Knowl Inf Syst (2008) 17:287–312
DOI 10.1007/s10115-008-0127-5

REGULAR PAPER

# Multirelational classification: a multiple view approach

**Hongyu Guo · Herna L. Viktor**

Received: 3 August 2007 / Revised: 9 January 2008 / Accepted: 19 January 2008 /
Published online: 26 February 2008


**Abstract** Multirelational classification aims at discovering useful patterns across multiple inter-connected tables (relations) in a relational database. Many traditional learning techniques, however, assume a single table or a flat file as input (the so-called propositional algorithms). Existing multirelational classification approaches either "upgrade" mature propositional learning methods to deal with relational presentation or extensively "flatten" multiple tables into a single flat file, which is then solved by propositional algorithms. This article reports a multiple view strategy—where neither "upgrading" nor "flattening" is required—for mining in relational databases. Our approach learns from multiple views (feature set) of a relational databases, and then integrates the information acquired by individual view learners to construct a final model. Our empirical studies show that the method compares well in comparison with the classifiers induced by the majority of multirelational mining systems, in terms of accuracy obtained and running time needed. The paper explores the implications of this finding for multirelational research and applications. In addition, the method has practical significance: it is appropriate for directly mining many real-world databases.

**Keywords** Multirelational data mining · Classification · Relational database · Multi-view learning · Ensemble

## 1 Introduction

Multirelational Data Mining (MRDM) aims to discover useful patterns across multiple relations (tables) in a relational database. In general, a relational database consists of multiple relations, which are inter-connected by means of foreign key joins. Since their first release in 1970s, relational databases have been routinely used to collect and organize many real-world

H. Guo (✉) · H. L. Viktor
School of Information Technology and Engineering,
University of Ottawa, Ottawa, ON, Canada
e-mail: hguo028@site.uottawa.ca

H. L. Viktor
e-mail: hlviktor@site.uottawa.ca

data—from financial transactions, medical records, to health informatics observation. Thus, mining these data offers a unique opportunity for the data mining community. However, most traditional data mining approaches assume that the data are represented in a single table or a flat file (the so-called propositional methods). As a result, such propositional strategies, such as Decision Trees [59] and Support Vector Machines (SVMs) [10,36], can not exploit knowledge embedded across multiple inter-linked tables. Mining from such structured data, therefore, has become strategic important.

In order to learn from relational data, in the past decade, existing approaches either "upgrade" propositional algorithms to deal with relational presentations by "re-inventing the wheels" or "flatten" multiple relations into a universal flat file, which is then used as the input of propositional learning methods. Unfortunately, existing "upgrading" approaches, especially those using Logic Programming techniques, often suffer not only from poor scalability when dealing with complex database schemas but also from unsatisfactory predictive performance while handling noisy or numeric values in real-world applications. Contrary to those "upgrading" algorithms, "flattening" methods aim to directly use propositional learning algorithms by transforming multiple relations of a relational database into a universal flat file. However, "flattening" strategies tend to require considerable time and effort for the data transformation, result in losing the compact representations of the normalized databases, and produce an extremely large table with huge number of additional attributes and numerous NULL values (missing values). As a result, these difficulties have prevented a wider application of multirelational mining, and post an urgent challenge to the data mining community.

To address the above mentioned problems, this article introduces a multiple view approach—where neither "upgrading" nor "flattening" is required—to bridge the gap between propositional learning algorithms and relational databases. On the one hand, our approach enables traditional data mining methods to utilize information across multiple relations in a relational database. Hence, many efficient and accurate propositional learning algorithms make a wider choice of mining methods available for multirelational data mining applications. On the other hand, the strategy excludes the need to transform multiple inter-connected tables into a universal relation. Therefore, the above mentioned shortcomings resulted from the "flattening" process can be avoided.

Our approach was inspired by a promising new strategy, i.e. multi-view learning. Multi-view learning describes the problem of learning from multiple independent sets of features, i.e. views, of the presented data. This framework has been applied successfully to many real-world applications such as information extraction [8] and face recognition [19]. As in the example given by Blum and Mitchell [8], one can classify segments of televised broadcast based *either* on the video *or* on the audio information. In fact, a multi-view learning problem with $n$ views can be seen as $n$ strongly *uncorrelated feature sets* which are distributed in the multiple relations of a relational database. Often, a relational database is designed by domain experts using an Entity Relationship (ER) model, where multiple relations are connected via entity-relationship links [24,62]. Each relation 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 Fig. 1, from the PKDD 99 discovery challenge [3]. Here a banking database which consists of 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. In other words, each relation from this database provides different types of information, or views, for the concept to be learned, i.e. whether a customer is a *risk* or *not*.

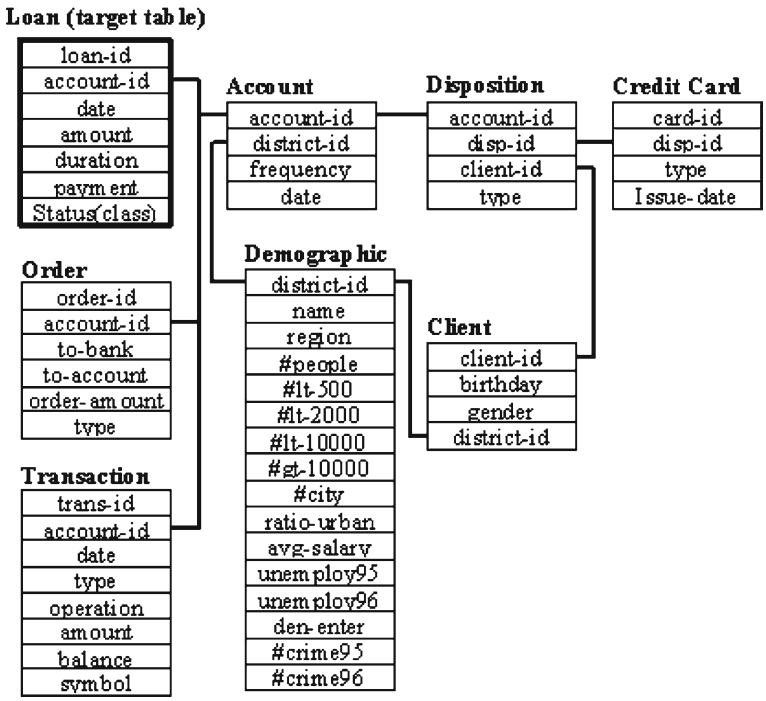


**Fig. 1** A simple database from PKDD 99

This problem is therefore a perfect candidate for learning from multiple views, as will be discussed in detail in Sect. 3.

Using a relational database as input, our so-called Multi-view Relational Classification (MRC) strategy learns from multiple views (feature set) of a relational database, and then information acquired by view learners are integrated to construct a final classification model. Our empirical studies show that our method compares well in comparison with the classifiers induced by the majority of multirelational mining systems, in terms of accuracy obtained and running time needed. The paper explores the implications of this finding for the multirelational research and applications. In addition, the MRC method has practical significance: it is appropriate for directly mining many real-world databases.

This paper is organized as follows. Section 2 introduces the preliminary concepts and related work. Next, in Sects. 3, 4, and 5, we present the MRC strategy. This is followed, in Section 6, by our experimental results. Section 7 concludes the paper.

## 2 Related work

### 2.1 Preliminary concept

Recall that, multirelational data mining aims to discover useful patterns across multiple relations (tables) in a relational database. 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 $T_{.key}$), 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 primary keys of other tables: this link specifies a *join* between two tables, and a foreign key attribute of table $T_2$ referencing table $T_1$ is denoted as $T_2.T_{1\text{-}key}$. Figure 1 gives a simple example of a relational database schema with eight tables. All relationships among tables are represented with bold lines. The bold 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.

Using a relational database as input, multirelational data mining approaches can be initiated to search for patterns. Classification is one of the most commonly addressed tasks within the data mining community [2,25]. 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 target tuple $x$ to the category $Y$:

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

Consider a two-class problem (e.g. positive and negative). The task of a multirelational classification is to search relevant features across different relations (i.e. both from $T_{target}$ and $\{T_i\}_1^n$) to build hypotheses in order to separate the positive and negative tuples in the target relation $T_{target}$.

## 2.2 Previous work

Two main categories of approaches for relational data mining, namely, "upgrading" and "flattening" strategies, have been actively investigated.

### *2.2.1 Upgrading approach*

"Upgrading" methods refer to strategies which "upgrade" conventional learning algorithms to deal with relational presentations. A number of methods, which are considered as first-order upgrades of existing propositional learners, have been proposed in the Inductive Logic Programming (ILP) community [54,61,66]. For example, the popular first-order inductive learner (FOIL) method [60] upgrades the well-known CN2 algorithm [15] to deal with first-order representations. This approach employs a general-to-specific search to build rules to explain many positive examples and cover few negative ones. The top-down induction of logical decision trees (TILDE) method [6] extends the popular C4.5 propositional learner to tackle relational representations. The TILDE algorithm applies logical queries in nodes of the decision tree instead of testing attribute values. The divide-and-conquer nature embedded in the decision tree construction makes the TILDE method an efficient one. The well-known Progol method [51] is considered as an upgrade of the AQ algorithm [50]. Scalability, however, is a big challenge to approaches using ILP [52,76]. 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. [76] present the CrossMine algorithm, an extension

[1] For simplicity, we only consider key attribute as a single attribute here.

of the FOIL approach. Combining with other learning techniques such as the look-one-ahead and sampling strategies, this approach demonstrates high scalability and accuracy.

The first-order upgrade method also attracts significant attention in the statistical relational learning (SRL) community. These approaches usually are proposed to combine the logic theory with probability methodology. Popular ones include Probabilistic Relational Models (PRMs) [26], Relational Markov Networks (RMNs) [68], and Relational Probability Trees (RPTs) [55]. In short, the ILP strategy was historically the first approach to deal with multirelational data and has been extensively studied since then.

Unfortunately, existing "upgrade" approaches, especially those using Logic Programming techniques, however, usually are insufficient to learn from large real-world business databases. ILP techniques tend to be time consuming when building hypotheses. The data sets to be dealt with by the ILP methods are, therefore, typically small [7]. As a result, efficiency and scalability concerns have been consistently raising in ILP-based approaches. In addition, using ILP approaches usually requires some efforts to decide on the right representation for the information (e.g. the training examples and background knowledge) and to choose the right trade-off between expressive power and efficiency (such as the hypothesis language bias). Another major drawback of the ILP-based approaches is that, these methods usually show unsatisfactory performance when handling noisy or numeric-value business data.

### *2.2.2 Flattening approach*

The second important category of strategies used to handle relational domains is the so-called propositionalization method, in which multiple relations are flattened into a universal one. Subsequently, this flattened relation is presented to propositional learners. The LINUS algorithm was first presented in detail in 1990 [49], 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. Following the same lines, propositionalization approaches such as DINUS [48] ("determinate LINUS") and SINUS [45] were designed. Well-known variants of propositionalization method such as the LINUS algorithm and its successors have shown their limitations when dealing with non-determinate relationships. For instance, it is difficult for these methods to handle a one-to-many relationship in a relational database. Recently, Knobbe and Siebes developed the RollUp algorithm to perform the "flattening" of multiple relations in a relational database [40]. The RollUp approach employs a depth-first search (DFS) in the relational database to aggregate information in deeper 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 the aggregate functions found in SQL. This summarization procedure stops if attributes from all relevant tables in the DFS search are aggregated onto the target table. However, the drawbacks of this approach are that, first, the "roll up" process may aggregate the same background table multiple times onto the target relation; second, the search depth of the DFS search has to be given. To overcomes the above-mentioned limitations, more recently, Krogel [44] introduced the so-called RELAGGS propositionalization method. This algorithm shows its superior predictive performance over the RollUp and Dinus algorithms [44]. In the RELAGGS strategy, new features are constructed for the target relation through an aggregation process, where function dependencies in the database are considered.

Unfortunately, severe drawbacks of propositionalization approaches are also noted. Firstly, "flattening" is a non-trivial task: here, extensive preprocessing efforts are needed. In addition, the resulting flat file may contain large amounts of NULL values. More importantly, flattening relational data often results in a very big table with exponentially large numbers

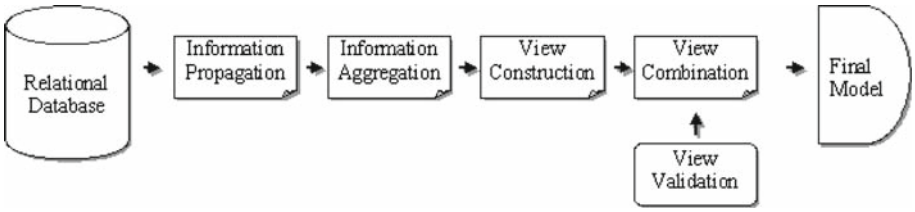


**Fig. 2** The MRC framework

of additional attributes, which causes further scaling challenges and over-fitting problems for propositional algorithms [35]. In fact, the shortcomings from the resulting large flat file mainly attribute to the development of many "upgrading" propositional approaches [22].

### 2.3 Distributed data mining

Our work presented here also relates to learning from distributed databases. In such a learning task, useful patterns are generalized from multiple distributed sources [11,13,14,38,56,64,65,74,75,77]. For example, Zhang et al. [77] presented the Multi-database Mining (MDM) framework to mine from multiple databases. The approach first classifies the multiple databases into different local groups. Next, local patterns are learned from each of these groups. Finally, a weighting process is initiated to synthesize the obtained local patterns. Also, Wu and Zhang [75] described a high performance model to synthesize high-frequency association rules learned from distributed databases. In addition, some key issues concerning distributed data mining strategies such as record linkage [4] and duplicate detection [5] have also been investigated. Record linkage and duplicate detection tackle the problem of determining which database records refer to the same entity.

The next section presents the MRC approach.

## 3 The multiple view approach

The MRC algorithm enables one to classify relational objects by applying conventional data mining methods, while there is no need to flatten multiple relations to a 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 Fig. 2, consists of five sequential stages:

*(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 efficient and various information, which then enables a propositional learner to efficiently 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 were propagated). By applying the basic aggregation

functions 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) Multiple Views Construction Stage:* In the third phase of the MRC algorithm, the *Multiple Views Construction Stage* constructs various hypotheses on the target concept, based on the multiple training data sets given by the *Aggregation Stage*. Conventional 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 differ from one another, are trained.

*(4) View Validation Stage:* All view learners constructed in the *Multiple Views Construction Stage* is then evaluated in the 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.

*(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. Sects. 4 and 5 will discuss related issues of the above five stages.

## 4 Learning from individual views

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. To apply this setting to multirelational classification tasks, consider a target variable $Y$ that 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 in detail next.

### 4.1 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 1** (*Directed Foreign Key Chain Propagation*) Given a database $\Re = \{T_1, \ldots, T_n, T_{target}\}$, where $T_{target}$ is the target table with primary key $T_{target.key}$ and target concept $Y$. Consider a background table $T_i$ , where $i \in \{1, 2, \ldots, n\}$, with an attribute set $A(T_i)$, and a foreign key $T_i.T_{target\text{-}key}$ 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 $T_i.T_{target\text{-}key}$, we say that this tuple is joinable. All joinable tuples in table $T_i$ are selected to form a new relation $T_{i\text{-}new}$. The attributes of table $T_{i\text{-}new}$ are described by the expression $A(T_{i\text{-}new}) = T_{target.key} + Y + A(T_i) - A_{key}(T_i)$, where $A_{key}(T_i)$ are the key attributes of table $T_i$.

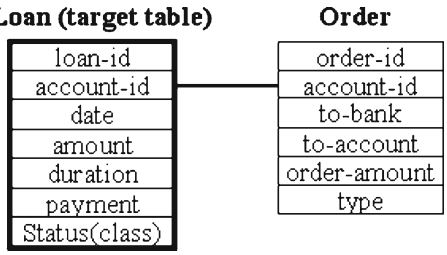


**Fig. 3** A directed foreign key chain

*Example 1* (*Directed Foreign Key Chain Propagation*) Let us use the sample database in Fig. 1 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 Fig. 3. By Definition 1, we know that this relation follows the *directed foreign key chain propagation* scenario. As such, the training data for the view *Order* 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*, i.e. a tuple ID and *status* from the target table, along with the attributes *to-bank*, *to-account*, *account*, and *type* from the *Order* table.

**Definition 2** (*Undirected Foreign Key Chain Propagation*) Given a database $\Re = \{T_1, \ldots, T_n, T_{target}\}$, where $T_{target}$ is the target table with primary key $T_{target.key}$ and target concept $Y$. Consider a background table $T_i$, where $i \in \{1, 2, \ldots, n\}$, with a foreign key $T_i.T_{target\text{-}key}$ referencing the target table $T_{target}$. Next, consider a join path $p = T_a \bowtie T_b \bowtie \cdots, \bowtie T_m$ (where $a, b, \ldots, m \neq i$ and $a, b, \ldots, m \in \{1, 2, \ldots, n\}$) and another background table $T_k$ , where $k \neq i$ and $k \in \{1, 2, \ldots n\}$ with attribute set $A(T_k)$ and foreign key $T_k.T_{i\text{-}key}$ that links to table $T_i$ through the join path $p$ (note that $p$ may consists of zero table). 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 $T_{k\text{-}new}$. We define the set of attributes for table $T_{k\text{-}new}$ as $A(T_{k\text{-}new}) = T_{target.key} + Y + A(T_k) - A_{key}(T_k)$, where $A_{key}(T_k)$ are the key attributes of table $T_k$.

*Example 2* (*Undirected Foreign Key Chain Propagation*) Consider now the situation presented in Fig. 4 which is taken from Fig. 1. For the background *Client* table, the training data from 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 2, 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 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

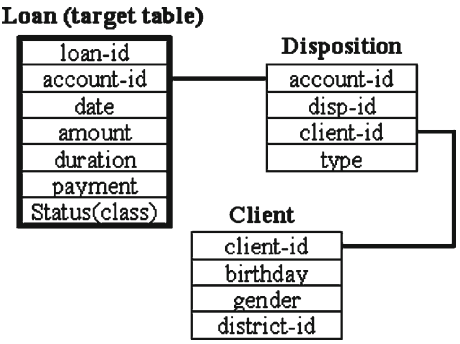


**Fig. 4** An undirected foreign key chain

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 leading to provide less information about the target tuples. Finally, the semantic link with too many joins usually becomes very weak in a relational database [76]. Such weak links, therefore, have less chance to offer sufficient knowledge for learning the target concept. Basing on these observations, the MRC algorithm discourages long join paths.

Two special cases must be given more attention when using the shortest foreign key chains to implement the information propagation process. First, 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. In addition, in some very rare cases, a relation may have more than one foreign key chain with the same lengths and the same number of related target tuples. In these cases, we have empirically determined that the choice of chain makes little difference on the final result.

Note that, Yin et al. presented the *Tuple ID Propagation* strategy in [76]. In their approach, the *numbers* of positive and negative class labels are propagated between different tables. In contrast, we here need to propagate the *original* class labels in order to enable learning from multiple views.

After the propagation, aggregation functions are then applied to the newly generated tables $T_{i\text{-}new}$, to finish constructing training data sets for the view learners. This aggregation procedure is described next in Sect. 4.2.

### 4.2 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 [22,69]. The relations $T_{i\text{-}new}$ obtained in Sect. 4 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 [41,46,57]. To address this problem, the MRC method applies aggregation functions taken from the field of relational databases.

Aggregation has been widely used to summarize the properties of a set of related objects. In the MRC algorithm, the aggregation function squeezes related tuples into a single row using the primary key from the target relation. Each new table $T_{i\text{-}new}$ is aggregated to form a view training set $T_{i\text{-}view}$. Each $T_{i\text{-}view}$ is then used to train separate view learners. For *Nominal* attributes, the MRC method applies the aggregation function $COUNT$. For *Numeric* attributes, the $SUM$, $AVG$, $MIN$, $MAX$, $STDDEV$, and $COUNT$ functions are employed. The aggregation-based feature generation procedure is formally defined as follows. Consider a relation $T_i$ with tuple ID attribute $T_{t.key}$. For each attribute $A_i$ in $T_i$, except the attribute $T_{t.key}$, new attributes are generated to replace the attribute $A_i$ based on the following constraints. If $A_i$ is a Nominal Attribute, the aggregation-based feature generation procedure produces a new attribute $\psi(A_i)$, in which the total number of occurrences of values of attribute $A_i$ is stored. The value is calculated using the $COUNT$ function in the database. If $A_i$ is a Numeric Attribute with a set of values $m$, six new attributes $\psi_{1\ldots6}(A_i)$ are generated. In each of these attributes, the values for $SUM(m)$, $AVG(m)$, $MIN(m)$, $MAX(m)$, $STDDEV(m)$, and $COUNT(m)$ are stored, respectively.

### 4.3 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. 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 Sect. 5.

## 5 Multiple view combination

### 5.1 View validation

The view learners trained in Sect. 4.3 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, strongly uncorrelated view learners are preferred in order to maximize the predictive performance of the combined model. These two issues are discussed next.

#### *5.1.1 View efficiency*

The view validation has to be able to differentiate the strong learners from the weak ones. In other words, it must identify those learners which are only able to learn concepts that are strictly more general or specific than the target one [18]. In the MRC method, we use the error rate in order to evaluate the predictive quality of a view learner. That is, learners with training error greater than 50% are discarded. In other words, only learners with predictive performance better than random guessing are used to construct the final model.

#### *5.1.2 View correlation*

Besides high efficiency, view learners prefer to be as uncorrelated to each other as possible.

The theoretical foundations of multi-view learning are based on the assumptions that the views are *independent* [8]. 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. [54], in real-world problems, one seldom encounters problems with independent views.

Disjoint views are preferred by multi-view learning. Dasupta et al. [18] give Probably Approximately Correct (PAC) bounds for the generalization error of co-training in terms of the agreement rate of hypothesis in two disjoint views. This also justifies Collins and Singer's approach of directly optimizing the agreement rate of learners over different views [16]. Following this line of thought, the MRC algorithm introduces a novel *Correlation-based View Validation (CVV)* algorithm.

The goal of the CVV method is to select a subset of views which are highly correlated with the target concept, but irrelevant to one another. Recall that this is an ideal assumption, since one rarely encounters real world problems with independent views. The CVV strategy uses a heuristic measure to evaluate the correlation between views. A similar heuristic principle has been applied in the test theory by Ghiselli [27] and feature selection approach by Hall [33]. The heuristic "goodness" of a subset of features is formalized in Equation 2 [33]:

$$\mathcal{C} = \frac{K\overline{R_{cf}}}{\sqrt{K + K(K-1)\overline{R_{ff}}}} \tag{2}$$

where $\mathcal{C}$ is the heuristic "goodness" of a selected 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.

To measure the correlations between features and the class, and between features, we adopt the *Symmetrical Uncertainty (U)* [58] to calculate $\overline{R_{cf}}$ and $\overline{R_{ff}}$. This measure is a modified *information gain* (*InfoGain*) measure [59]. It compensates for *InfoGain*'s bias toward attributes with more values. 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

$$\begin{aligned} InfoGain = &-\sum_{y \in Y} p(y)\lg(p(y))) \\ &+ \sum_{x \in X} p(x) \sum_{y \in Y} p(y|x)\lg(p(y|x)) \end{aligned} \tag{3}$$

In order to apply the heuristic "goodness" $\mathcal{C}$ to select a subset of strongly uncorrected views, we need *representatives* of multiple views. In the CVV method, views are represented by *View Features*. *View Features* describe the knowledge, against the target concept, possessed by view learners.

**Definition 3** (View Feature) Let $\{V^1, \ldots, V^n\}$ be $n$ views to be validated. Given a *view validation data set* $\mathcal{T}_v$ with $m$ labels $\{y_1, \ldots, y_m\}$. For each instance $t$ (with label $L$) in

$\mathcal{T}_v$, each view learner is called upon to produce predictions $\{f_{V^i}^{y_k}(t)\}$ ($i \in \{1, \ldots, n\}$ and $k \in \{1, \ldots, m\}$) 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 $\{f_{V^1}^{y_k}(t)\}$ ($k \in \{1, \ldots, m\}$) 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. In this way, a subset of view features can then be selected based on measure $\mathcal{C}$.

After finishing constructing the view feature set, the CVV algorithm subsequently ranks *view feature* subsets according to the correlation-based heuristic evaluation measure $\mathcal{C}$. 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 $\mathcal{C}$.

To search the view feature space, various heuristic approaches such as *hill climbing*, *beam search*, and *best first* are often employed [42,43]. The CVV method uses the best first search strategy [28,63], since this method has demonstrated its successes in many feature selection approaches [33,43,29]. 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 $\mathcal{C}$ 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 usually imposed to a best first search. The CVV algorithm will terminate the search if a number of consecutive non-improvement expansions occur. Based on our experimental observations, we set the number to five heuristically.

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* (*View Validation*) Consider a final view feature subset $\{ f_{V^1}^{y_1}(t), f_{V^1}^{y_2}(t), f_{V^4}^{y_2}(t) \}$. 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 1 shows the CVV method in detail. As presented in Algorithm 1, 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 Eq. (2), 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.

**Algorithm 1** Correlation-based View Validation

**Input:** Candidate view set $\{V_d^1, \ldots, V_d^m\}$,
View Validation Data Set $\mathcal{T}_v$.
**Output:** View set $\{V^1, \ldots, V^k\}$ $(k \leq m)$.

1: Let View Set $V = \emptyset$;
2: Remove candidate views with view learners' accuracy less than 50% from $\{V_d^1, \ldots, V_d^m\}$, 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** $V$.

## 5.2 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 [1,9,12,23,30,70,71,73]. The most popular are those such as bagging [9], boosting [23] and stacking [73]. Voting approaches usually only make sense if the learners perform comparably well [72]. A meta learner method, such as the one used in stacking, is designed to learn which base classifiers are the reliable ones, using another learning algorithm. In the multi-view learning framework, multiple view learners may results in various performances, thus it is hard to guarantee the comparable performances of the view learners. Therefore, the meta-learning method is more suitable to the multi-view learning framework.

**Algorithm 2** MRC Algorithm

**Input:** A DB= $\{T_{target}, T_1, T_2, \ldots, T_n\}$,
View learner $\mathcal{L}$, meta learner $\mathcal{M}$.
**Output:** Classification model $\mathcal{F}$.

1: Propagate and aggregate information, forming candidate view set $\{V_d^1, \ldots, V_d^n\}$;
2: Select a set $\{\mathcal{V}^i\}_1^{n^{'}}$ from $\{V_d^1, \ldots, V_d^n\}$ (Algorithm 1);
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** $\mathcal{F}$.

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.

Algorithm 2 describes the entire steps of the MRC strategy. As shown in Algorithm 2, the MRC method initially propagates and aggregates information to form multiple views from a relational databases. Subsequently, Algorithm 1 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.

## 6 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 [60], CrossMine method [76], TILDE first-order logical trees [6], and RelAggs algorithm [44], along with a "baseline" "flattening" approach (denoted as SimFlat).

Recall from Sect. 2 that, these three logic-based "upgrading" methods and two "flattening" algorithms employ different strategies to deal with structured data. The FOIL algorithm is the best known approach to deal with relational data, and 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 [76]. 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 [6] 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 devise 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 to provide a universal flat file which consists of all features used by individual views in the MRC strategy. As a consequence of such a special flattening, models built on the resulting flat file will provide us a "baseline" performance. In this way, the performance achieved by the MRC algorithm, which uses multiple sets of feature set (here, each feature set corresponds to an individual view), can be compared to performance obtained by the "baseline" approach, which uses all features available in a flat file.

### 6.1 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,

**Table 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 |
| EMCL98 | 7, 329 | 8 | 3705:3624 | 197,478 |
| THROM | 770 | 4 | 695:75 | 4,780 |

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 different number of target tuples but shares the same background relations.

We implemented the MRC algorithm using Weka [72]. Also, our experiments applied C4.5 decision trees [59] and Naive Bayes probabilistic classifiers [37] 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 were chose because of their sensitivity to the changes of the input attributes [20]. The default settings of these two learning methods were used. Each of these experiments produces accuracy results using tenfold 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. In addition, the running time of the RelAggs approach included the time used to convert multiple relations into a flat file. Also, 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.

## 6.2 The data sets used for experimental comparison

### *6.2.1 Mutagenesis database*

Our first experiment (denoted as MUT188) was conducted against the Mutagenesis data set [67]. 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 4,893 atoms and 5,244 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 1. For this database, 3 views, namely *molecule, bond*, and *atom* are constructed by the MRC algorithm.

### 6.2.2 Financial database

Our second experiment was conducted against the Financial database, which was used in the PKDD 1999 discovery challenge [3]. The database was offered by a Czech bank and contains typical business data. The schema of the Financial database is shown in Fig. 1. 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 Fig. 1. 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 [76], 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 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 [76].

### 6.2.3 Thrombosis database

Our next experiment used a database derived from the Thrombosis database used for the PKDD 2001 Discovery Challenge [17]. 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 patients. 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 1 (identified by Throm).

### 6.2.4 ECML98 database

Our last experiment used the database for the ECML 1998 Sisyphus Workshop. This database was extracted from a customer data warehouse of a Swiss insurance company [39]. 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 start schemes prepared in [44]. A summary of the this learning data set is also presented in Table 1 (denoted as ECML98).

**Table 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) |

### 6.3 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.

#### *6.3.1 Experiment #1: Using decision trees as propositional learners and view learners as well as 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 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 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 2, the highest results are highlighted in *bold*. In addition, in the parentheses of this table, we provide the accuracy *gains (denoted by "+")* or *lost (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, where the best results for each data set are also highlighted in *bold*.

#### *6.3.2 Discussion of experiment #1*

The predictive performance results, as presented in Table 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 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

**Table 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 |

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 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) 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 2 and 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.

**Table 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) |

### *6.3.3 Experiment #2: Using Decision Trees as propositional learners and view learners as well as 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 4 and running time needed (in seconds) in Table 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.

### *6.3.4 Discussion of experiment #2*

The experimental results, as presented in Tables 4 and 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 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 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 5, the MRC algorithms with and without the view validation resulted in little difference.

In summary, these results indicate that when using Naive Bayes, which may be 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 that of feature independence [34]. Research has shown that correlations between features degrade the predictive performance of the Naive Bayes methods [21,47]. Our experimental results here imply that,

**Table 5** Time required using C4.5 as view learners and Naive Bayes meta learners (s)

| 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 |

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 [59] 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.

#### *6.3.5 Experiment #3: Using Naive Bayes as propositional learners and view learners as well as 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).

We present the predictive accuracy obtained for each of the six learning tasks in Table 6 and running time needed (in seconds) in Table 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.

#### *6.3.6 Discussion of experiment #3*

Our results, as presented in Table 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 SimFlat 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

**Table 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 7** Time required using Naive Bayes as view learners and C4.5 as meta learners (s)

| 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 |

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 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 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 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 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.

#### *6.3.7 Summary of the three experiments*

In summary, the three experimental results (shown in Tables 2, 3, 4, 5, 6, and 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. In addition, our results also suggest the following findings.

Firstly, 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. In addition, 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.

In addition to the promising predictive and scaling results achieved by the MRC algorithm, the results also suggest that the MRC framework yields robust models, in terms of predictive performance and run time required, regardless of the view learners and meta learners employed.

Another important finding of our experiments is that, the results imply that the cooperation of multiple feature sets can yield significantly predictive performance gains over the method which uses all features available. This observation suggest that the multiple view techniques can enable the use of rich features to benefit the learning in the multirelational domains.

## 7 Conclusions

Vast amounts of real world data—from financial transactions, medical records, to health informatics observation—are routinely collected into and organized in relational databases. These repositories offer unique opportunities to data miners, and thereby precious knowledge to decision makers, if knowledge discovery techniques can efficiently exploit the multiple interconnected relations in databases. Existing approaches either “upgrade” propositional learning methods to deal with multiple interlinked relations or “flattening” multiple tables into a single flat file. Unfortunately, the former strategies often result in problems such as unsatisfactory predictive and scaling performance against large business databases, where noisy and numeric value often presented; the latter approaches usually require extensive

preprocessing of the data before the learning process can start and often result in a big table with large amounts of redundant attributes.

This article reports a multiple view approach—where neither "upgrading" nor "flattening" is required—to bridge the gap between propositional learning algorithms and relations 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 compares well in comparison with the classifiers induced by the majority of multirelational systems in terms of accuracy obtained and running time needed.

This paper here makes two chief contributions to the multirelational data mining community. First, a novel approach, namely, the MRC strategy is devised for multirelational mining. Employing knowledge discovery methods which can be chosen from a wide range of existing propositional mining algorithms, learning directly from relational databases, and excluding the extensively "flattening" preprocessing make the MRC algorithm appropriate for mining useful patterns from many real-world databases. Another contribution is that, the paper here suggests the benefits of incorporating sets of features (views) when dealing with multirelational data. In other words, the adoption of multi-view learning framework may shed light on the issue of making good use of the rich feature space presented in structured domains, where attributes to describe an object are usually large and often highlight from different aspects.

Typically, a relational database is designed by database experts who use an entity relationship diagram. In these diagrams, each entity thus intuitively corresponds to a different concept or view of the problem domain. However, in some cases, two attributes in different tables may be related. That is, the combination of these two seemingly unrelated attributes may provide us with new knowledge about the problem domain. A possible solution would be to join these two tables together as a database view, and treat this view as a background table. Currently, the MRC algorithm does not take this scenario into considerations. Investigating this issue will be part of our future work. In addition, we will also focus our attention on data streams, i.e. handling evolving relational databases. Our future work will also involve testing this method's scalability against very large databases. Also, experimental evaluation on learning tasks with more than two classes will be further investigated. It would also be interesting to examine the influence of feature selection strategies on the multi-view framework.

**Acknowledgements** This paper extends our earlier work, as reported in [31,32].

## References

1. Aggarwal CC (2004) On leveraging user access patterns for topic specific crawling. Data Min Knowl Discov 9(2):123–145
2. Agrawal R, Imielinski T, Swami AN (1993) Database mining: a performance perspective. IEEE Trans Knowl Data Eng 5(6):914–925
3. Berka P (2000) Guide to the financial data set. In: Siebes A, Berka P (eds) PKDD2000 discovery challenge
4. Bilenko M, Kamath B, Mooney RJ (2006) Adaptive blocking: learning to scale up record linkage. In: ICDM '06: Proceedings of the sixth international conference on data mining. Washington, DC, USA, IEEE Computer Society pp. 87–96
5. Bilenko M, Mooney RJ (2003) Adaptive duplicate detection using learnable string similarity measures. In: KDD '03: Proceedings of the ninth ACM SIGKDD international conference on knowledge discovery and data mining. ACM Press, New York, pp. 39–48
6. Blockeel H, Raedt LD (1998) Top-Down Induction of First-Order Logical Decision Trees. Artif Intell 101(1–2):285–297
7. Blockeel H, Raedt LD, Jacobs N, Demoen B (1999) Scaling up inductive logic programming by learning from interpretations. Data Min Knowl Discov 3(1):59–93

8. Blum A, Mitchell T (1998) Combining labeled and unlabeled data with co-training. In: Proceedings of the workshop on computational learning theory
9. Breiman L (1996) Bagging predictors. Mach Learn 24(2):123–140
10. Burges CJC (1998) A tutorial on support vector machines for pattern recognition. Data Mining Knowl Discov 2(2):121–167
11. Chen R, Sivakumar K, Kargupta H (2004) Collective mining of Bayesian networks from distributed heterogeneous data. Knowl Inf Syst 6(2):164–187
12. Cheng J, Sweredoski MJ, Baldi P (2005) Accurate prediction of protein disordered regions by mining protein structure data. Data Min Knowl Discov 11(3):213–222
13. Cheung DW, Ng VT, Fu AW, Fu Y (1996) Efficient mining of association rules in distributed databases. IEEE Trans Knowl Data Eng 8(6):911–922
14. Cho V, Wüthrich B (2002) Distributed mining of classification rules. Knowl Inf Syst 4(1):1–30
15. Clark P, Niblett T (1989) The CN2 induction algorithm. Mach Learn 3(4):261–283
16. Collins M, Singer Y (1999) Unsupervised models for named entity classification. In: Proceedings of the joint SIGDAT Conference on empirical methods in natural language processing and very large corpora
17. Coursac I, Duteil N, Lucas N (2002) PKDD 2001 discovery challenge — medical domain. In: The PKDD discovery challenge 2001, vol 3(2)
18. Dasgupta S, Littman ML, McAllester DA (2001) PAC generalization bounds for co-training. In: NIPS, pp 375–382
19. de Sa VR, Ballard DH (1998) Category learning through multi-modality sensing. Neural Comput 10(5):1097–1117
20. Domingos P (1999) MetaCost: a general method for making classifiers cost-Sensitive. In: KDD'99, pp 155–164
21. Domingos P, Pazzani MJ (1996) Beyond independence: conditions for the optimality of the simple bayesian classifier. In: ICML '96: Proceedings of the 13th international conference on machine learning. pp 105–112
22. Dzeroski S, Raedt LD (2003) Multi-relational data mining: an introduction. SIGKDD Explor Newsl 5(1):1–16
23. Freund Y, Schapire RE (1996) Experiments with a new boosting algorithm. In: International conference on machine learning, pp 148–156
24. Garcia-Molina H, Ullman J, Widom J (2002) Database systems: the complete book. Prentice Hall, Englewood Cliffs
25. Gehrke J, Ramakrishnan R, Ganti V (2000) RainForest — a framework for fast decision tree construction of large datasets. Data Min Knowl Discov 4(2–3):127–162
26. Getoor L, Friedman N, Koller D, Taskar B (2001) Learning probabilistic models of relational structure. In: Proceedings of the 18th international conference on machine learning, pp 170–177
27. Ghiselli EE (1964) Theory of psychological measurement. McGrawHill, New York
28. Ginsberg M (1994) Essentials of artificial intelligence. Kaufmann, San Francisco
29. Glocer K, Eads D, Theiler J (2005) Online feature selection for pixel classification. In: ICML '05: Proceedings of the 22nd international conference on machine learning. ACM Press, New York pp 249–256
30. Guo H, Viktor HL (2004) Learning from imbalanced data sets with boosting and data generation: the DataBoost-IM approach. SIGKDD Explor Newsl 6(1):30–39
31. Guo H, Viktor HL (2005) Mining relational databases with multi-view learning. In: MRDM '05: Proceedings of the 4th International Workshop on Multi-relational Mining. ACM Press, pp 15–24
32. Guo H, Viktor HL (2006) Mining relational data through correlation-based multiple view validation. In: KDD '06. ACM Press, New York, pp 567–573
33. Hall M (1998) Correlation-based feature selection for machine learning. Ph.D dissertation Waikato University
34. Han J, Kamber M (2005) Data mining: concepts and techniques, 2nd edn. Morgan Kaufmann, San Francisco
35. Hulten G, Domingos P, Abe Y (2003) Mining massive relational databases. In: Proceedings of the IJCAI-2003 workshop on learning statistical models from relational data, pp 53–60
36. Joachims T (1999) Support vector machines (Aktuelles Schlagwort). KI 13(4):54–55
37. John GH, Langley P (1995) Estimating continuous distributions in Bayesian classifiers. In: UAI, pp 338–345
38. Kargupta H, Huang W, Sivakumar K, Johnson E (2001) Distributed clustering using collective principal component analysis. Knowl Inf Syst 3(4):422–448
39. Kietz J-U, Zücker R, Vaduva A (2000) MINING MART: Combining case-based-reasoning and multistrategy learning into a framework for reusing KDD-applications. In: 5th Int'l workshop on multistrategy learning (MSL 2000). Guimaraes, Portugal

40. Knobbe AJ (2004) Multi-relational data mining. Ph.D. thesis, University Utrecht
41. Knobbe AJ, de Haas M, Siebes A (2001) Propositionalisation and aggregates. In: PKDD '01: Proceedings of the 5th European conference on principles of data mining and knowledge discovery. Springer, London, pp 277–288
42. Kohavi R (1995) Wrappers for performance enhancement and oblivious decision graphs. Ph.D. thesis, Stanford University
43. Kohavi R, John GH (1997) Wrappers for feature subset selection. Artif Intell 97(1–2):273–324
44. Krogel M-A (2005) On propositionalization for knowledge discovery in relational databases. Ph.D. thesis, Fakultät fuer Informatik, Otto-von-Guericke-Universität Magdeburg
45. Krogel M-A, Rawles S, Zelezny F, Flach PA, Lavrac N, Wrobel S (2003) Comparative evaluation of approaches to propositionalization. In: ILP, pp 197–214
46. Krogel M-A, Wrobel S (2001) Transformation-based learning using multirelational aggregation. In: ILP, pp 142–155
47. Langley P, Sage S (1994) Induction of selective Bayesian classifiers. In: UAI '94: Proceedings of the 10th annual conference on uncertainty in AI). pp 399–40, Morgan Kaufmann, San Francisco
48. Lavrac N, Dzeroski S (1993) Inductive logic programming: techniques and applications. Routledge, New York
49. Lavrač N (1990) Principles of knowledge acquisition in expert systems. Ph.D. thesis, Faculty of Technical Sciences, University of Maribor
50. Michalski RS, Mozetic I, Hong J, Lavrac N (1986) The multi-purpose incremental learning system AQ15 and its testing application to three medical domains. In: AAAI, pp 1041–1047
51. Muggleton S (1995) Inverse entailment and progol. New Generat Comput, Special issue on Inductive Logic Programming 13(3–4):245–286
52. Muggleton S, Feng C (1990) Efficient induction of logic programs. In: Proceedings of the 1st conference on algorithmic learning theory. Ohmsma, Tokyo pp 368–381
53. Muggleton S, Raedt LD (1994) Inductive logic programming: theory and methods. J Log Programm 19/20:629–679
54. Muslea IA (2002) Active learning with multiple views. Ph.D. thesis, Department of Computer Science, University of Southern California
55. Neville J, Jensen D, Friedland L, Hay M (2003) Learning relational probability trees. In: KDD '03. pp 625–630, ACM Press, New York
56. Parthasarathy S, Zaki MJ, Ogihara M, Li W (2001) Parallel data mining for association rules on shared-memory systems. Knowl Inf Syst 3(1):1–29
57. Perlich C, Provost FJ (2003) Aggregation-based feature invention and relational concept classes. In: KDD'03, pp 167–176
58. Press WH, Flannery BP, Teukolsky SA, Vetterling WT (1988) Numerical recipes in C: the art of scientific computing. Cambridge University Press, Cambridge
59. Quinlan JR (1993) C4.5: programs for machine learning. Morgan Kaufmann, San Francisco
60. Quinlan JR, Cameron-Jones RM (1993) FOIL: a midterm report. In: ECML, pp 3–20
61. Raedt LD, Laer WV (1995) Inductive constraint logic. In: Proceedings of the 6th conference on algorithmic learning theory, vol 997. Springer, Heidelberg
62. Ramakrishnan R, Gehrke J (2003) Database management systems. McGraw-Hill, New York
63. Russell S, Norvig P (1995) Artificial Intelligence: a modern approach. Prentice Hall, Englewood Cliffs
64. Sayal M, Scheuermann P (2001) Distributed web log mining using maximal large itemsets. Knowl Inf Syst 3(4):389–404
65. Skillicorn DB, Wang Y (2001) Parallel and sequential algorithms for data mining using inductive logic. Knowl Inf Syst 3(4):405–421
66. Srinivasan A, King RD (1999) Feature construction with inductive logic programming: a study of quantitative predictions of biological activity aided by structural attributes. Data Min Knowl Discov 3(1):37–57
67. Srinivasan A, Muggleton SH, Sternberg MJE, King RD (1996) Theories for mutagenicity: a study in first-order and feature-based induction. Artif Intell 85(1–2):277–299
68. Taskar B, Abbeel P, Koller D (2002) Discriminative probabilistic models for relational data. In: UAI, pp 485–492
69. Vens C, Assche AV, Blockeel H, Dzeroski S (2004) First order random forests with complex aggregates. In: ILP, pp 323–340
70. Webb G, Zheng Z (2004) Multistrategy ensemble learning: reducing error by combining ensemble learning techniques. IEEE Trans Knowl Data Eng 16(8):980–991
71. Webb GI (2000) MultiBoosting: a technique for combining boosting and bagging. Mach Learn 40(2): 159–196

72. Witten IH, Frank E (2000) Data mining: practical machine learning tools and techniques with Java implementations. Morgan Kaufmann, San Francisco
73. Wolpert DH (1992) Stacked generalization. Neural Netw 5(2):241–259
74. Wu X, Zhang C, Zhang S (2005) Database classification for multi-database mining. Inf Syst 30(1):71–88
75. Wu X, Zhang S (2003) Synthesizing high-frequency rules from different data sources. IEEE Trans Knowl Data Eng 15(2):353–367
76. Yin X, Han J, Yang J, Yu PS (2004) CrossMine: efficient classification across multiple database relations. In: ICDE'04, Boston, pp 399–410
77. Zhang S, Wu X, Zhang C (2003) Multi-database mining. IEEE Comput Intell Bull 2(1):5–13

## Author Biographies

**Hongyu Guo** is a Ph.D. candidate in the Department of Computer Science at the University of Ottawa, Ontario, Canada. He graduated from Shanghai Jiao Tong University, China, with a B.Eng. degree in Computer Science, and received his M.Sc. degree from University of Ottawa in 2004. He has been working in the area of data mining since 2002. His research work focuses on the analysis and development of strategies for knowledge discovery from relational data in the form of databases, graphs, and linkages. He has published 11 refereed papers on data mining. He expects to obtain the Ph.D degree in 2008.

**Herna L. Viktor** is an associate professor at the School of IT and Engineering (SITE), University of Ottawa, Canada and the co-director of the Intelligent Data Analysis Lab (IDeAL) at SITE. She holds a Ph. D. in Computer Science from the University of Stellenbosch, which she received in 1999. Her research interests include data mining, machine learning and data warehousing, and their application to anthropometry and health informatics. She has published more than 70 international journal and conference articles and is on a number of international programme committees. Her research is sponsored by the Canadian National Science and Engineering Research council (NSERC), Canada Foundation for Innovation (CFI) and the Ontario Network for Research in e-Commerce (ORNEC).