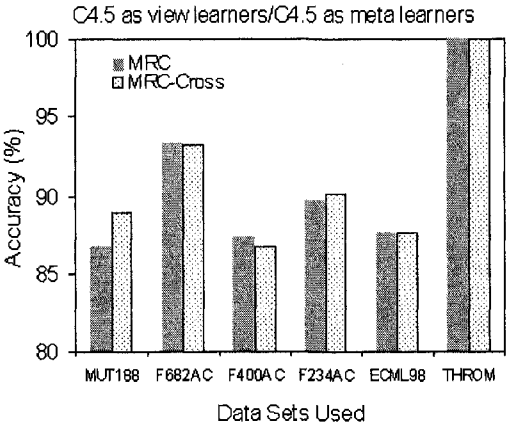


Figure 5.6: Accuracies obtained using C4.5 as view learners and meta learners (%) for the MRC and MRC-Cross algorithms

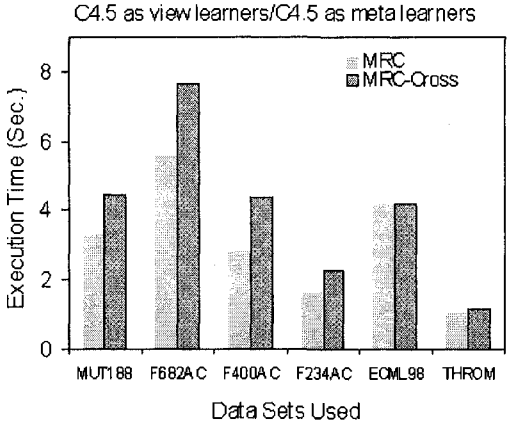


Figure 5.7: Running time required using C4.5 as view learners and meta learners for the MRC and MRC-Cross algorithms (the graphing of the running time against the ECML98 data set was scale-downed by a factor of 100)

MRC-Cross and MRC algorithms.

We present the predictive accuracy obtained for each of the six learning tasks in Figure 5.6. To evaluate the performance of the MRC-Cross strategy in terms of execution time required, we also provide the running time needed (in seconds) for each learning tasks in Figure 5.7. Note that the graphing of the running time against the ECML98 data set was scale-downed by a factor of 100 in order to clearly show all bars in the same graph.

Discussion of Experiment #1

The predictive performance results, as presented in Figure 5.6, show that the MRC-Cross and MRC algorithms appear to produce very comparable accuracy outcomes. For example, when against the F682AC, ECML98, and THROM data sets, the experimental results indicate that the two compared methods almost obtained the same accuracy results. Against the other three data sets, i.e. MUT188, F234AC and F400AC, the MRC-Cross strategy yielded slightly lower accuracy than that of the MRC method in the first two cases (MUT188 and F234AC), but it achieved predictive gains over that of the MRC algorithm when against the last tested case, namely against the F400AC data set.

In terms of running time needed, one can see from the experimental results (shown in Figure 5.7) that the MRC-Cross and MRC methods required very close running time. The execution time difference of these two tested methods against all six data sets were less than 2 seconds. In all cases, the MRC-Cross algorithm required a bit more time than that of the MRC method because of the search and construction of relational features.

Experiment #2: Using Decision Trees as view learners and Naive Bayes as meta learners

The second experiment aims to investigate the impact of the meta learning algorithms on the MRC-Cross approach. In this experiment, we used C4.5 decision

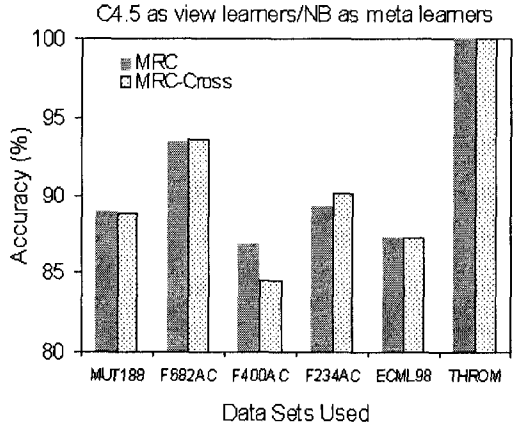


Figure 5.8: Accuracies obtained using C4.5 as view learners and Naive Bayes meta learners (%) for the MRC and MRC-Cross algorithms

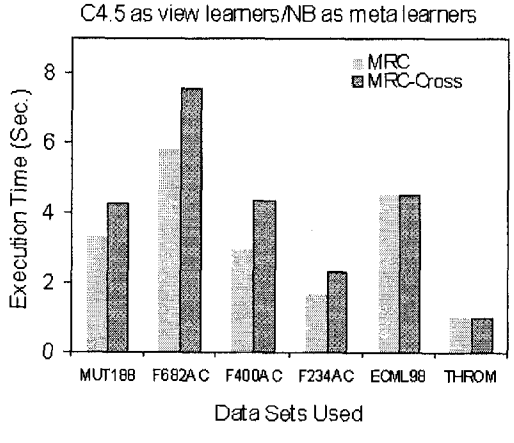


Figure 5.9: Running time required using C4.5 as view learners and Naive Bayes meta learners for the MRC and MRC-Cross algorithms (the graphing of the running time against the ECML98 data set was scale-downed by a factor of 100)

trees as view learners of the MRC-Cross and MRC algorithms (as we did in experiment #1). However, we applied Naive Bayes as the meta learners of both the MRC-Cross and MRC algorithms in this setting, instead of using C4.5 decision trees.

We present the predictive accuracy obtained for each of the six learning tasks in Figure 5.8. To evaluate the performance of the MRC-Cross strategy in terms of run time, we also provide the execution time needed (in seconds) for each learning tasks in Figure 5.9. Again, the graphing of the running time against the ECML98 data set was scale-downed by a factor of 100 in order to clearly show all bars in the same graph.

Discussion of Experiment #2

The predictive performance results, as presented in Figure 5.8, show that the accuracy results produced by the MRC-Cross and MRC algorithms appear to be very comparable. These results show that against four of the six data sets, namely against the MUT188, F682AC, ECML98, and THROM data sets, the two tested methods almost obtained the same accuracy results. In addition, although the MRC-Cross strategy yielded slightly lower accuracy than that of the MRC method when against the F234AC data sets, but it achieved predictive gains over that of the MRC algorithm when against the F400AC data set.

When considering running time needed, one can see from the experimental results (shown in Figure 5.9) that execution time required by the MRC-Cross and MRC methods against all tested cases is very close. The time difference of these two tested methods against all six data sets were less than 2 seconds. Against, in all cases, the MRC-Cross algorithm required slightly more time than that of the MRC method due to the construction of relational features.

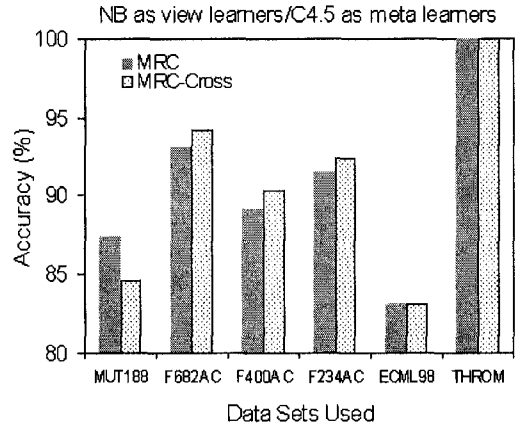


Figure 5.10: Accuracies obtained using Naive Bayes as view learners and C4.5 as meta learners (%) for the MRC and MRC-Cross algorithms

Experiment #3: Using Naive Bayes as view learners and Decision Trees as meta learners

In the last experiment, we aim to evaluate the performance impact of the MRC-Cross algorithm when different propositional learning methods are applied as view learners. Contrary to the settings in the experiments #1 and 2, we here used Naive Bayes as the multiple view learners of the MRC-Cross and MRC algorithms (where the C4.5 decision trees were employed as meta learners).

We present the predictive accuracy obtained for each of the six learning tasks in Figure 5.10. In order to evaluate the performance of the MRC-Cross strategy in terms of run time, we also provide the execution time required (in seconds) for each learning tasks in Figure 5.11. Again, the graphing of the running time against the ECML98 data set was scale-downed by a factor of 100 in order to clearly show all bars in the same graph.

Discussion of Experiment #3

The predictive performance results, as presented in Figure 5.10, show that the MRC-Cross and MRC algorithms appear to produce very comparable predictive

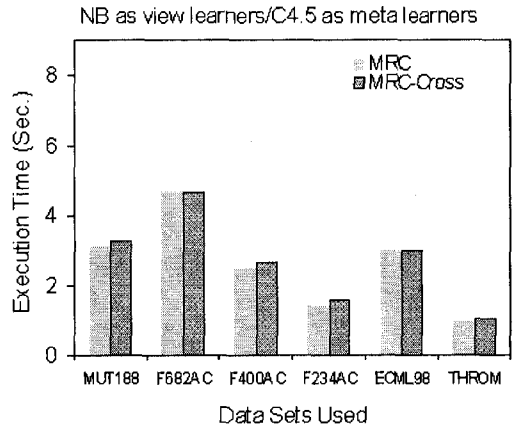


Figure 5.11: Running time required using Naive Bayes as view learners and C4.5 as meta learners for the MRC and MRC-Cross algorithms (the graphing of the running time against the ECML98 data set was scale-downed by a factor of 100)

performance results in terms of accuracy obtained. For example, as shown in Figure 5.10, against all the six data sets, namely against the F682AC, F400AC, F234AC, ECML98, MUT188, and THROM data sets, the two tested methods almost obtained the same accuracy results. One exception is against the MUT188 data, where the MRC-Cross strategy yielded slightly better accuracy than that of the MRC approach.

When considering running time needed, one can see from the experimental results (shown in Figure 5.11) that the MRC-Cross and MRC methods required very close running time. Against each of the tested cases, the difference of the required running time of the two tested strategies is very small.

Summary of the three experiments

In summary, the three experimental results (shown in Figures 5.6, 5.7, 5.8, 5.9, 5.10, and 5.11) indicate that the MRC-Cross and MRC approaches achieved very close results, when evaluated in terms of overall accuracy obtained. Such comparable results were yielded regardless of the view learners and meta learners employed. In other words, the study presented here suggests that, for the multiple

view learning framework, constructing individual views from multiple tables yields little performance gains, in terms of accuracy obtained, over that of forming each individual view only from a sole relation.

a) Decision tree conducted from the view Loan

```
LOAN_payment <= 6930: 1 (338.0/48.0)
LOAN_payment > 6930
| LOAN_date <= 950530: 2 (13.0/1.0)
| LOAN_date > 950530: 1 (49.0/16.0)
```

b) Decision tree conducted from the view Loan ⋈ Account

```
LOAN_payment <= 6930: 1 (338.0/48.0)
LOAN_payment > 6930
| LOAN_date <= 950530: 2 (13.0/1.0)
| LOAN_date > 950530: 1 (49.0/16.0)
```

c) Decision tree conducted from the view Loan ⋈ Account ⋈ Demographic

```
LOAN_payment <= 6930: 1 (338.0/48.0)
LOAN_payment > 6930
| LOAN_date <= 950530: 2 (13.0/1.0)
| LOAN_date > 950530
| | LOAN_duration <= 24
| | | Demographic _a114_SUM <= 10177: 1 (14.0)
| | | Demographic _a114_SUM > 10177
| | | | LOAN_date <= 970928: 2 (2.0)
| | | | LOAN_date > 970928: 1 (2.0)
| | LOAN_duration > 24
| | | Demographic_a94_SUM <= 7
| | | | Demographic_a144_SUM <= 145: 2 (19.0/6.0)
| | | | Demographic_a144_SUM > 145: 1 (4.0)
| | | Demographic_a94_SUM > 7: 1 (8.0/1.0)
```

Figure 5.12: Dominating features in induced decision trees

Our further analysis of the results suggests one possible reason for the above observation. That is some very "powerful" features (in terms of predictivity capability of the class label) may dominate multiple views in the MRC-Cross strategy due to its collection of features from related tables. In this way, the ability of building diverse view learners of the multiple views framework is harmed. As an example, Figure 5.12 depicted three trees built using C4.5 decision trees against three different views of the F400AC database. These three views were constructed from join paths *Loan*, *Loan* ⋈ *Account*, and *Loan* ⋈ *Account* ⋈ *Demographic*, respectively. The trees constructed by these three views are provided in parts a), b), and c) of Figure 5.12, respectively. From these figures, one can see that the views *Loan* ⋈ *Account* and *Loan* ⋈ *Account* ⋈ *Demographic* consists of more than one table, thus the view learners can explore features from the related multiple tables.

However, Figures a), b), and c) indicate that, the decision trees inducted from these three views were dominated by two very "powerful" features, i.e. *payment* and *date* from the *Loan* relation (highlighted in red in the figures). This observation suggests that in this case, the MRC-Cross method failed to construct diverse view learners.

5.4.2 Experiments against Synthetic Databases

In order to examine the performance of the MRC-Cross strategy with respect to the *maximum length of join path* $MaxJ$, we generated a series of synthetic databases with different characteristics. The database generator was created by Yin et al. in [242]. Using this generator, users can specify the expected number of tuples, attributes, relations, and joins, etc., in a database to obtain various kinds of databases.

For each database in this experiment, we set the expected number of tuples and attributes as 1000 and 15, respectively. Seven such databases were generated with 10, 20, 30, 50, 80, 100, and 150 relations, respectively (denoted as R10, R20, R30, R50, R80, R100, and R150, respectively). In this experiment, C4.5 decision trees were used for the multi-view learners as well as for the meta-learners.

Against each of the seven synthetic databases, the MRC-Cross strategy varied its $MaxJ$ value from two (2) to seven (7). Recall that the value of $MaxJ$ sets the maximum searching depth while the MRC-Cross method is exploring various join paths in the database. In other words, we here allow a join path to involve up to eight (8) tables. We provided the predictive accuracy obtained and running time required by the MRC-Cross methods in Figure 5.13.

From Figure 5.13, one can see that when $MaxJ$ equals to three, the MRC-Cross method provided us a good trade-off between predictive performance obtained and execution time needed. Against four of the seven databases (each tested with six different $MaxJ$ values), the MRC-Cross algorithm obtained the highest accuracy when the maximum length of join path was set to three. Three exceptions were

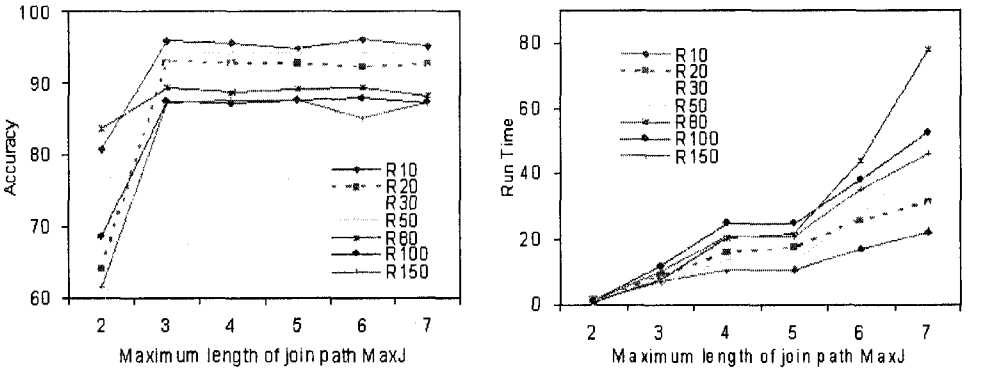


Figure 5.13: Accuracy obtained and execution time required w.r.t maximum length of Join Path

against the databases R10, R30, and R100 when the $MaxJ$ value was set to six. In these three cases, the MRC-Cross approach with $MaxJ$ of three achieved slightly lower accuracy (less that 1%), compared to that of setting the $MaxJ$ to six. However, the execution time required (as provided on the right hand side of the figure) suggests that the running time needed for the tested cases with the $MaxJ$ value of six may double, compared to that of cases with value of three. For example, against databases R80 and R100, the running times required for the $MaxJ$ value of six were at least three times more than that of setting the $MaxJ$ value to three. Further analysis of the results on the right hand side of the figure suggest that in most of the tested cases, when $MaxJ$ equals to three, the MRC-Cross algorithm required reasonable execution time. In order words, this number provided us a good trade-off between accuracy achieved and execution time needed.

In order to further test this heuristic number, we compared the MRC-Cross algorithm with the CrossMine method since our empirical results from Chapter 3 demonstrate that the CrossMine method achieved promising accuracy, and scales well, when compared with the other tested relational algorithms. In these experiments, we set the $MaxJ$ value to *three (3)*. This number was determined heuristically based on our previous experimental observations.

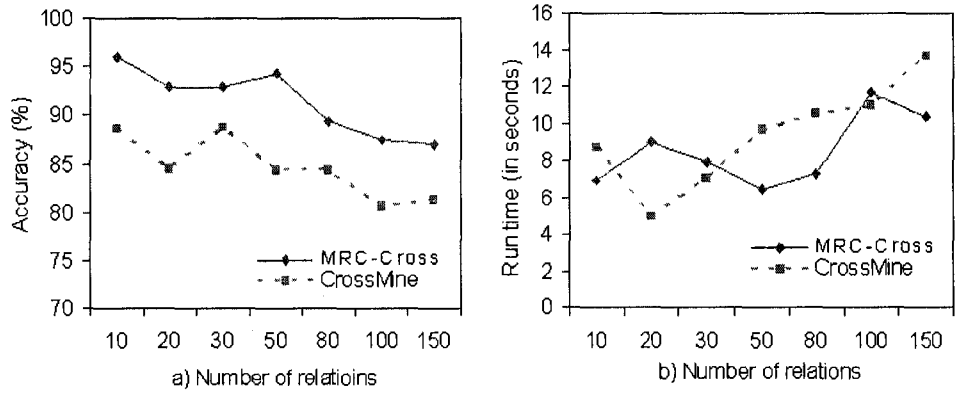


Figure 5.14: Accuracy obtained and execution time needed, compared to CrossMine

The above seven synthetic databases were used. Figures 5.14 (a) and (b) provide the accuracies obtained and the run time needed for the MRC-Cross and the CrossMine methods. The results suggest that, when using the heuristic number of three, the MRC-Cross algorithm was able to have a good trade-off between accuracy and execution time required. The results indicate that the MRC-Cross algorithm consistently reduces the error rate for all of the databases, without increasing the execution time required in comparison to the CrossMine method. Further analysis of our experimental results show that, in some databases, the accuracy improvements are very meaningful. For example, in the databases with 20 and 50 tables, the predictive accuracy improvements are more than 10%, when compared to the CrossMine approach.

In summary, our experimental results against the seven synthetic databases with different number of relations suggest that, when the $MaxJ$ equals to three, the MRC-Cross algorithm achieved very good accuracy results, when compared with both the MRC-Cross strategy with other different $MaxJ$ values and the CrossMine method. The results indicate that, in most cases searching for longer join paths did not benefit the accuracy of the MRC-Cross approach. Also, the MRC-Cross method with such a number run very fast, even when compared with the CrossMine strategy, which has demonstrated its efficiency for complex relational learning problems [242].

5.5 Conclusions

In this chapter we extended the MRC framework in order to answer the following question. Does the multiple views algorithm gains further performance improvement, if its view construction component is capable of combining features from multiple tables in the database? To this end, we enabled the MRC-Cross method to collect relevant features across multiple tables into an individual view through the use of relational features.

Our experimental results produced against various datasets show that the MRC-Cross and MRC strategies performed very closely in terms of accuracy obtained. The study presented here suggests that, for the multiple view learning framework, constructing individual views from multiple tables yields little performance gains, in terms of accuracy obtained and execution time needed, over that of forming each individual view only from a sole relation.

Further studies of the empirical results suggest that the comparable predictive performance of the MRC-Cross and MRC methods are probably due to the following reasons. On the one hand, the characteristics of the structural schemas of relational databases naturally fit the multiple view framework well. That is, in a relational database, features in each table are often naturally divided. In other words, the feature sets in individual relations are often disjoint. Such a characteristic of relational databases fits the MRC framework. On the other hand, when collecting related features from different relations, some "powerful" features from certain relations may prevent the MRC-Cross approach from constructing diverse view learners.

Our studies here also imply that when collecting related features from a complex schema database, there is no need to search relations which are very far, in terms of the number of foreign key joins involved, from the target relation. This implication also justifies the observations as presented in 5.3.1. That is, semantic link with long join paths usually becomes very weak in a relational database. In other words, in a multirelational classification task, relations which are closer to the target relation

should receive more attentions.

The next chapter will present an approach for pre-pruning uninteresting sub-structures of multi-relational databases in order to reduce the learning scale of multi-relational classification tasks.

CHAPTER 6

Pruning Relations of Multi-relational Databases

Chapter 6

Pre-pruning Relations of Multi-relational Databases

The previous chapter presented a strategy which enables the MRC framework to collect related features across multiple relations. In this chapter, we further extend this method to develop an approach for pre-pruning relational databases in order to reduce the learning scale of relational classification tasks.

In practice, multirelational data mining usually involves a large number of interconnected relations collected from multiple parties. Different phases of the knowledge discovery process in such data are affected by economic utility. For instance, in the data preprocessing process, one must consider the cost associated with acquiring the training data, cleaning the data, and transforming the data. When training and testing the data mining models, one has to consider the impact of the data on the running time of the learning algorithm. In order to address such utility-based issues, our method aims to create a pruned structure while minimizing predictive performance loss on the final classification model. The approach initially *decomposes* the relational domain into subgraphs. From this set, it subsequently identifies a subset of subgraphs which are highly diverse, but correlated with the target class. All other subgraphs are discarded. Our algorithm not only reduces the size of the relational schema, but also produces compact pruned schemas that

provide comparable multi-relational classification models in terms of the accuracy obtained.

We start this chapter with an overview of the pruning algorithm. Next, a detailed discussion of the pruning strategy is then provided. Finally, we conclude the chapter with a presentation of experimental studies.

6.1 Overview

To our best knowledge, little attention has been paid to pre-prune relational schemas for the purpose of improving multi-relational data mining. Cohen [41] introduced a method to filter irrelevant literals out of relational examples in a text mining context. In the strategy, literals refer to words and relations between words, and literals with low frequency in the learning corpus are considered as less irrelevant. Alphonse and Matwin [74] presented a literal pruning approach for ILP learning problems. The method in [74] first approximates the ILP problem with a bounded multi-instance task [248]. Here, boolean attributes are used to describe corresponding literals in the ILP example set and a set of new created instances is created from each relational example. Next, the method employs a relief-like algorithm to filter these boolean features in the resultant multi-instance problem. Finally, the reduced boolean features are then converted back to the relational representation. In this way, relational examples of the original ILP problem are pruned. Singh et al. proposed a method to pre-prune social networks (consist of nodes and edges) using both structural properties and descriptive attributes in [216]. Their algorithm selects nodes according to the number of connections they have. Those nodes deemed not to have enough connections are removed. Edges are also pruned based on their descriptive attribute values. Habrard et al. [100] described a probabilistic method to prune noisy or irrelevant subtrees based on the use of confidence intervals on tree-structured data. They aim to improve the learning of a statistical distribution over the data, instead of constructing a classification model.

We here propose the Subgraph Ensemble Structure Pruning (SESP) method for pre-pruning relational databases. The goal of the SESP approach is to create a pruned relational schema that models only the most informative substructures, while maintaining satisfactory predictive performance. This is achieved by removing either *irrelevant* or *redundant* substructures from the relational database. The SESP algorithm assumes that strongly correlated substructures contain redundant information, since these substructures can predict or imply each other, and provide similar information with one another (as discussed in Section 3.3.1). Substructures which are weakly correlated to the class are said to be of low relevance, because they contain little useful information in terms of helping predicting the class.

In contrast to related work as discussed above, the SESP approach uses a subgraph-based strategy, which aims to pre-prune relational database for classification. The SESP approach initially *decomposes* the relational domain into subgraphs. From this set, it subsequently prunes subgraphs which are considered as *irrelevant* or *redundant*.

6.1.1 Problem Setting and Example Task

The problem setting for our SESP strategy is defined by a relational database schema $\Re$. Recall from Chapter 1 that $\Re$ is described by a set of tables $\{R_1, \cdots, R_n\}$ and a set of joins (J) among tables. Taking a relational database $\Re = \{R_t, R_b, J\}$ as input, the SESP method aims to form a pruned version $\Re' = \{R_t, R'_b, J'\}$ for which the predictive accuracy of a relational classifier constructed from it (and used on target relation R_t) is as high as for one built on $\Re$. That is, we aim to prune the relational structure without loss of predictive accuracy.

To help understand the problem setting, consider the following example task that applies the SESP strategy. Suppose that a bank is interested in using the example database from Figure 6.1 to predict a new customer's risk level for a *personal loan*. The database consists of five tables and four relationships. Tables *Demographic, Transaction, Life Style* and *Order* are the background relations and

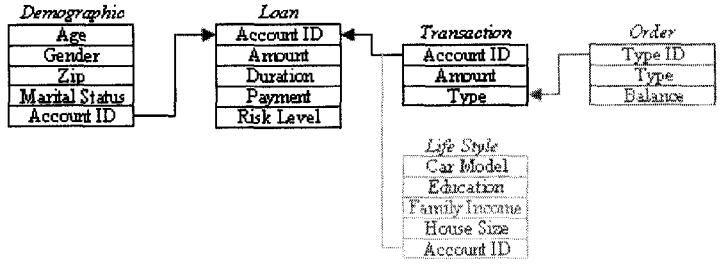


Figure 6.1: A sample database

Loan is the target relation. Multi-relational data mining algorithms often need to use information from both the five relations and the four join relationships to build a relational model to categorize each tuple from the target relation *Loan* as either *High Risk* or *Low Risk*. However, suppose that the *Order* and *Life Style* relations contain private information with restricted access privileges. Let us consider a subset of the relational model without the *Order* and *Life Style* tables and their corresponding relationships. If this reduced relational model can generate comparable accuracy, we can take advantage of some obvious benefits: 1) the cost for acquiring and maintaining the data from the *Order* and *Life Style* relations can be avoided; 2) the information held by the *Order* and *Life Style* tables is not used when training and deploying the model, thus enhancing privacy preservation.

In the above example, the proposed SESP method works as follows. First, it decomposes the sample database into a set of subgraphs. Each subgraph corresponds to a join path (or slot chain) starting from the target table *Loan*. That is, subgraphs $Loan \bowtie Demographic$, $Loan \bowtie Transaction$, $Loan \bowtie LifeStyle$, $Loan \bowtie Transaction \bowtie LifeStyle$, and $Loan \bowtie Transaction \bowtie Order$, as well as the subgraph involving just the target relation *Loan*, will be created. Second, the correlations between these six subgraphs are evaluated. Subsequently, a subset of strongly uncorrelated subgraphs, ($Loan \bowtie Demographic$, and $Loan \bowtie Transaction$ in this case) is identified. Finally, irrelevant and/or redundant relations *Order* and *Life Style* along with their relationships are pruned (shown as gray in Figure 6.1). The final pruned relational schema consists of the non-gray

elements of Figure 6.1.

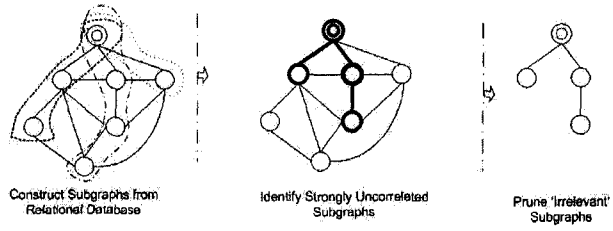


Figure 6.2: The core idea of the SESP approach

In the next section, the details of the SESP algorithm are presented.

6.2 The SESP Pruning Approach

The core idea of the SESP method, as shown in Figure 6.2, is to identify a small set of strongly uncorrelated subgraphs given a database schema. The process, as described in detail in Algorithm 7, consists of two key steps: 1) to initially decompose the relational database schema into subgraphs (*subgraph construction*) and 2) to identify the subset of strongly uncorrelated subgraphs (*subgraph evaluation*). Each of these steps is discussed next.

6.2.1 Subgraph Construction

The purpose of the *subgraph construction* process is to build a set of subgraphs given a relational database schema. Each such subgraph corresponds to a unique join path which contains the target relation. In fact, the purpose here states the exact goals of the view construction component of the MRC-Cross strategy as presented in Section 5.3.1. Recall from Section 5.3.1 that the view construction process in the MRC-Cross method initially converts the relational database schema into an undirected graph, using the tables as the nodes and joins as edges. The graph is then traversed to extract a set of unique subgraphs, each mapping to a unique join path in the relational database. Basing on this observation, we here, therefore, use

Algorithm 7 The SESP Pruning Approach

Input: A relational database $\Re = \{R_t, R_b, J\}$

Output: A pruned database $\Re' = \{R_t, R'_b, J'\}$

1: Divide $\Re$ into training data set $\mathcal{T}_t$ and evaluation data set $\mathcal{T}_m$;
2: Convert $\Re$ into undirected graph $\mathcal{G}$;
3: Decompose $\mathcal{G}$ into a set of subgraphs $\{\mathcal{S}\}$;
4: Build subgraph-based classifier for each subgraph in $\{\mathcal{S}\}$, using $\mathcal{T}_t$;forming $\{C_d^1, \cdots, C_d^m\}$;
5: Let subgraph set $C = \emptyset$;
6: Generate an evaluation examples set $\mathcal{T}'_m$, using $\mathcal{T}_m$ and $\{C_d^i\}_1^m$;
7: Select a subgraph feature set $\mathcal{A}'$ from $\mathcal{T}'_m$;
8: **for each** C_d^i which has at least one attribute in $\mathcal{A}'$ **do**
9: $\quad C$.add(C_d^i);
10: **end for**
11: Form $\{C^1, \cdots, C^k\}$ $(k \leq m)$;
12: Remove duplicate relations and relationships from $\{C^1, \cdots, C^k\}$; forming $\Re' = \{R_t, R'_b, J'\}$;
13: **return** $\Re'$.

the view construction strategies employed in the MRC-Cross algorithm in order to construct the set of subgraphs for the SESP strategy.

Example 6.2-1 (Subgraphs). Figure 6.3 shows all six (6) subgraphs (from SG1 to SG6) constructed from the sample database (in Figure 6.1). In this figure, SG1 depicts the subgraph which consists of only the target relation. SG2, SG3, and SG4 describe subgraphs with two involved relations. Subgraphs containing three tables are shown in SG5 and SG6. Also, the original database schema and the pruned schema are depicted in the leftmost and rightmost sides, respectively, of the figure.

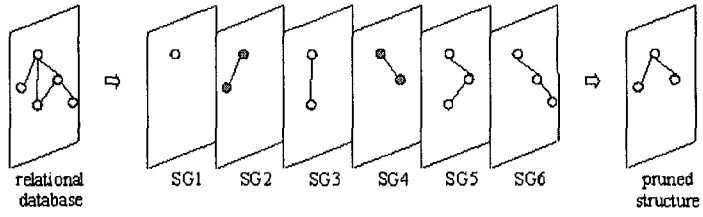


Figure 6.3: Search and construct subgraphs

In this stage of the SESP algorithm, the information contained in a relational database schema is decomposed into a number of subgraphs. All unnecessary subgraphs are identified and pruned by the *subgraph evaluation* process, discussed next.

6.2.2 Subgraph Evaluation

The SESP strategy aims to identify a set of strongly *uncorrelated* subgraphs from the original relational schema only. In order to select which subgraphs will be best for classification, subgraphs are evaluated according to the following methodology. Firstly, each of the created subgraphs is used to construct a separate relational classifier (*subgraph-based relational classification*). Secondly, the constructed classification models are used to generate an evaluation data set in which each instance is described by sets of feature sets (*subgraph features*). Each feature set describes the knowledge contained in one of the classification models. Thirdly, the set of *subgraph features* (from this evaluation data set) which are strongly uncorrelated with each other, but correlated with the target class, are identified. These subgraph feature sets *correspond*, therefore, to a set of subgraphs. Lastly, subgraphs which are not selected are discarded. Each of these four steps is discussed in detail next.

Subgraph-based Relational Classification

Each subgraph created in Section 6.2.1 can be used to build a relational classifier using single-table learning algorithms. In order to do so, we here apply the *relational feature construction* strategy employed in the MRC-Cross method (as presented in Section 5.3.1). Recall from Section 5.3.1 that, through generating relational features, each subgraph can separately "flatten" into a set of attribute-based training instances. Traditional learning algorithm such as decision trees and SVMs can therefore be applied to learn the relational target concept, forming a set of subgraph-based classifiers.

Subgraph Features

Subgraph features are used to describe the knowledge held by subgraph-based classifiers and represent the corresponding subgraphs. The subgraph features are generated as follows. Let $\{C^1, \cdots, C^n\}$ be n classifiers (each is built using a different subgraph). Let $\mathcal{T}_m$ be an *evaluation data set* with m labels $\{y_1, \cdots, y_m\}$. For each instance t (with label L) in $\mathcal{T}_m$, each classifier is called upon to produce predictions $\{f^{y_k}_{C^i}(t)\}$ ($i \in \{1, \cdots, n\}$ and $k \in \{1, \cdots, m\}$) for it. Here, $f^{y_k}_{C^i}(t)$ denotes the probability that instance t belongs to class y_k, as predicted by classifier C^i. In this way, a set of *evaluation examples* is constructed where each consists of sets of prediction set $\{P^{y_k}_{C^i}(t)\}$ (each describes the hypothesis knowledge of the corresponding classifier) and the original class labels L. For instance, C^1 is described and represented by features $\{f^{y_k}_{C^1}(t)\}$ ($k \in \{1, \cdots, m\}$). We define $\{f^{y_i}_{C^1}(t)\}$ to be the *subgraph features* of classifier C^1.

Example 6.2-2 (Subgraph Feature). As an example, let's continue the running example as presented in Example 6.2-1. Recall from Example 6.2-1 that, the example database is decomposed into six subgraphs, namely SG1, SG2, SG3, SG4, SG5, and SG6, and these six subgraphs correspond to six join paths of the example database, i.e. *Loan*, *Loan* $\bowtie$ *Demographic*, *Loan* $\bowtie$ *Transaction*, *Loan* $\bowtie$ *LifeStyle*, *Loan* $\bowtie$ *Transaction* $\bowtie$ *LifeStyle*, and *Loan* $\bowtie$ *Transaction* $\bowtie$ *Order*. In other

words, as discussed in the previous section, six subgraph-based relational classifiers (denoted as C^1, C^2, C^3, C^4, C^5, and C^6) are constructed, each built from one of the six join paths. In this learning task, the target variable *Risk Level* in the target relation *Loan* has two values, namely *High Risk* and *Low Risk*. Hence, for each tuple t in the evaluation data set, each classifier $C^i, i \in \{1 \cdots 6\}$, generates two probabilistic predictions $f^{C^i}_{high}$ and $f^{C^i}_{low}$. For example, subgraph SG1 creates two subgraph features f^{sg1}_{high} and f^{sg1}_{low}. Similarly, two subgraph features f^{sg2}_{high} and f^{sg2}_{low} are generated by subgraph SG2. Figure 6.4 shows four (4) examples generated by the six subgraphs. Each of the four examples also contains the original class labels of the evaluation data.

SG1		SG2		SG3		SG4		SG5		SG6		Class label
f^{sg1}_{high}	f^{sg1}_{low}	f^{sg2}_{high}	f^{sg2}_{low}	f^{sg3}_{high}	f^{sg3}_{low}	f^{sg4}_{high}	f^{sg4}_{low}	f^{sg5}_{high}	f^{sg5}_{low}	f^{sg6}_{high}	f^{sg6}_{low}	
0.93	0.07	0.21	0.79	0.73	0.27	0.01	0.99	0.43	0.57	0.51	0.49	High Risk
0.66	0.34	0.32	0.68	0.46	0.54	0.12	0.88	0.86	0.14	0.82	0.18	Low Risk
0.89	0.11	0.81	0.19	0.69	0.31	0.61	0.39	0.39	0.61	0.31	0.69	Low Risk
0.22	0.78	0.47	0.53	0.02	0.98	0.27	0.73	0.73	0.27	0.97	0.03	High Risk

Figure 6.4: Sample subgraph feature set generated by the six subgraphs

In this way, examples with subgraph features are created. Next, the degree of correlation between the different subgraph features is calculated.

Subgraph Evaluation and Pruning

Recall that the goal of the SESP algorithm is to keep the most informative subgraphs while pruning those deemed to be redundant. Therefore, the SESP strategy prefers disjoint knowledge obtained by subgraphs. In other words, the SESP algorithm is only interested in finding the set of subgraphs which are strongly uncorrelated with each other, but correlated with the target class. In this sense, the *view validation* component from the MRC algorithm (as presented in Section 3.3.1) can be applied. Recall from Section 3.3.1 that the *view validation* component of the MRC method aims to identify a set of views which are strongly correlated with the

class but irrelevant to each other. The calculation of such a subset of views is based on two heuristic methods, namely the "goodness" metric $\mathcal{C}$ and the *Symmetrical Uncertainty* measurement $\mathcal{U}$.

As presented in Section 3.3.1, in order to identify a set of uncorrelated subgraphs, the evaluation procedure explores different subgraph feature subsets, and constructs a ranking on them. The best ranking subset will be selected, i.e. the subset with the highest $\mathcal{C}$ value.

Subgraphs are selected based on the final best subset of subgraph features. Subgraph features in this subset are uncorrelated with one another. Also, this subset is considered to be the one with elements most highly correlated with the target concept. If a subgraph has no features that are strongly correlated to the class to be learned, the knowledge possessed by this subgraph can be said to be unimportant for the task at hand. Thus, it makes sense to prune this subgraph. The SESP algorithm, therefore, keeps a subgraph *if* and *only if* any of its *subgraph features* appear in the final best ranking subset.

SG1		SG2		SG3		SG4		SG5		SG6		Class label
f^{sg1}_{high}	f^{sg1}_{low}	f^{sg2}_{high}	f^{sg2}_{low}	f^{sg3}_{high}	f^{sg3}_{low}	f^{sg4}_{high}	f^{sg4}_{low}	f^{sg5}_{high}	f^{sg5}_{low}	f^{sg6}_{high}	f^{sg6}_{low}	
0.93	0.07	0.21	0.79	0.73	0.27	0.01	0.99	0.43	0.57	0.51	0.49	High Risk
0.66	0.34	0.32	0.68	0.46	0.54	0.12	0.88	0.86	0.14	0.82	0.18	Low Risk
0.89	0.11	0.81	0.19	0.69	0.31	0.61	0.39	0.39	0.61	0.31	0.69	Low Risk
0.22	0.78	0.47	0.53	0.02	0.98	0.27	0.73	0.73	0.27	0.97	0.03	High Risk

Figure 6.5: Identified subgraph feature set

Example 6.2-3 (Subgraph Pruning). As an example, let's continue the running example 6.2-2. Recall that, twelve subgraph features (as shown in Figure 6.4) have been constructed for the six subgraphs, i.e. subgraphs SG1, SG2, SG3, SG4, SG5, and SG6. Consider the SESP algorithm identified a final subgraph feature subset with two features, namely, $f^{sg_2}_{high}$ and $f^{sg_4}_{high}$ (as highlighted in blue cells in Figure 6.5). This subset implies that only knowledge from subgraphs SG2 and SG4 really contribute to building the final model. Thus, subgraphs SG2 and SG4

are selected by the *subgraph evaluation* method. All other subgraphs are pruned, because they are considered as either irrelevant or redundant. In this way, the running example database as depicted in Figure 6.1 results in pruning relations *Order* and *Life Style* (as highlighted in red in Figure 6.6). The pruned schema is pictured at the bottom half of Figure 6.6.

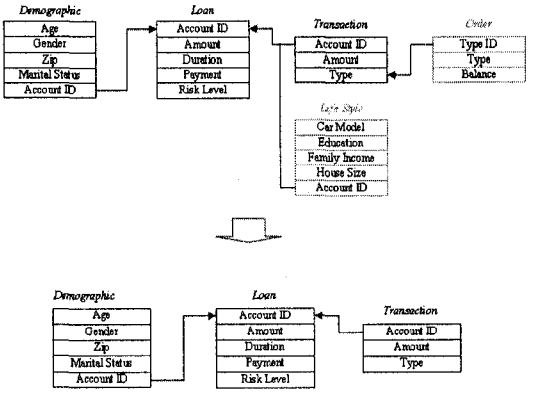


Figure 6.6: Subgraph evaluation and pruning

Algorithm 7 describes the SESP algorithm in detail. In the first step, it converts the relational database into a graph. Secondly, it decomposes this graph into a set of subgraphs. Each is then used to form a subgraph-based relational classifier. Thirdly, the SESP method generates, using subgraph-based relational classifiers, an evaluation data set in which the hypothesis knowledge possessed by a subgraph is represented by its subgraph features. Fourthly, a best first search strategy is employed to select the best subset of subgraph features. Finally, subgraphs are pruned if none of their subgraph features appear in the final subgraph feature subset.

This section discussed the details of the SESP method for pruning a relational database. In the next section we present a performance study of our algorithm.

6.3 Experimental Studies

In our evaluation, we compare the accuracy of a relational classifier constructed from the original schema with the accuracy of one built from a pruned schema. We performed our experiments, against benchmarking databases using MRC, RelAggs [136], TILDE [20], and CrossMine algorithms as the tested learning methods. Recall from Chapter 2 that, the MRC and RelAggs approaches are aggregation-based algorithms. Here the C4.5 decision trees [196] were applied as their single-table learners. In contrast, the CrossMine and TILDE methods are two logic-based strategies.

Also, in order to test our method against databases with large numbers of relations, we conducted experiments against six synthetic databases. In addition, we only consider join paths which contain less than five (5) tables. Recall from Section 5.4.2 that this number was empirically determined. This number provides a good trade off between accuracy and execution time. Also, C4.5 decision tree was applied as the subgraph-based classifiers of the SESP strategy. All experiments were conducted using ten-fold cross validation. We report the average running time of each fold (run on a 3 GHz Pentium4 PC with 1 GByte of RAM).

6.3.1 Benchmarking Databases

In this experiment, we used the same data sets as used in Section 3.4.2, except for the MUT188 and THROM. The MUT188 and THROM data sets consist of very small numbers of relations. They contain only three and five relations, respectively. In other words, we here used the data sets F682AC, F400AC, F234AC, and ECML98. Each of these data sets consists of at least eight relations. Note that, experiments against databases with a very large number of relations will be presented in Section 6.3.2.

Experimental Results and Discussion: The predictive accuracy we obtained, using MRC, RelAggs, TILDE, and CrossMine is presented in Table 6.1. The results

obtained with the respective original and pruned schemas are shown side by side. We also present the compression rates achieved by the SESP approach in the last column of Table 6.1. The compression rate considers the number of relations of the original schema ($N_{original}$) and the number of relations pruned (N_{pruned}) and is calculated as $(N_{original} - N_{pruned})/N_{original}$. In table 6.2, we also provide the execution time of the pruning process, as well as the running time required for the four tested algorithms against the original and pruned schemas.

Table 6.1: Accuracies obtained using methods MRC, RelAggs, TILDE, and CrossMine against the original and pruned schemas, along with the compression rate

Schema	MRC		RelAggs		TILDE		CrossMine		Compress.
	Original	Pruned	Original	Pruned	Original	Pruned	Original	Pruned	Rate
F682AC	93.4 %	93.4 %	92.1 %	92.9 %	88.9 %	88.8%	90.3 %	90.3 %	75.0 %
F400AC	87.3 %	88.0 %	89.0 %	86.8 %	81.0 %	81.0%	85.8 %	87.3 %	50.0 %
F234AC	89.7 %	92.3 %	90.2 %	90.2 %	86.8 %	86.8%	88.0 %	89.4 %	25.0 %
ECML98	87.6 %	87.5 %	88.0 %	86.2 %	53.7 %	52.0%	85.3 %	83.7 %	55.5 %

From Table 6.1, one can see that the SESP algorithm not only reduces the size of the relational schema, but also produces compact pruned schemas that provide comparable multi-relational classification models in terms of the accuracy obtained. The results shown in Table 6.1 provide us with two meaningful observations. The first is that the SESP strategy is capable of pruning the database schemas meaningfully. The number of tables, originally eight, eight, eight, and nine, were pruned to two, four, six, and four for tasks F682AC, F234AC, F400AC and ECML98, respectively. The compression rates for these four learning schemas are 75%, 50%, 25%, and 55.5%, respectively. The second finding is that the pruned schemas produce comparable accuracies, when compared to the results obtained with the original schemas. The comparability was found to be independent of the learning algorithms used. Results as obtained by the aggregation-based methods show that, for three of the four databases (F682AC, F400AC and F234AC), the MRC algorithm

obtained the same or slightly better predictive results when pruned. Only against the ECML98 database, did the pruned MRC algorithm obtain a slightly lower accuracy than the original (lower by only 0.1%). When considering the RelAggs algorithm, the results also convince us that the predictive accuracy produced by the RelAggs method against both the pruned and full schemas was comparable. Against the F234AC and F682AC data sets, the RelAggs algorithm achieved the same or slightly better predictive results. Only against the F400AC and ECML98 data sets, did the RelAggs method yield slightly lower accuracy than the original (lower by 2.2% and 1.8%, respectively).

When testing with the logic-based strategies, Table 6.1 shows that, the TILDE algorithm obtained almost the same accuracy against three of the four tested data sets (F682AC, F400AC and F234AC). Only against the ECML98 database, did the TILDE algorithm obtain a slightly lower accuracy than the original (lower by only 1.7%). When considering the CrossMine method, the accuracy produced by this method against both the pruned and full schemas was also very close. In two cases (F400AC and F234AC) the predictive performance on the pruned schemas outperformed that of the original structures. One exception is the performance with the ECML98 database, where a slight decrease of 1.6% against the full schema was observed.

Table 6.2: Execution time (seconds) required using the four tested methods against the original and pruned schemas, along with the computational time of the SESP method

Schema	MRC		RelAggs		TILDE		CrossMine		Pruning
	Original	Pruned	Original	Pruned	Original	Pruned	Original	Pruned	Time
F682AC	5.59	3.28	89.54	57.10	1051.90	152.22	11.60	8.57	2.91
F400AC	2.83	2.25	60.00	51.83	650.00	132.32	8.10	6.76	1.97
F234AC	1.60	1.17	40.80	34.13	568.30	80.36	5.00	3.41	1.07
ECML98	418.37	220.99	1703.58	1206.39	1108.60	167.76	570.90	366.78	356.24

In terms of computational cost of the SESP method, results presented in Ta-

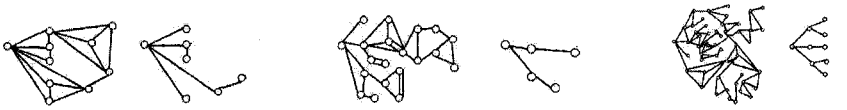
ble 6.2 show that the pruning processes were fast. The fast pruning time is especially relevant when considering the time required when training all four methods against the original schemas. Also, the results indicate that meaningful execution time reductions may be achieved when building the models against the pruned schemas.

In short, these results imply that the SESP strategy can reduces the size of the relational schema, while still maintaining predictive accuracy of the final classification model. Further analysis suggests that the achievement is due to the fact that the SESP algorithm is able to keep the most informative relations and relationships while pruning irrelevant substructures.

6.3.2 Synthetic Databases with Large Numbers of Relations

To further examine the pruning effect of the SESP algorithm, we generated six synthetic databases with large numbers of relations. The aim of these experiments was to further explore the applicability of the SESP algorithm when considering relational domains with a varying large number of relations. We here again used the synthetic database generator as presented in Section 5.4.2. Recall that using this generator, users can specify the expected number of tuples, attributes, relations, and joins, etc., in a database to obtain various kinds of databases. For each database in this paper, we set the expected number of tuples and attributes to 1000 and 15, respectively. The six databases were generated with 10, 20, 50, 80, 100, and 150 relations (denoted as SynR10, SynR20, SynR50, SynR80, SynR100, and SynR150), respectively. In this experiment, we applied the MRC and CrossMine algorithms. As indicated in our empirical results from Section 6.3.1, the MRC and CrossMine strategies achieved good results, in terms of both pruning outcomes achieved and execution time needed.

The relational database structures (before and after the SESP approach) for experiments SynR10, SynR20, SynR50 are provided in Figures 6.7(a), 6.7(b), and



(a) SynR10 (b) SynR20 (c) SynR50

Figure 6.7: The original and pruned structure of databases SynR10, SynR20, and SynR50

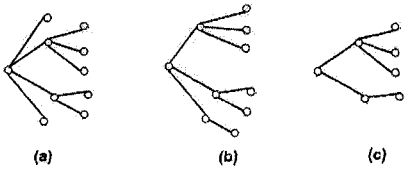


Figure 6.8: The pruned schemas of databases SynR80, SynR100, and SynR150

6.7(c), respectively. The original schemas are on the left hand side and the pruned structures on the right. We also provide the pruned database structures for the tasks SynR80, SynR100, and SynR150 in Figures 6.8. The accuracies obtained (on both original and pruned schemas) by the MRC and CrossMine methods are shown in Figures 6.9 and 6.10, respectively. The compression rates obtained for each of the six databases are provided in Figure 6.11. We also provide the execution time needed when constructing the relational models using the MRC and CrossMine algorithms against the original and pruned schemas in Figure 6.12.

From these Figures one can again see that the SESP algorithm not only reduces the complexity of the structural schemas, but also produces very comparable classification models in terms of the accuracy obtained. The results also show that the accuracies are comparable regardless the relational learning algorithms used. The MRC algorithm, for example, produced equal or higher accuracies for all databases, except for a slight decrease of 0.3% with the SynR80 database. When using the CrossMine method, the results also convince us that the pruned schemas produce comparable classifiers in terms of accuracies obtained. For only one of the databases (SynR100), the CrossMine method noted a loss in accuracy of 2.65%. For the other five databases, the differences noted in predictive performance are all less

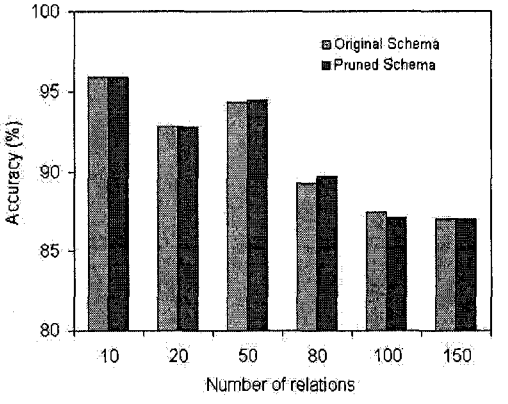


Figure 6.9: Accuracy obtained using the MRC method against the original and pruned schemas of the synthetic databases

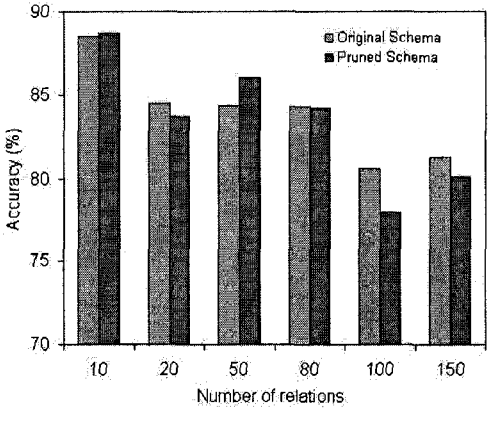


Figure 6.10: Accuracy obtained using the CrossMine method against the original and pruned schemas of the synthetic databases

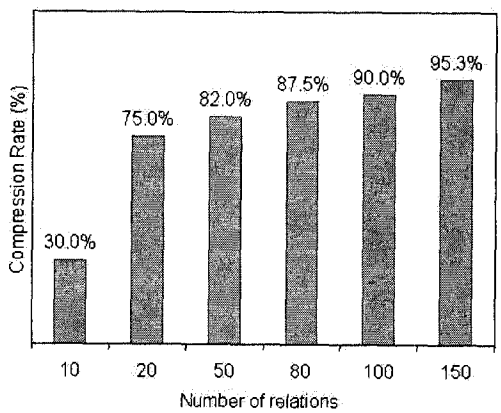


Figure 6.11: Compression rate achieved using the SESP method against the six synthetic databases

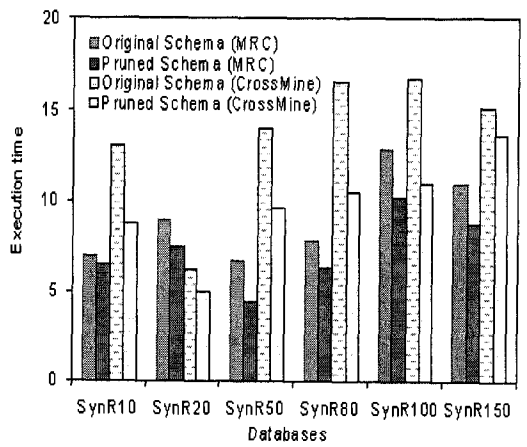


Figure 6.12: Running time required when constructing the relational models using the MRC and CrossMine algorithms against the original and pruned schemas

than 1.5%. In terms of compression rate, the results obtained were quite significant. As shown in Figure 6.11 that the compression rates were more than 80% for databases with more than 50 relations. These results suggest that for complex database schemas, one can use a small part of the whole structure to construct an accurate classifier. In addition, results as presented in Figure 6.12 show that the execution time needed for constructing relational models, using the two tested algorithms was meaningful reduced when pruned.

In summary, the results obtained from the four benchmarking learning problems and the six synthetic databases show that the SESP strategy can reduce the size of the relational schema, while still maintaining predictive accuracy of the final classification model.

6.4 Conclusion

Multirelational data mining application usually involves a large number of relations, where each may come from a different party. Unfortunately, acquiring such data is often expensive, in terms of data mining overheads. Another important issue is preserving the privacy of the data when the relational data is from multiple sources, e.g. from different relations with different levels of access privileges. To date, few attempts have been made to pre-prune relational schemas and reduce the scale of the tasks prior to conducting relational learning.

In this chapter, we present a novel approach, the SESP strategy, which aims to pre-prune uninteresting relations and relationships in order to reduce the scale of relational learning tasks. Our method creates a pruned structure and minimizes the predictive performance loss to the final classification model. The pruned schema is obtained by selecting only those subgraphs which are strongly uncorrelated with each other, but correlated with the target class. All other subgraphs are discarded. Our experiments, on both real-world and synthetic databases, demonstrate that the SESP strategy is able to reduce the size of the relational schema while still maintaining the predictive accuracy of the final classification model. This research

suggests that one can build an accurate relational classification model using only a small subset of the original schema.

This research has application in privacy preservation and utility-based data mining, as follows. If a relation is found to violate privacy restrictions, the SESP algorithm can be used to determine the level of correlation with the other relations. Potentially, privacy breaching relations with little correlation with the target can thus be removed from the learning process, without significant impact on the predictive accuracy. In addition, unnecessary relations can be excluded from the KDD applications, thus significantly cutting the cost of acquiring and maintaining the data.

Consider a practical example. Suppose the Canadian government wants to build a prediction model in order to have an in-depth inside into the quality of life of Canadians. To build such a model, the government may need to buy data from a commercial bank, an insurance company, and a hospital. Subsequently, efforts have to be made to collect and clean the data. Also, serious attention must be paid to preserve the privacy of the data. In such an application, the SESP strategy may fit in as follows. First, certain sample data from the three parties are gathered. Next, the correlation information of this subset is then calculated. Suppose that the SESP method identifies the data from the bank and the data from the insurance company to be highly correlated. It then suggests that the data from the insurance company may be excluded. That is, the government needs to buy data only from the bank and the hospital. In other words, the government does not need purchasing and processing the data from the insurance company, and no longer need to be concerned about exposing any personal data embedded in the insurance data.

The next chapter concludes this thesis and outlines our future work.

CHAPTER 7
CONCLUSION AND FUTURE WORK

Chapter 7

Conclusion and Future Work

授人以鱼不如授人以渔
Give a man a fish, and you feed him for a day; teach him to fish, and you feed him for a lifetime
-Chinese proverb

Relational databases, with vast amounts of data and complex schemas, are ubiquitous in our society. For example, financial establishments, companies, and government institutions such as Bank of America, Google, or Health Canada, to name but a few, are the owners of very large-scale relational databases which needs to be analyzed for trend analysis, fraud detection, customer profiling, privacy-preservation, law enforcement, epidemiological research and so on. This type of data repository introduces new challenges for the data mining community. Such relational repositories involve multiple *interconnected* tables and traditional single-table learning algorithms, therefore, *cannot* be applied directly. Furthermore, the need to explore different combinations of features as *distributed* over multiple tables poses severe efficiency and scalability challenges for many existing multirelational mining algorithms. This study investigated the implicit association between relational database schemas and the application nature of multi-view learning strategy. A multiple view framework, where propositional learning methods can be *directly* deployed for *efficiently* mining multirelational repositories, is presented.

7.1 Summary

In this study, we have investigated the problems that have arisen when mining relational databases and then devised a new solution for multirelational knowledge discovery. In particular, we investigated a novel learning setting, i.e. learning from multiple views for mining multirelational data. We devised the MRC method, which learns from multiple feature sets (views) of a relational database and then integrates the information acquired by the individual view learners to construct a hypothesis model. Also, we extended the multiple views framework to deal with imbalanced multirelational data. We developed the MRC-IM method, which can better predict both the majority and minority classes of imbalanced data. Basing on the MRC framework, we have also studied an alternative view construction strategy and its impact on the multiple views learning framework. In addition, by identifying uninteresting views in the MRC framework, we have devised a novel approach for pre-pruning uninteresting relations of a relational database in order to reduce the learning scale of multirelational classification tasks.

7.2 Contributions

Overall, the work presented here demonstrates four major contributions to the multirelational data mining community.

First, our study is the first to incorporate the idea of learning from multiple views into relational domains. We have illustrated the benefits of using sets of features when dealing with multirelational data, through the development of the MRC, MRC-Cross, MRC-IM, and SESP algorithms. As a result, the idea of learning through multiple views may shed light on the issues of both making good use of the rich feature space presented in structured domains and alleviating the scalability problem when mining multirelational data.

Second, we provide a novel solution to build classification model from multirelational databases. The MRC strategy distinguishes itself from existing mul-

tirelational mining algorithms by excluding the need to *either* transform multiple relations into a universal single table *or* devise new relational learning techniques when it comes to learning algorithms. The MRC method offers both predictive performance and efficiency gains over current relational models, when mining diverse relational databases.

Third, our studies are among the very few research to investigate imbalanced or/and misclassification issues in multirelational data. Our MRC-IM strategy is the first algorithm available for dealing with imbalanced class problems in relational databases. In particular, this strategy achieves superior performance results when the class imbalanced is very high.

Fourth, our research is also among the very few to study pre-pruning substructures for multirelational classification tasks. Our SESP strategy is the first algorithm available for pruning uninteresting relations of multi-relational databases. Our method creates a pruned structure while minimizing predictive performance loss on the final classification model, resulting in reducing the cost of acquiring and managing training data and improving the scalability of the mining process.

Finally, our study here has significance in real-world applications. The MRC and MRC-IM approaches employ knowledge discovery methods as selected from a wide range of existing propositional mining algorithms available. Also, they learn directly from relational databases, while avoiding an extensive propositionalization stage. As a result, the MRC and MRC-IM strategies should proof to be more appropriate for mining a large number of real-world databases than other existing methods. In addition, the SESP strategy can be used to significantly reduce the learning scale of many real world knowledge discovery applications against large scale commercial databases.

7.3 Future Work

There are a number of interesting directions for future work to extend the ideas presented in this thesis. We first discuss further research for the four learning

strategies previously discussed, and then outline farther directions inspired by these methods.

For the MRC and MRC-Cross algorithms, our future work will include thoroughly investigating the performance impact of varying view combination techniques. We also plan to study different "goodness" heuristic measurements and their impact on these algorithms. In addition, the idea of using heterogeneous learners will further increase our understanding of the multiple view learning scheme. For example, we can apply a learning protocol as employed by the Stacking approach, where different types of learning algorithms are trained and then combined. Also, prior work has shown that more complex aggregation functions can improve the generalization accuracy of relational learning [184]. It would also be interesting to investigate this further. Furthermore, we intend to study applying data preprocessing techniques such as feature selection in order to further improve the performance of the two algorithms.

When considering the MRC-IM strategy, our future work will involve evaluating this method against learning tasks with more than two classes. In addition, we intend to further study how the total tuples and imbalanced ratio in each resulting view impacts the result of the final combination model. Furthermore, as discussed in Section 4.4, it would be interesting to integrate the "hyper-ROC" concept into the MRC-IM algorithm and to study upgrading single-table sampling and cost-sensitive learning techniques into relational settings.

Our future work for the SESP algorithm will include extending the subgraph definition. For example, we could allow subgraphs to have more than one slot chain. Also, it would be very interesting to further investigate relational schemas with composite keys. In addition, another area for future work is the study which extends this approach to deal with relational data stored in the form of graph and social network.

In broader ranging future work, we plan to investigate the behavior of the multiple view learning framework, while developing more sophisticated view construction techniques. Recall that, the multi-view strategies presented in this thesis

rely on the fact that a relational database is often designed by domain experts using an ER model, where each different relation usually groups a set of features with very close semantic meaning. An interesting direction for our future work would be to allow the view construction element to search and group features from any relations in the relational domain. In other words, the view construction procedure will search the entire feature space in order to determine how to better group the features into different views. However, the downside of such strategy would be the scalability challenge. This is due to the huge feature space in many relational domains. To mitigate this issue, prior research has shown that heuristic strategies could be applied. For example, a "look-ahead" approach could be employed to decide whether or not a new feature will benefit a specific views. In our opinion, such direction will lead towards a more general multi-view relational data mining method, in which we can expect the final classification model to achieve superior performance.

Another area for future work is employing relational data mining algorithms as hypotheses construction methods, rather than generating relational features and then applying single-table learning strategies, while training a set of diverse individual view learners. In this way, we will have multiple relational mining algorithms, which are trained using different subsets of the entire feature space of the presented data set. That is, we will create an ensemble of relational learning approaches. Research has shown that popular ensemble methods such as Bagging, Boosting, and Stacking can significantly improve the predictive performance of an individual model in some cases. Through employing relational mining algorithms as view learners in the multiple view learning framework, we will be able to explore the impact of popular ensemble techniques on the relational learning strategies.

7.4 Conclusion

In conclusion, this thesis has explored the benefits of learning from multiple views of relational repositories. We have examined the shortcomings of existing multire-

lational learning methods. In addition, we have devised a novel multirelational classification solution which constructs hypotheses from multiple views of a relational database. We also explored different view construction strategies for a given relational database. Furthermore, we have developed a multiple view approach for dealing with skewed class multirelational data. Finally, a pre-pruning strategy which aims to reduce the learning scale of a relational learning problem has also been designed.

In short, this is a first step towards a general multiple view strategy for mining relational domains. Such a general framework will lead towards a better theory and application for efficiently mining data in any relational form, be it databases, social networks, graphs, web linkages, and others.

Appendix A

ER Diagrams and Specifications for Databases Used

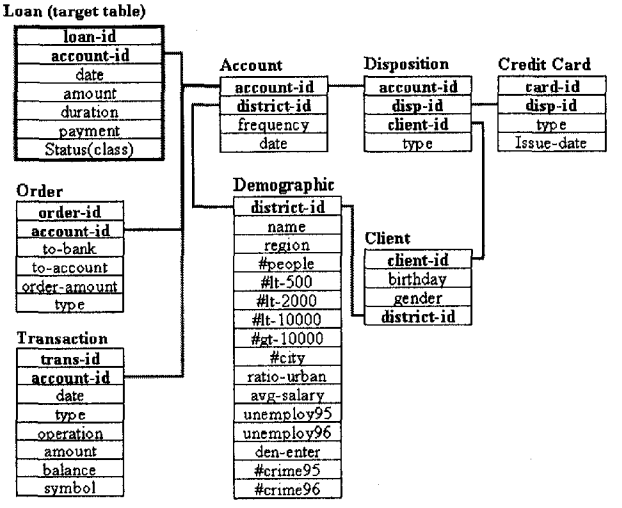


Figure A.1: The FINANCIAL Database

Table A.1: Specification of the FINANCIAL Database

relation name	loan	account	order	transaction	disposition	credit card	demographic	client
# tuples	234(400,682)	4500	6471	52904	5369	892	77	5369
# attributes	6	4	6	8	4	4	16	4

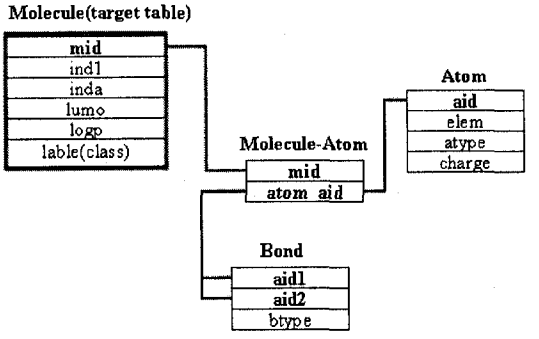


Figure A.2: The MUTAGENISIS Database

Table A.2: Specification of the MUTAGENISIS Database

relation name	molecule	atom	molecule-atom	bond
# tuples	188	4893	4893	5244
# attributes	5	4	2	3

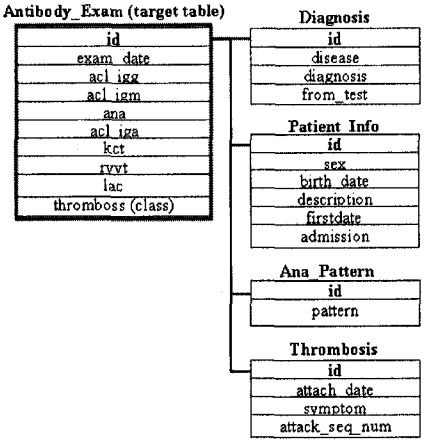


Figure A.3: The THROMBOSIS Database

Table A.3: Specification of the THROMBOSIS Database

relation name	antibody_exam	diagnosis	patient_info	ana_pattern	thrombosis
# tuples	770	1957	1216	644	193
# attributes	9	4	6	2	4

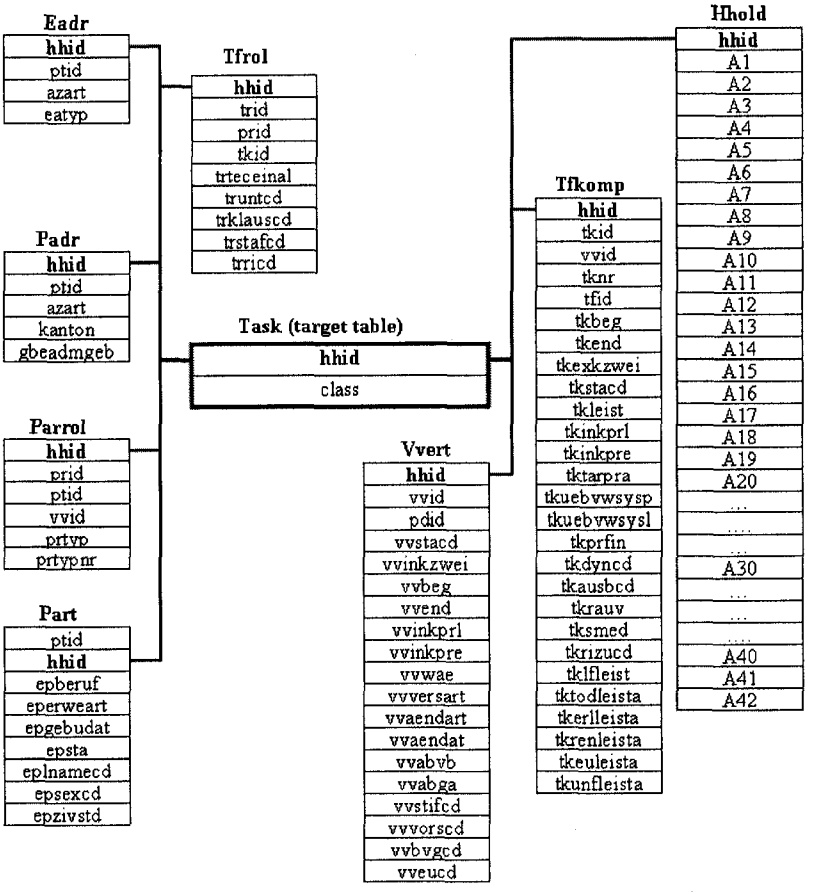


Figure A.4: The WAREHOUSE Database

Table A.4: Specification of the WAREHOUSE Database

relation name	task	eadr	hhold	padr	parrol	part	tfkomp	tfrol	vvert
# tuples	7329	222	7329	7554	62748	7333	43809	40213	20941
# attributes	1	4	43	5	6	9	27	9	19

Appendix B

Publications

REFERRED JOURNAL/CONFERENCE/WORKSHOP PAPERS

- Hongyu Guo and Herna L. Viktor, Multirelational Classification: A Multiple View Approach, *Knowledge and Information Systems: An International Journal*, Springer Publlishers, 2008.

- Hongyu Guo, Herna L. Viktor, and Eric Paquet, Pruning Relations for Substructure Discovery of Multi-relational Databases, *in Proc. 11th European Conference on Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD 2007)*, Warsaw, Poland, September 2007. Pages 462-470.

- Hongyu Guo and Herna L. Viktor, Mining Imbalanced Classes in Multirelational Classification, *in Proc. of the 6th Multi-Relational Data Mining Workshop (PKDD/MRDM'07)*, in conjunction with 11th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'07), Warsaw, Poland, September 2007. Pages 46-57.

- Eric Paquet, Herna L. Viktor, Hongyu Guo, and Isis Pena Sanchez, Constrained Virtual Tailoring from Anthropometric Data, 3-D Shape and Data Mining, *in Proc. of the Second International WEAR Conference (WEAR'07)*, Alberta, Canada, Aug. 2007.

- Hongyu Guo and Herna L. Viktor, Mining Relational Data through Correlation-based Multiple View Validation, *in Proc. the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2006)*, Philadelphia, PA, August 2006. Pages 567-573.

- Herna L. Viktor, Eric Paquet and Hongyu Guo, Measuring to Fit: Virtual Tailoring through Cluster Analysis and Classification, *in Proc. 10th European Conf. on Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD 2006)*, Berlin, Germany, September 2006. Pages 395-406.

- Hongyu Guo and Herna L. Viktor, Multi-view Artificial Neural Networks for Multi-relational Classification, *in Proc. IEEE International Joint Conference on Neural Networks (IJCNN 2006)*, Vancouver, BC, July. 2006. Pages 5259-5266. IEEE Press.

- Hongyu Guo and Herna L. Viktor, Mining Relational Databases with Multi-view Learning, *in Proc. the 4th Multi-Relational Data Mining Workshop (KDD/MRDM'05)*, in conjunction with the Eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2005), Chicago, IL, Aug. 2005. Pages 15-24. ACM Press.

Bibliography

[1] S. Abney. Bootstrapping. In *ACL '02: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 360–367, Morristown, NJ, USA, 2001. Association for Computational Linguistics.

[2] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In J. B. Bocca, M. Jarke, and C. Zaniolo, editors, *VLDB '94: Proceedings of the 20th International Conference on Very Large Data Bases*, pages 487–499. Morgan Kaufmann, 1994.

[3] H. Almuallim and T. G. Dietterich. Learning with many irrelevant features. In *AAAI '91: Proceedings of the 9th National Conference on Artificial Intelligence*, volume 2, pages 547–552, Anaheim, California, 1991. AAAI Press.

[4] H. Almuallim and T. G. Dietterich. Efficient algorithms for identifying relevant features. Technical report, Corvallis, OR, USA, 1992.

[5] C. Anderson, P. Domingos, and D. Weld. Relational markov models and their application to adaptive web navigation. In *KDD '02: Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 143–152. Edmonton, Canada, 2002.

[6] A. Assche, C. Vens, H. Blockeel, and S. Džeroski. First order random forests: Learning relational classifiers with complex aggregates. *Mach. Learn.*, 64(1-3):149–182, 2006.

[7] A. Atramentov, H. Leiva, and V. Honavar. A multi-relational decision tree learning algorithm - implementation and experiments. In *ILP '03: Proceedings of the 13th International Conference on Inductive Logic Programming*, pages 38–56, 2003.

[8] K. Barnard, P. Duygulu, D. Forsyth, N. de Freitas, D. M. Blei, and M. I. Jordan. Matching words and pictures. *J. Mach. Learn. Res.*, 3:1107–1135, 2003.

[9] M. Becker, B. Hachey, B. Alex, and C. Grover. Optimising selective sampling for bootstrapping named entity recognition. In *Proceedings of the ICML-2005 Workshop on Learning with Multiple Views*, Bonn, Germany, 2005.

[10] S. Becker and G. E. Hinton. Self-organizing neural network that discovers surfaces in random-dot stereograms. *Nature*, 355:161 – 163, 1992.

[11] B. Belkhouche and C. Lemus-Olalde. Multiple view analysis of designs. In *Joint Proceedings of the 2nd International Software Architecture Workshop (ISAW-2) and International Workshop on Multiple Perspectives in Software Development (Viewpoints '96) on SIGSOFT '96 workshops*, pages 159–161, New York, NY, USA, 1996. ACM Press.

[12] B. Belkhouche and C. Lemus-Olalde. Multiple views analysis of software designs. *International Journal of Software Engineering and Knowledge Engineering*, 10:557–579, 2000.

[13] P. Berka. Guide to the financial data set. In *A. Siebes and P. Berka, editors, PKDD2000 Discovery Challenge*, 2000.

[14] S. Bickel and T. Scheffer. Multi-view clustering. In *ICDM '04: Proceedings of the 4th IEEE International Conference on Data Mining*, pages 19–26, Washington, DC, USA, 2004.

[15] C. M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, Oxford, UK, 1996.

[16] J. Blatak, P. Turkova, and J. Brezina. Preprocessing and visualization tools for. inductive logic programming. In *ILP '06: 16th International Conference on Inductive Logic Programming*, pages 28–30, 2006.

[17] D. M. Blei and M. I. Jordan. Modeling annotated data. In *SIGIR '03: Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 127–134, New York, NY, USA, 2003. ACM Press.

[18] H. Blockeel. *Top-down induction of first order logical decision trees.* PhD thesis, Department of Computer Science, K.U.Leuven, Leuven, Belgium, 1998.

[19] H. Blockeel and L. D. Raedt. Relational knowledge discovery in databases. In *ILP '96: Selected Papers from the 6th International Workshop on Inductive Logic Programming*, pages 199–211, London, UK, 1997. Springer-Verlag.

[20] H. Blockeel and L. D. Raedt. Top-down induction of first-order logical decision trees. *Artificial Intelligence*, 101(1-2):285–297, 1998.

[21] H. Blockeel, L. D. Raedt, N. Jacobs, and B. Demoen. Scaling up inductive logic programming by learning from interpretations. *Data Mining and Knowledge Discovery*, 3(1):59–93, 1999.

[22] H. Blockeel and M. Sebag. Scalability and efficiency in multi-relational data mining. *SIGKDD Explor. Newsl.*, 5(1):17–30, 2003.

[23] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the Workshop on Computational Learning Theory*, 1998.

[24] J. Bockhorst and I. M. Ong. Foil-d: Efficiently scaling foil for multi-relational data mining of large datasets. In *ILP '04: Proceedings of the 14th International Conference on Inductive Logic Programming*, pages 63–79, 2004.

[25] U. Brefeld, C. Buscher, and T. Scheffer. Multi-view discriminative sequential learning. In *ECML '05: Proceedings of the 16th European Conference on Machine Learning*, pages 60–71, 2005.

[26] U. Brefeld and T. Scheffer. Co-em support vector learning. In *ICML '04: Proceedings of the 21st International Conference on Machine Learning*, page 16, New York, NY, USA, 2004. ACM Press.

[27] L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[28] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[29] R. Bryll, R. Gutierrez Osuna, and F. Quek. Attribute bagging: improving accuracy of classifier ensembles by using random feature subsets. *PR*, 36(6):1291–1302, June 2003.

[30] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[31] L. Cabibbo and R. Torlone. A framework for the investigation of aggregate functions in database queries. In *ICDT '99: Proceedings of the 7th International Conference on Database Theory*, pages 383–397, London, UK, 1999. Springer-Verlag.

[32] C. Cardie and N. Nowe. Improving minority class prediction using case-specific feature weights. In *ICML '97: Proceedings of the 14th International Conference on Machine Learning*, pages 57–65, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc.

[33] P. K. Chan and S. J. Stolfo. Experiments on multistrategy learning by meta-learning. In *CIKM '93: Proceedings of the 2nd International Conference on Information and Knowledge Management*, pages 314–323, New York, NY, USA, 1993. ACM Press.

[34] P. K. Chan and S. J. Stolfo. A comparative evaluation of voting and meta-learning on partitioned data. In *ICML '95: Proceedings of the 12th International Conference on Machine Learning*, pages 90–98, 1995.

[35] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence and Research*, 16:321–357, 2002.

[36] Y. Chen and Y. Yao. Multiview intelligent data analysis based on granular computing. In *Proceedings of IEEE International Conference on Granular Computing (Grc06)*, pages 281–286, 2006.

[37] Y. Chen and Y. Yao. A multiview approach for intelligent data analysis based on data operators. *Inf. Sci.*, 178(1):1–20, 2008.

[38] J. Cheng, C. Hatzis, H. Hayashi, M.-A. Krogel, S. Morishita, D. Page, and J. Sese. KDD Cup 2001 report. *SIGKDD Explorations*, 3(2):47–64, 2002.

[39] M. Chiang and D. Poole. Dynamic predicate construction for learning relational concepts. In *ILP '07: Proceedings of the 12th International Conference on Inductive Logic Programming*, 2007.

[40] P. Clark and T. Niblett. The CN2 induction algorithm. *Mach. Learn.*, 3(4):261–283, 1989.

[41] W. Cohen. Learning to classify English text with ILP methods. In L. De Raedt, editor, *ILP '95: Proceedings of the 5th International Conference on Inductive Logic Programming*, pages 3–24. DEPTCW, 1995.

[42] M. Collins and Y. Singer. Unsupervised models for named entity classification. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, 1999.

[43] D. J. Cook and L. B. Holder. Substructure discovery using minimum description length and background knowledge. *Journal of Artificial Intelligence Research*, 1:231–255, 1994.

[44] D. J. Cook and L. B. Holder. Graph-based data mining. *IEEE Intelligent Systems*, 15(2):32–41, 2000.

[45] C. Cortes and M. Mohri. *AUC Optimization vs. Error Rate Minimization, Advances in Neural Information Processing Systems 16, eds., Sebastian Thrun, Lawrence Saul, and Bernhard Sch olkopf.* MIT Press, Cambridge, MA, 2004.

[46] I. Coursac, N. Duteil, and N. Lucas. PKDD 2001 Discovery Challenge - medical domain. *In: The PKDD Discovery Challenge 2001*, 3(2), 2002.

[47] M. Craven and S. Slattery. Relational learning with statistical predicate invention: better models for hypertext. *Machine Learning*, 43(1/2):97–119, 2001.

[48] C. Curotto, N. Ebecken, and H. Blockeel. *Multi-relational data mining in Microsoft SQL Server 2005, Data Mining VII - Data, Text and Web Mining, and their Business Applications (Zanasi, A. and Brebbia, C.A. and Ebecken, N.F.F., eds.).* 2006.

[49] S. Dasgupta, M. L. Littman, and D. A. McAllester. PAC generalization bounds for co-training. In *Proceedings of NIPS '01, Neural Information Processing Systems*, pages 375–382, 2001.

[50] J. Davis, E. S. Burnside, I. de Castro Dutra, D. Page, and V. S. Costa. An integrated approach to learning bayesian networks of rules. In *ECML '05: Proceedings of the 16th European Conference on Machine Learning*, pages 84–95, 2005.

[51] J. Davis, E. S. Burnside, I. de Castro Dutra, D. Page, R. Ramakrishnan, V. S. Costa, and J. W. Shavlik. View learning for statistical relational learning: With an application to mammography. In *IJCAI '05: Proceedings of the 19th International Joint Conference on Artificial Intelligence*, pages 677–683, 2005.

[52] J. Davis, I. M. O. V. S. Costa, D. Page, , and I. Dutra. Using bayesian classifiers to combine rules. In *Working Notes of the 3rd Workshop on Multi-Relational Data Mining (MRDM-2004) in conjunction with the Tenth ACM SIGKDD International Conference on Knowlege Discovery and Data Mining (KDD-2004)*, Seattle, Washington, USA, 2004.

[53] J. Davis, I. M. Ong, J. Struyf, E. S. Burnside, D. Page, and V. S. Costa. Change of representation for statistical relational learning. In *IJCAI '07: Proceedings of the 21st International Joint Conference on Artificial Intelligence*, pages 2719–2726, 2007.

[54] J. V. Davis and I. S. Dhillon. Estimating the global pagerank of web communities. In *KDD '06: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 116–125, New York, NY, USA, 2006. ACM Press.

[55] V. R. de Sa. Learning classification with unlabeled data. In J. D. Cowan, G. Tesauro, and J. Alspector, editors, *Proceedings of NIPS'93, Neural Information Processing Systems*, pages 112–119, San Francisco, CA, 1993. Morgan Kaufmann Publishers.

[56] V. R. de Sa. Spectral clustering with two views. In *Proceedings of the ICML-2005 Workshop on Learning with Multiple Views*, Bonn, Germany, 2005.

[57] V. R. de Sa and D. H. Ballard. Category learning through multimodality sensing. *Neural Computation*, 10(5):1097–1117, 1998.

[58] J. Dean and M. R. Henzinger. Finding related pages in the world wide web. In *WWW '99: Proceedings of the 8th International Conference on World Wide Web*, pages 1467–1479, New York, NY, USA, 1999. Elsevier North-Holland, Inc.

[59] R. Dechter and J. Pearl. Generalized best-first search strategies and the optimality af A*. *Journal of the ACM*, 32(3):505–536, July 1985.

[60] A. Degenne and M. Forse. *Introducing Social Networks*. Sage Publications Ltd, 1999.

[61] L. Dehaspe and H. Toivonen. Discovery of relational association rules. In S. Dzeroski and N. Lavrac, editors, *Relational Data Mining*, pages 189–212. Springer-Verlag, 2001.

[62] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

[63] U. Dick and K. Kersting. Fisher kernels for relational data. In *ECML '06: Proceedings of the 17th European Conference on Machine Learning*, pages 114–125, 2006.

[64] T. G. Dietterich. Machine-learning research: Four current directions. *The AI Magazine*, 18(4):97–136, 1998.

[65] P. Domingos. Metacost: A general method for making classifiers cost-sensitive. In *KDD '99: Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 155–164, 1999.

[66] P. Domingos and M. J. Pazzani. Beyond independence: Conditions for the optimality of the simple bayesian classifier. In *ICML '96: Proceedings of the 13th International Conference on Machine Learning*, pages 105–112, 1996.

[67] J. Dougherty, R. Kohavi, and M. Sahami. Supervised and unsupervised discretization of continuous features. In *ICML '95: Proceedings of the 12th International Conference on Machine Learning*, pages 194–202, 1995.

[68] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2000.

[69] I. Dutra, D. Page, V. Costa, and J. Shavlik. An empirical evaluation of bagging in inductive logic programming. In *ILP '02: Proceedings of the 12th International Conference on Inductive Logic Programming*, pages 48–65, 2002.

[70] S. Dzeroski and N. Lavrac. *Editors, Relational Data Mining*. Springer, Berlin, 2001.

[71] S. Dzeroski and L. D. Raedt. Multi-relational data mining: an introduction. *SIGKDD Explor. Newsl.*, 5(1):1–16, 2003.

[72] R. Elmasri and S. B. Navathe. *Fundamentals of Database Systems*. CA, USA, 1989.

[73] W. Emde and D. Wettschereck. Relational instance based learning. In L. Saitta, editor, *ICML '96: Proceedings of the 13th International Conference on Machine Learning*, pages 122 – 130. Morgan Kaufmann Publishers, 1996.

[74] érick Alphonse and S. Matwin. Filtering multi-instance problems to reduce dimensionality in relational learning. *J. Intell. Inf. Syst.*, 22(1):23–40, 2004.

[75] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD '96: Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, pages 226–231, 1996.

[76] T. Fawcett. An introduction to ROC analysis. *Pattern Recogn. Lett.*, 27(8):861–874, 2006.

[77] U. Fayyad and K. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *Proceedings of the International Joint Conference on Uncertainty in AI*, pages 1022–1027, 1993.

[78] I. Fischer and T. Meinl. Graph based molecular data mining - an overview. In *IEEE Conference on Systems, Man & Cybernetics*, pages 4578–4582. IEEE, 2004.

[79] P. A. Flach and N. Lachiche. Naive bayesian classification of structured data. *Mach. Learn.*, 57(3):233–269, 2004.

[80] R. Frank, F. Moser, and M. Ester. A method for multi-relational classification using single and multi-feature aggregation functions. In *PKDD '07: Proceedings of the 11th European Conference on Principles of Data Mining and Knowledge Discovery*, 2007.

[81] Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *ICML '96: Proceedings of the 13th International Conference on Machine Learning*, pages 148–156, 1996.

[82] N. Friedman, L. Getoor, D. Koller, and A. Pfeffer. Learning probabilistic relational models. In *IJCAI '99: Proceedings of the 16th International Joint Conference on Artificial Intelligence*, pages 1300–1309, 1999.

[83] H. Garcia-Molina, J. Ullman, and J. Widom. *Database Systems: The Complete Book.* Prentice Hall, 2002.

[84] T. Gärtner. A survey of kernels for structured data. *SIGKDD Explor. Newsl.*, 5(1):49–58, 2003.

[85] T. Gartner, J. W. Lloyd, and P. A. Flach. Kernels and distances for structured data. *Mach. Learn.*, 57(3):205–232, 2004.

[86] L. Getoor. Link mining: a new data mining challenge. *SIGKDD Explor. Newsl.*, 5(1):84–89, 2003.

[87] L. Getoor. Link-based classification. In U. Maulik, L. Holder, and D. Cook, editors, *Advanced Methods for Knowledge Discovery from Complex Data.* Springer-Verlag, 2005.

[88] L. Getoor and C. P. Diehl. Special issue: Link mining. *SIGKDD Explor. Newsl.*, 7(2), December 2005.

[89] L. Getoor, N. Friedman, D. Koller, and B. Taskar. Learning probabilistic models of relational structure. In *ICML '01: Proceedings of the 18th International Conference on Machine Learning*, pages 170–177, 2001.

[90] L. Getoor and B. Taskar. *editors, Statistical Relational Learning.* MIT Press, In press, 2007.

[91] R. Ghani. Combining labeled and unlabeled data for multiclass text categorization. In *ICML '02: Proceedings of the 19th International Conference on Machine Learning*, pages 187–194, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc.

[92] R. Ghani, R. Jones, T. Mitchell, and E. Riloff. Active learning for information extraction with multiple view feature sets. In *ICML '03: Proceedings of the 20th International Conference on Machine Learning*, pages 21–24, 2003.

[93] E. E. Ghiselli. *Theory of Psychological Measurement.* McGrawHill Book Company, 1964.

[94] M. Ginsberg. *Essentials of artificial intelligence.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1994.

[95] C. G. Giraud-Carrier, R. Vilalta, and P. Brazdil. Introduction to the special issue on meta-learning. *Machine Learning*, 54(3):187–193, 2004.

[96] K. Glocer, D. Eads, and J. Theiler. Online feature selection for pixel classification. In *ICML '05: Proceedings of the 22nd International Conference on Machine Learning*, pages 249–256, New York, NY, USA, 2005. ACM Press.

[97] J. Gonzalez, I. Jonyer, L. Holder, and D. JCook. Efficient mining of graph-based data. In *AAAI '00: Proceedings of the 17th National Conference on Artificial Intelligence*, 2000.

[98] H. Guo and H. L. Viktor. Learning from imbalanced data sets with boosting and data generation: the databoost-im approach. *SIGKDD Explor. Newsl.*, 6(1):30–39, 2004.

[99] A. Gupta and S. Dasgupta. Hybrid hierarchical clustering: Forming a tree from multiple views. In *Proceedings of the ICML-2005 Workshop on Learning with Multiple Views*, Bonn, Germany, 2005.

[100] A. Habrard, M. Bernard, and M. Sebban. Detecting irrelevant subtrees to improve probabilistic learning from tree-structured data. *Fundamenta Informaticae: Special Issue on Mining Graphs, Trees and Sequences*, 2005.

[101] M. Hall. Correlation-based feature selection for machine learning, PH.D diss., Waikato University, Hamilton, New Zealand, 1998.

[102] J. Han and M. Kamber. *Data Mining: Concepts and Techniques, 2nd Edition.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005.

[103] D. Hand, H. Mannila, and P. Smyth. *Principles of Data Mining.* MIT Press, 2001.

[104] J. A. Hartigan. *Clustering Algorithms.* John Wiley & Sons. Inc, 1975.

[105] D. Heckerman. A tutorial on learning with bayesian networks. In *Proceedings of the NATO Advanced Study Institute on Learning in Graphical Models*, pages 301–354, Norwell, MA, USA, 1998. Kluwer Academic Publishers.

[106] D. Heckerman, D. Geiger, and D. M. Chickering. Learning bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20(3):197–243, 1995.

[107] S. Hoche and S. Wrobel. Relational learning using constrained confidence-rated boosting. In *ILP '01: Proceedings of the 11th International Conference on Inductive Logic Programming*, pages 51–64, London, UK, 2001. Springer-Verlag.

[108] R. Hogarth. Methods for aggregating opinions. In *H. Jungermann and G. de Zeeuw, editors, Decision Making and Change in Human Affairs.* Dordrecht-Holland, 1977.

[109] R. C. Holte. Very simple classification rules perform well on most commonly used datasets. *Mach. Learn.*, 11(1):63–90, 1993.

[110] W. H. Inmon and W. H. Inmon. *Building the Data Warehouse, 3rd Edition.* John Wiley & Sons, Inc., New York, NY, USA, 2002.

[111] A. Inokuchi, T. Washio, and H. Motoda. An apriori-based algorithm for mining frequent substructures from graph data. In *PKDD '00: Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery*, pages 13–23, London, UK, 2000. Springer-Verlag.

[112] G. James. *Majority Vote Classifiers: Theory and Applications.* PhD thesis, Stanford University, 1998.

[113] N. Japkowicz. Learning from imbalanced data sets: a comparison of various strategies. In *AAAI Workshop on Learning from Imbalanced Data Sets. Tech. rep. WS-00-05, Menlo Park, CA: AAAI Press.*, 2000.

[114] T. Jo and N. Japkowicz. Class imbalances versus small disjuncts. *SIGKDD Explor. Newsl.*, 6(1):40–49, 2004.

[115] G. H. John and P. Langley. Estimating continuous distributions in bayesian classifiers. In *UAI '95: Proceedings of the 11th Conference on Uncertainty in AI*, pages 338–345, 1995.

[116] K. Kailing, H.-P. Kriegel, A. Pryakhin, and M. Schubert. Clustering multi-represented objects with noise. In *PAKDD '04: Advances in Knowledge Discovery and Data Mining, 8th Pacific-Asia Conference*, pages 394–403, 2004.

[117] S. M. Kakade and D. P. Foster. Multi-view regression via canonical correlation analysis. In *COLT' 07: Proceedings of the 20th Annual Conference on Learning Theory*, pages 82–96, 2007.

[118] K. Kersting, L. D. Raedt, and S. Kramer. Interpreting bayesian logic programs. In *L. Getoor and D. Jensen, editors, Working Notes of the AAAI-2000 Workshop on Learning Statistical Models from Relational Data (SRL)*, 2000.

[119] N. S. Ketkar, L. B. Holder, and D. J. Cook. Comparison of graph-based and logic-based multi-relational data mining. *SIGKDD Explor. Newsl.*, 7(2):64–71, 2005.

[120] J.-U. Kietz, R. Zcker, and A. Vaduva. Mining mart: Combining case-based-reasoning and multistrategy learning into a framework for reusing kdd-applications. In *5th International Workshop on Multistrategy Learning (MSL 2000)*, Guimaraes, Portugal, 2000.

[121] K. Kira and L. A. Rendell. A practical approach to feature selection. In *ML92: Proceedings of the 9th International Workshop on Machine Learning*,

pages 249–256, San Francisco, CA, USA, 1992. Morgan Kaufmann Publishers Inc.

[122] S. Kiritchenko and S. Matwin. Email classification with co-training. In *CASCON '01: Proceedings of the 2001 Conference of the Centre for Advanced Studies on Collaborative Research*, page 8. IBM Press, 2001.

[123] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, 1999.

[124] A. J. Knobbe. *Multi-Relational Data Mining.* PhD thesis, University Utrecht, 2004.

[125] A. J. Knobbe, M. de Haas, and A. Siebes. Propositionalisation and aggregates. In *PKDD '01: Proceedings of the 5th European Conference on Principles of Data Mining and Knowledge Discovery*, pages 277–288, London, UK, 2001. Springer-Verlag.

[126] A. J. Knobbe, A. Siebes, and D. van der Wallen. Multi-relational decision tree induction. In *PKDD '99: Proceedings of the 3rd European Conference on Principles of Data Mining and Knowledge Discovery*, pages 378–383, London, UK, 1999. Springer-Verlag.

[127] M. Kockelkorn, A. Lüneburg, and T. Scheffer. Using transduction and multiview learning to answer emails. In *PKDD '03: Proceedings of the 7th European Conference on Principles of Data Mining and Knowledge Discovery*, pages 266–277, 2003.

[128] R. Kohavi. *Wrappers for Performance Enhancement and Oblivious Decision Graphs.* PhD thesis, Stanford University, 1995.

[129] R. Kohavi and G. H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1-2):273–324, 1997.

[130] R. Kohavi, G. H. John, R. Long, D. Manley, and K. Pfleger. MLC++: A machine learning library in C++. In *ICTAI '94: Proceedings of the 6th International Conference on Tools with Artificial Intelligence*, pages 740–743, 1994.

[131] R. Kohavi, P. Langley, and Y. Yun. The utility of feature weighting in nearest-neighbor algorithms. In *ECML '97: Proceedings of the 9th European Conference on Machine Learning*, Prague, Czech Republic, 1997. Springer-Verlag.

[132] R. Kohavi and M. Sahami. Error-based and entropy-based discretization of continuous features. In *KDD '96: Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, pages 114–119, 1996.

[133] D. Koller and M. Sahami. Toward optimal feature selection. In *ICML '96: Proceedings of the 13th International Conference on Machine Learning*, pages 284–292, 1996.

[134] R. I. Kondor and J. D. Lafferty. Diffusion kernels on graphs and other discrete input spaces. In *ICML '02: Proceedings of the 19th International Conference on Machine Learning*, pages 315–322, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc.

[135] I. Kononenko. On biases in estimating multi-valued attributes. In *IJCAI '95: Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pages 1034–1040, 1995.

[136] M.-A. Krogel. *On Propositionalization for Knowledge Discovery in Relational Databases*. PhD thesis, The Faculty of Computer Science, Otto-von-Guericke-Universitt Magdeburg, 2005.

[137] M.-A. Krogel, S. Rawles, F. Zelezny, P. A. Flach, N. Lavrac, and S. Wrobel. Comparative evaluation of approaches to propositionalization. In *ILP '03: Proceedings of the 13th International Conference on Inductive Logic Programming*, pages 197–214, 2003.

[138] M.-A. Krogel and T. Scheffer. Multi-relational learning, text mining, and semi-supervised learning for functional genomics. *Mach. Learn.*, 57(1-2):61–81, 2004.

[139] M.-A. Krogel and S. Wrobel. Transformation-based learning using multirelational aggregation. In *ILP '01: Proceedings of the 11th International Conference on Inductive Logic Programming*, pages 142–155, 2001.

[140] M.-A. Krogel and S. Wrobel. Facets of aggregation approaches to propositionalization. In *Proceedings of the Work-in-Progress Track at the ILP*, 2003.

[141] M. Kubat, R. C. Holte, and S. Matwin. Machine learning for the detection of oil spills in satellite radar images. *Machine Learning*, 30(2-3):195–215, 1998.

[142] R. Kumar, K. Punera, and A. Tomkins. Hierarchical topic segmentation of websites. In *KDD '06: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 257–266, 2006.

[143] M. Kuramochi and G. Karypis. An efficient algorithm for discovering frequent subgraphs. *IEEE Transactions on Knowledge and Data Engineering*, 16(9):1038–1051, 2004.

[144] N. Landwehr, K. Kersting, and L. D. Raedt. Integrating naive bayes and foil. *J. Mach. Learn. Res.*, 8:481–507, 2007.

[145] N. Landwehr, A. Passerini, L. D. Raedt, and P. Frasconi. kfoil: Learning simple relational kernels. In *AAAI '06: Proceedings of the the 21st National Conference on Artificial Intelligence*, 2006.

[146] P. Langley and S. Sage. Induction of selective bayesian classifiers. In *UAI '94: Proceedings of the 10th Annual Conference on Uncertainty in AI)*, pages 399–40, San Francisco, CA, 1994. Morgan Kaufmann.

[147] N. Lavrac and S. Dzeroski. *Inductive Logic Programming: Techniques and Applications*. Routledge, New York, NY, 10001, 1993.

[148] N. Lavrac, F. Zelezny, and P. Flach. Rsd: Relational subgroup discovery through first-order feature construction. In *ILP '02: Proceedings of the 12th International Conference on Inductive Logic Programming*. Springer-Verlag, 2002.

[149] N. Lavrač. *Principles of knowledge acquisition in expert systems*. PhD thesis, Faculty of Technical Sciences, University of Maribor, 1990.

[150] A. Lazarevic and V. Kumar. Feature bagging for outlier detection. In *KDD '05: Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 157–166, 2005.

[151] C. Ling and V. Sheng. *Cost-sensitive Learning and the Class Imbalanced Problem, Encyclopedia of Machine Learning. C. Sammut (Ed.).* Springer, 2007.

[152] C. X. Ling and C. Li. Data mining for direct marketing: Problems and solutions. In *KDD '98: Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining*, pages 73–79, 1998.

[153] C. X. Ling and V. S. Sheng. Test strategies for cost-sensitive decision trees. *IEEE Transactions on Knowledge and Data Engineering*, 18(8):1055–1067, 2006.

[154] H. Liu and R. Setiono. A probabilistic approach to feature selection - a filter solution. In *ICML '96: Proceedings of the 13th International Conference on Machine Learning*, pages 319–327, 1996.

[155] Q. Lu and L. Getoor. Link-based classification. In *ICML '03: Proceedings of the 20th International Conference on Machine Learning*, 2003.

[156] M. Maloof. Learning when data sets are imbalanced and when costs are unequal and unknown. In *ICML Workshop on Learning from Imbalanced Data Sets II*, 2003.

[157] T. Matsuda, T. Horiuchi, H. Motoda, and T. Washio. Extension of graph-based induction for general graph structured data. In *PADKK '00: Proceedings of the 4th Pacific-Asia Conference on Knowledge Discovery and Data Mining, Current Issues and New Applications*, pages 420–431, London, UK, 2000. Springer-Verlag.

[158] S. Merugu, S. Rosset, and C. Perlich. A new multi-view regression approach with an application to customer wallet estimation. In *KDD '06: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 656–661, New York, NY, USA, 2006. ACM Press.

[159] C. J. Merz. Using correspondence analysis to combine classifiers. *Mach. Learn.*, 36(1-2):33–58, 1999.

[160] R. S. Michalski, I. Mozetic, J. Hong, and N. Lavrac. The multi-purpose incremental learning system AQ15 and its testing application to three medical domains. In *AAAI '86: Proceedings of the 5th National Conference on Artificial Intelligence*, pages 1041–1047, 1986.

[161] A. J. Miller. *Subset Selection in Regression.* Chapman and Hall, New York, 1990.

[162] M. Modrzejewski. Feature selection using rough sets theory. In *ECML '93: Proceedings of the European Conference on Machine Learning*, pages 213–226, London, UK, 1993. Springer-Verlag.

[163] S. Muggleton. Inverse Entailment and Progol. *New Generation Computing, Special issue on Inductive Logic Programming*, 13(3-4):245–286, 1995.

[164] S. Muggleton. Stochastic logic programs. In L. De Raedt, editor, *Proceedings of the 5th International Workshop on ILP*, page 29, 1995.

[165] S. Muggleton. *Inductive Logic Programming.* MIT Press, 1999.

[166] S. Muggleton and C. Feng. Efficient induction of logic programs. In *Proceedings of the 1st Conference on Algorithmic Learning Theory*, pages 368–381. Ohmsma, Tokyo, Japan, 1990.

[167] S. Muggleton and J. Firth. Relational rule induction with CPROGO14.4: a tutorial introduction. In S. Dzeroski and N. Lavrac, editors, *Relational Data Mining*, pages 160–187. Springer-Verlag, New York, NY, USA, 2000.

[168] S. Muggleton and L. D. Raedt. Inductive logic programming: Theory and methods. *Journal of Logic Programming*, 19/20:629–679, 1994.

[169] I. Muslea, S. Minton, and C. Knoblock. Adaptive view validation: A first step towards automatic view detection. In *ICML '02: Proceedings of the 19th International Conference on Machine Learning*, pages 443–450, 2002.

[170] I. Muslea, S. Minton, and C. A. Knoblock. Active semi-supervised learning = robust multi-view learning. In *ICML '02: Proceedings of the 19th International Conference on Machine Learning*, pages 435–442, 2002.

[171] I. A. Muslea. *Active learning with multiple views.* PhD thesis, Department of Computer Science, University of Southern California, 2002.

[172] J. Neville and D. Jensen. Iterative classification in relational data. In *AAAI Workshop on Learning Statistical Models from Relational Data,13-20*, 2000.

[173] J. Neville and D. Jensen. Leveraging relational autocorrelation with latent group models. In *MRDM '05: Proceedings of the 4th International Workshop on Multi-relational Mining*, pages 49–55, New York, NY, USA, 2005. ACM Press.

[174] J. Neville and D. Jensen. Bias/variance analysis for relational domains. In *ILP '07: Proceedings of the 12th International Conference on Inductive Logic Programming*, 2007.

[175] J. Neville, D. Jensen, L. Friedland, and M. Hay. Learning relational probability trees. In *KDD '03: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 625–630, New York, NY, USA, 2003. ACM Press.

[176] K. Nigam and R. Ghani. Analyzing the effectiveness and applicability of co-training. In *CIKM '00: Proceedings of the 9th International Conference on Information and Knowledge Management*, pages 86–93, New York, NY, USA, 2000. ACM Press.

[177] N. J. Nilsson. *Artificial intelligence: a new synthesis.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1998.

[178] D. Opitz and R. Maclin. Popular ensemble methods: An empirical study. *Journal of Artificial Intelligence Research*, 11:169–198, 1999.

[179] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

[180] Z. Pawlak. *Rough Sets: Theoretical Aspects of Reasoning about Data.* Kluwer Academic Publishers, Norwell, MA, USA, 1992.

[181] J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988.

[182] C. Perlich and F. Provost. Distribution-based aggregation for relational learning with identifier attributes. *Mach. Learn.*, 62(1-2):65–105, 2006.

[183] C. Perlich, F. Provost, and J. S. Simonoff. Tree induction vs. logistic regression: a learning-curve analysis. *J. Mach. Learn. Res.*, 4:211–255, 2003.

[184] C. Perlich and F. J. Provost. Aggregation-based feature invention and relational concept classes. In *KDD '03: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 167–176, 2003.

[185] B. Pfahringer and G. Holmes. Propositionalization through stochastic discrimination. In *Proceedings of the Work-in-Progress Track at the 13th International Conference on Inductive Logic Programming*, pages 60–68, 2003.

[186] D. Pierce and C. Cardie. Limitations of co-training for natural language learning from large datasets. In *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing (EMNLP-2001)*, 2001.

[187] U. Pompe and I. Kononenko. Naive Bayesian classifier within ILP-R. In L. De Raedt, editor, *Proceedings of the 5th International Workshop on Inductive Logic Programming*, pages 417–436. Department of Computer Science, Katholieke Universiteit Leuven, 1995.

[188] A. Popescul and L. H. Ungar. Statistical relational learning for link prediction. In *IJCAI03 Workshop on Learning Statistical Models from Relational Data*, 2003.

[189] A. Popescul, L. H. Ungar, S. Lawrence, and D. M. Pennock. Towards structural logistic regression: Combining relational and statistical learning. In *Proccedings of the Workshop on Multi-Relational Data Mining (MRDM-2002) at KDD-2002*, pages 130–141, Edmonton, Canada, 2002.

[190] C. Preisach and L. Schmidt-Thieme. Relational ensemble classification. In *ICDM '06: Proceedings of the 6th International Conference on Data Mining*, pages 499–509, Washington, DC, USA, 2006. IEEE Computer Society.

[191] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes in C: the Art of Scientific Computing.* NY, USA, 1988.

[192] F. Provost. Machine learning from imbalanced data sets 101 (extended abstract). In *Proceedings of the AAAI 2000 Workshop on Imbalanced Data Sets.*, 2000.

[193] F. Provost and T. Fawcett. Robust classification for imprecise environments. *Mach. Learn.*, 42(3):203–231, 2001.

[194] Z. Qin, C. X. Ling, and S. Sheng. "Missing is useful': Missing values in cost-sensitive decision trees. *IEEE Transactions on Knowledge and Data Engineering*, 17(12):1689–1693, 2005.

[195] J. R. Quinlan. Induction of decision trees. *Mach. Learn.*, 1(1):81–106, 1986.

[196] J. R. Quinlan. *C4.5: programs for machine learning.* Morgan Kaufmann Publishers Inc., USA, 1993.

[197] J. R. Quinlan. Bagging, Boosting, and C4.5. In *AAAI/IAAI, Vol. 1*, pages 725–730, 1996.

[198] J. R. Quinlan. Boosting first-order learning. In *ALT '96: Proceedings of the 7th International Workshop on Algorithmic Learning Theory*, pages 143–155, London, UK, 1996. Springer-Verlag.

[199] J. R. Quinlan and R. M. Cameron-Jones. Foil: A midterm report. In *ECML '93: Proceedings of the European Conference on Machine Learning*, pages 3–20, 1993.

[200] L. D. Raedt. Attribute-value learning versus inductive logic programming: The missing links (extended abstract). In *ILP '98: Proceedings of the 8th International Workshop on Inductive Logic Programming*, pages 1–8, London, UK, 1998. Springer-Verlag.

[201] L. D. Raedt and S. Dzeroski. First-order jk-clausal theories are PAC-learnable. In S. Wrobel, editor, *ILP '94*, volume 237 of *GMD-Studien*, pages 49–50. Gesellschaft für Mathematik und Datenverarbeitung MBH, 1994.

[202] L. D. Raedt and W. V. Laer. Inductive constraint logic. In *Proceedings of the 6th Conference on Algorithmic Learning Theory*, volume 997. Springer-Verlag, 1995.

[203] R. Ramakrishnan and J. Gehrke. *Database Management Systems.* McGraw-Hill Companies, 2003.

[204] P. Reutemann, B. Pfahringer, and E. Frank. A toolbox for learning from relational data with propositional and multi-instance learners. In *Australian Conference on Artificial Intelligence*, pages 1017–1023, 2004.

[205] M. Richardson and P. Domingos. Mining knowledge-sharing sites for viral marketing. In *KDD '02: Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 61–70, New York, NY, USA, 2002. ACM Press.

[206] M. Richardson and P. Domingos. The Intelligent Surfer: Probabilistic Combination of Link and Content Information in PageRank. In *Advances in Neural Information Processing Systems 14.* MIT Press, 2002.

[207] M. Richardson and P. Domingos. Markov logic networks. *Mach. Learn.*, 62(1-2):107–136, 2006.

[208] J. Rissanen. Modeling by shortest data description. *Automatica*, 14:465–471, 1978.

[209] S. Ruping and T. Scheffer. Workshop on learning with multiple views, Stefan Ruping and Tobias Scheffer, editors. In *Proceedings of the ICML Workshop on Learning with Multiple Views*, 2005.

[210] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach.* Prentice Hall, Englewood Cliffs, New Jersey, 1995.

[211] T. Sato. A statistical learning method for logic programs with distribution semantics. In *ICLP: International Conference on Logic Programming*, pages 715–729, 1995.

[212] J. Scott. *Social Network Analysis: A Handbook.* Newbury Park, CA: Sage Publications, 1991.

[213] P. Sen and L. Getoor. Cost-sensitive learning with conditional markov networks. In *ICML '06: Proceedings of the 23rd International Conference on Machine Learning*, pages 801–808, New York, NY, USA, 2006. ACM Press.

[214] P. Sen and L. Getoor. Link-based classification. Technical Report CS-TR-4858, University of Maryland, February 2007.

[215] V. S. Sheng and C. X. Ling. Thresholding for making classifiers cost-sensitive. In *AAAI '06: The 21st National Conference on Artificial Intelligence*, 2006.

[216] L. Singh, L. Getoor, and L. Licamele. Pruning social networks using structural properties and descriptive attributes. In *ICDM '05: Proceedings of the 5th International Conference on Data Mining*, pages 773–776, 2005.

[217] A. Srinivasan. The Aleph Manual. 2001.

[218] A. Srinivasan, S. H. Muggleton, M. J. E. Sternberg, and R. D. King. Theories for mutagenicity: a study in first-order and feature-based induction. *Artif. Intell.*, 85(1-2):277–299, 1996.

[219] B. Taskar, P. Abbeel, and D. Koller. Discriminative probabilistic models for relational data. In *UAI '02: Proceedings of the 18th Conference on Uncertainty in AI*, pages 485–492, 2002.

[220] K. Ting. Discretization of continuous-valued attributes and instance-based learning. In *Discretization of Continuous-Valued Attributes and Instance-Based Learning, Technical Report No.491, Basser Department of Computer Science, University of Sydney*, 1994.

[221] K. M. Ting and I. H. Witten. Issues in stacked generalization. *J. Artif. Intell. Res. (JAIR)*, 10:271–289, 1999.

[222] S. Tong and D. Koller. Support vector machine active learning with applications to text classification. In *ICML '00: Proceedings of 17th International Conference on Machine Learning*, pages 999–1006, 2000.

[223] C. Vens, A. V. Assche, H. Blockeel, and S. Dzeroski. First order random forests with complex aggregates. In *ILP '04: Proceedings of the 14th International Conference on Inductive Logic Programming*, pages 323–340, 2004.

[224] R. Vilalta, C. Giraud-Carrier, P. Brazdil, and C. Soares. Using meta-learning to support data-mining. *International Journal of Computer Science Applications*, I(1):131–35, 2004.

[225] H. Wang and C. Zaniolo. Using sql to build new aggregates and extenders for object- relational systems. In *VLDB '00: Proceedings of the 26th International Conference on Very Large Data Bases*, pages 166–175, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[226] J. Wang, H.-J. Zeng, Z. Chen, H. Lu, L. Tao, and W.-Y. Ma. Recom: reinforcement clustering of multi-type interrelated data objects. In *SIGIR '03: Proceedings of the 26th Annual International ACM SIGIR Conference*, pages 274–281, 2003.

[227] Y. Wang and M. Kitsuregawa. Link based clustering of Web search results. *Lecture Notes in Computer Science*, 2118:225–236, 2001.

[228] S. Wasserman and K. Faust. *Social Network Analysis: Methods and Applications.* Cambridge: Cambridge University Press, 1994.

[229] G. Weiss and F. Provost. The effect of class distribution on classifier learning. In *Technical Report ML-TR 43, Department of Computer Science, Rutgers University*, 2001.

[230] G. M. Weiss. Mining with rarity: a unifying framework. *SIGKDD Explor. Newsl.*, 6(1):7–19, 2004.

[231] A. P. White and W. Z. Liu. Bias in information-based measures in decision tree induction. *Machine Learning*, 15(3):321–329, 1994.

[232] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations.* Morgan Kaufmann, CA, USA, 2000.

[233] D. H. Wolpert. Stacked generalization. Technical Report LA-UR-90-3460, Los Alamos, NM, 1990.

[234] D. H. Wolpert. Stacked generalization. *Neural Netw.*, 5(2):241–259, 1992.

[235] T. H. Wonnacott and R. J. Wonnacott. *Introductory Statistics.* Wiley, 1977.

[236] X. Yan and J. Han. gspan: Graph-based substructure pattern mining. In *ICDM '02: Proceedings of the 2nd International Conference on Data Mining*, page 721, Washington, DC, USA, 2002. IEEE Computer Society.

[237] X. Yan and J. Han. Closegraph: mining closed frequent graph patterns. In *KDD '03: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 286–295. ACM Press, 2003.

[238] X. Yan, J. Han, and R. Afshar. Clospan: Mining closed sequential patterns in large databases. In *SDM '03: Proceedings of the 3rd SIAM International Conference on Data Mining*, 2003.

[239] X. Yan, P. S. Yu, and J. Han. Substructure similarity search in graph databases. In *SIGMOD '05: Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 766–777, NY, USA, 2005.

[240] Y. Yao. The art of granular computing. In *Proceeding of the International Conference on Rough Sets and Emerging Intelligent Systems Paradigms*, pages 101–112, 2007.

[241] D. Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd Annual Meeting on Association for Computational Linguistics*, pages 189–196, Morristown, NJ, USA, 1995. Association for Computational Linguistics.

[242] X. Yin, J. Han, J. Yang, and P. S. Yu. Crossmine: Efficient classification across multiple database relations. In *ICDE '04: Proceedings of the 20th International Conference on Data Engineering*, Boston, 2004.

[243] X. Yin, J. Han, J. Yang, and P. S. Yu. Efficient classification across multiple database relations: A crossmine approach. *IEEE Transactions on Knowledge and Data Engineering*, 18(6):770–783, 2006.

[244] R. Zajonic. A note on group judgements and group size. *Human Relations*, 15:177–180, 1962.

[245] T. Zhang, A. Popescul, and B. Dom. Linear prediction models with graph regularization for web-page categorization. In *KDD '06: Proceedings of the*

12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 821–826, New York, NY, USA, 2006.

[246] N. Zhong and Y. Yao. Grid based multi-aspect data analysis in a brain informatics portal. In *Proceedings of ICDM'05 Workshop on Foundation of Semantic Oriented Data and Web Mining, Lin, T.Y. and Xie, Y. (Eds.)*, pages 130–137, Houston, USA, 2005.

[247] J. Zhu, Z. Nie, J.-R. Wen, B. Zhang, and W.-Y. Ma. Simultaneous record detection and attribute labeling in web data extraction. In *KDD '06: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 494–503, New York, NY, USA, 2006.

[248] J.-D. Zucker and J.-G. Ganascia. Representation changes for efficient learning in structural domains. In *ICML '96: Proceedings of the 13th International Conference on Machine Learning*, pages 543–551, 1996.