NRC Publications Archive Archives des publications du CNRC

Reducing the size of databases for multirelational classification : a subgraph-based approach

Guo, Hongyu; Viktor, Herna L.; Paquet, Eric

This publication could be one of several versions: author's original, accepted manuscript or the publisher's version. / La version de cette publication peut être l'une des suivantes : la version prépublication de l'auteur, la version acceptée du manuscrit ou la version de l'éditeur.

For the publisher's version, please access the DOI link below. / Pour consulter la version de l'éditeur, utilisez le lien DOI ci-dessous.

Publisher's version / Version de l'éditeur:

<https://doi.org/10.1007/s10844-012-0229-0>

Journal of Intelligent Information Systems, November 2012, 2012-11-29

NRC Publications Record / Notice d'Archives des publications de CNRC:

<https://nrc-publications.canada.ca/eng/view/object/?id=490f5160-072f-428f-b1f7-1ba0011ead5e>

<https://publications-cnrc.canada.ca/fra/voir/objet/?id=490f5160-072f-428f-b1f7-1ba0011ead5e>

Access and use of this website and the material on it are subject to the Terms and Conditions set forth at

<https://nrc-publications.canada.ca/eng/copyright>

READ THESE TERMS AND CONDITIONS CAREFULLY BEFORE USING THIS WEBSITE.

L'accès à ce site Web et l'utilisation de son contenu sont assujettis aux conditions présentées dans le site

<https://publications-cnrc.canada.ca/fra/droits>

LISEZ CES CONDITIONS ATTENTIVEMENT AVANT D'UTILISER CE SITE WEB.

Questions? Contact the NRC Publications Archive team at

PublicationsArchive-ArchivesPublications@nrc-cnrc.gc.ca. If you wish to email the authors directly, please see the first page of the publication for their contact information.

Vous avez des questions? Nous pouvons vous aider. Pour communiquer directement avec un auteur, consultez la première page de la revue dans laquelle son article a été publié afin de trouver ses coordonnées. Si vous n'arrivez pas à les repérer, communiquez avec nous à PublicationsArchive-ArchivesPublications@nrc-cnrc.gc.ca.

Reducing the Size of Databases for Multirelational Classification: a Subgraph-based Approach

Hongyu Guo · Herna L. Viktor · Eric Paquet

Received: date / Accepted: date

Abstract Multirelational classification aims to discover patterns across multiple interlinked tables (relations) in a relational database. In many large organizations, such a database often spans numerous departments and/or subdivisions, which are involved in different aspects of the enterprise such as customer profiling, fraud detection, inventory management, financial management, and so on. When considering classification, different phases of the knowledge discovery process are affected by economic utility. For instance, in the data preprocessing process, one must consider the cost associated with acquiring, cleaning, and transforming large volumes of data. When training and testing the data mining models, one has to consider the impact of the data size on the running time of the learning algorithm. In order to address these utility-based issues, the paper presents an approach to create a pruned database for multirelational classification, while minimizing predictive performance loss on the final model. Our method identifies a set of strongly uncorrelated subgraphs from the original database schema, to use for training, and discards all others. The experiments performed show that our strategy is able to, without sacrificing predictive accuracy, significantly reduce the size of the databases, in terms of the number of relations, tuples, and attributes. The approach prunes the sizes of databases by as much as 94%. Such reduction also

Hongyu Guo
National Research Council of Canada,
1200 Montreal Road, Ottawa, ON., K1A 0R6, Canada
E-mail: hongyu.guo@nrc-cnrc.gc.ca

Herna L. Viktor
School of Electrical Engineering and Computer Science, University of Ottawa, Ontario, Canada
800 King Edward Avenue, Ottawa, ON., K1N 6N5, Canada
E-mail: hlviktor@site.uottawa.ca

Eric Paquet
School of Electrical Engineering and Computer Science, University of Ottawa, Ontario, Canada
National Research Council of Canada
1200 Montreal Road, Ottawa, ON., K1A 0R6, Canada
E-mail: eric.paquet@nrc-cnrc.gc.ca

results in decreasing computational cost of the learning process. The method improves the multirelational learning algorithms' execution time by as much as 80%. In particular, our results demonstrate that one may build an accurate model with only a small subset of the provided database.

Keywords Multi-relational Classification · Relational Data Mining

1 Introduction

Multirelational classification, which aims to discover patterns across multiple interlinked tables (relations) in a relational database, poses a unique opportunity for the data mining community (Quinlan and Cameron-Jones, 1993; Zhong and Ohsuga, 1995; Dehaspe et al, 1998; Blockeel and Raedt, 1998; Dzeroski and Lavrac, 2001; Jensen et al, 2002; Jamil, 2002; Han and Kamber, 2005; Krogel, 2005; Burnside et al, 2005; Ceci and Appice, 2006; Yin et al, 2006; Frank et al, 2007; Getoor and Taskar, 2007; Bhattacharya and Getoor, 2007; Landwehr et al, 2007; Rückert and Kramer, 2008; De Raedt, 2008; Chen et al, 2009; Landwehr et al, 2010; Guo et al, 2011). Such relational databases are currently one of the most popular types of relational data repositories. A relational database $\mathfrak{R}$ is described by a set of tables $\{R_1, \dots, R_n\}$. Each table R_i consists of a set of tuples T_{R_i} , a primary key, and a set of foreign keys. Foreign key attributes link to primary keys of other tables. This type of linkage defines a *join* (relationship) between the two tables involved. A set of joins with n tables $R_1 \bowtie \dots \bowtie R_n$ describes a join path, where the *length* of it is defined as the *number of joins* it contains.

A multirelational classification task involves a relational database $\mathfrak{R}$ which consists of a target relation R_t , a set of background relations $\{R_b\}$, and a set of joins $\{J\}$. Each tuple in this target relation, i.e. $x \in T_{R_t}$, is associated with a class label which belongs to Y (target classes). Typically, the task here is to find a function $F(x)$ which maps each tuple x from the target table R_t to the category Y . That is,

$$Y = F(x, R_t, \{R_b\}, \{J\}), x \in T_{R_t}$$

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 R_t and $\{R_b\}$, to separate the positive and negative tuples of the target relation R_t .

In practice, such a relational database in many large organizations spans numerous departments and/or subdivisions. As the complexity of the structured schema increases, it becomes computationally prohibitive to train and maintain the relational model. Furthermore, knowledge discovery processes are affected by economic utility, such as cost associated with acquiring the training data, cleaning the data, transforming the data, and managing the relational databases. These problems, fortunately, may be mitigated by pruning uninteresting relations and tuples before constructing a classification model.

Consider the following simple example task. Suppose that a bank is interested in using the example database from Figure 1 to predict a new customer's risk level for a *personal loan*. This database will be used as a running example throughout the paper. The database consists of five tables and four relationships. Tables *Demographic*, *Transaction*, *Life Style* and *Order* are the background relations and

Fig. 1 A simple sample database

Loan is the target relation. Multirelational 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 may be avoided; 2) the information held by the *Order* and *Life Style* tables is not used when training and deploying the model; 3) the hypothesis search space is reduced, resulting in reduction of the running time of the learning algorithm.

This paper presents a new subgraph-based strategy, the so-called SESP method, for pre-pruning relational databases. The approach aims to create a pruned relational schema that models only the most informative substructures, while maintaining satisfactory predictive performance. The SESP approach initially *decomposes* the relational domain into subgraphs. From this set, it subsequently identifies a subset of subgraphs which are strongly uncorrelated with one another, but correlated with the target class. All other subgraphs are discarded. We compare the classifiers constructed from the original schema with those constructed from the pruned database. The experiments performed, against both real-world and synthetic databases, show that our strategy is able to, without sacrificing predictive accuracy, significantly reduce the size of the databases, in terms of the number of relations, tuples, and attributes, for multirelational classification. The approach prunes the sizes of databases by as much as 94%. Such reduction also results in improving the computational cost of the learning process. The method decreases the multirelational learning algorithms' execution time by as much as 80%. In particular, our results demonstrate that one may build an accurate classification model with only a small subset of the provided database.

The paper is organized as follows. Section 2 introduces the related work. Next, a detailed discussion of the SESP algorithm is provided in Section 3. Section 4

presents a comparative evaluation of the SESP approach. Section 5 concludes the paper.

2 Related Work

Pre-pruning relational data has been a very promising research topic (Bringmann and Zimmermann, 2009). Cohen (1995) 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 relevant.

Following the same trend, Alphonse and Matwin (2004) presented a literal pruning approach for ILP learning problems. The method first approximates the ILP problem with a bounded multi-instance task (Zucker and Ganascia, 1996). 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.

In addition, Singh et al. proposed a method to pre-prune social networks (consist of nodes and edges) using both structural properties and descriptive attributes, as described in (Singh et al, 2005). Their algorithm selects nodes according to the number of connections they have. The nodes deemed not to have enough connections are removed. Edges are also pruned based on their descriptive attribute values.

Following the same line of research, Habrard et al (2005) 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, in contrast, address multirelational classification tasks. We propose a subgraph-based strategy for pre-pruning relational databases in order to improve multirelational classification learning. 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 may predict or imply one another, and provide similar information. Substructures which are weakly correlated to the class are thus said to be of low relevance, because they contain little useful information in terms of helping predicting the class.

In addition to research in the data mining and machine learning fields, sampling in the relational database community, has been an active research area for decades. Many of these works mostly target the efficiency and accuracy of database operations through statistical techniques (Olken and Rotem, 1986; Lipton et al, 1993), or aim at semantic representation and understanding of a database (De Marchi and Petit, 2007). The latter research line is more related to our work because approaches in this field also aim to obtain a compact subschema. For example, the semantic sampling technique introduced by De Marchi and Petit (2007) may

significantly reduce the size of the provided database, in terms of the number of tuples. In detail, their algorithm selects a smaller subset of the original database. The resulted subset aims to have the same functional and inclusion dependencies as the one present in the original database. Aiming at database understanding and reverse engineering, their algorithm intends to prune tuples while maintaining the dependencies of the database. On the contrary, we aim to prune relations, which are often the major concerns for utility-based learning applications. That is, their method is reluctant to prune tables and foreign keys in the database. Their objective is to use a smaller number of tuples to reflect the same linkages and dependencies of the original database. In particular, the resulted subschema needs to maintain the same integration as that of the original database. On the other hand, we mainly seek to remove irrelevant tables, unnecessary foreign keys, and redundant dependencies in the database, based on their relationships with the given target variable, regardless of the interplays in between non-target variables. That is, even a set of relations contain strongly semantic dependencies, our method may still prune them if they are less irrelevant to the learning of the target concept defined by the target variable.

This paper builds on our earlier work as presented in (Guo et al, 2007). In comparison with the earlier paper, this manuscript contains additional material, expanded experimental results, and new insights into our studies. Specifically, we offer new observations on how databases are pruned for multirelational classification. In addition, we present new experiments against additional database as well as additional experimental results from the previous paper to highlight our new observations.

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

3 The SESP Pruning Approach

The core goal of the SESP method is to identify a small set of strongly uncorrelated subgraphs, given a database schema. As presented in Algorithm 1, the process consists of two key steps: *subgraph construction* and *subgraph evaluation*.

The *subgraph construction* process initially converts the relational database schema into an undirected graph (the directions of the foreign key joins will not impact our model here), using the tables as the nodes and joins as edges. Subsequently, the graph is traversed in order to extract unique subgraphs. In this way, the original database is decomposed into different subgraphs. Next, the *subgraph evaluation* element calculates the correlation scores between these extracted subgraphs. Accordingly, it identifies a subset of those subgraphs which are strongly uncorrelated to one another. Each of these steps is discussed next.

3.1 Subgraph Construction

Using a relational database as input, the *subgraph construction* process aims to construct a number of subgraphs, each corresponding to a unique join path of the provided database. By doing so, this set of subgraphs have two essential characteristics. First, each of such subgraphs describes a unique set of related relations of the relational database. Consequently, each subgraph contains different

Algorithm 1 The SESP Approach

Input: a relational database $\mathbb{R} = (R_t, \{R_b\}, \{J\})$; R_t is the target relation with m classes
Output: a pruned database $\mathbb{R}' = (R_t, \{R'_b\}, \{J'\})$

```

1: divide data in  $\mathbb{R}$  into training set  $\mathcal{T}_t$  and evaluation set  $\mathcal{T}_e$ 
2: convert schema  $\mathbb{R}$  into undirected graph  $\mathcal{G}(\mathcal{V}, \mathcal{E})$ , there  $R_t$  and  $R_b$  as nodes  $\mathcal{V}$  and joins  $J$ 
   as edges  $\mathcal{E}$ 
3: call Algorithm 2, provided  $\mathcal{G} \Rightarrow$  subgraphs set  $\{\mathcal{G}_{s_1}, \dots, \mathcal{G}_{s_n}\}$ 
4: for each  $\mathcal{G}_{s_i} \in \{\mathcal{G}_{s_1}, \dots, \mathcal{G}_{s_n}\}$  do
5:   construct classifier  $F_i$ , using  $\mathcal{T}_t$ 
6: end for
7: for each instance  $t \in \mathcal{T}_e$  do
8:   for each  $F_i \in \{F_1, \dots, F_n\}$  do
9:     apply  $F_i \Rightarrow$  creating SubInfo variables  $\{V_i^k(t)\}_{k=1}^m$ ;
10:    forming new data set  $\mathcal{T}'_e$ ;
11:    each instance  $t' \in \mathcal{T}'_e$  is described by  $\mathcal{A} = \{\{V_1^k(t')\}_{k=1}^m, \dots, \{V_n^k(t')\}_{k=1}^m\}$ 
12:   end for
13: end for
14: select  $\mathcal{A}'$  ( $\mathcal{A}' \subseteq \mathcal{A}$ )
15: subgraph set  $S \leftarrow \emptyset$ 
16: for each  $F_i$  do
17:   if  $\exists v, (v \in \{V_i^k(t')\}_{k=1}^m) \wedge (v \in \mathcal{A}')$  then
18:      $S.add(\mathcal{G}_{s_i})$ 
19:   end if
20: end for
21: remove duplicate nodes and joins from  $\{S\}$ 
22: forming  $\mathbb{R}' = (R_t, \{R'_b\}, \{J'\})$ 
23: RETURN  $\mathbb{R}'$ 

```

subset of information embedded in the original database provided. For example, two of the join paths¹ in the example database (as described in Figure 1), e.g. $Loan \bowtie Transaction \bowtie LifeStyle$ and $Loan \bowtie Transaction \bowtie Order$ describes two sets of information concerning the target classes of the original database. Second, the sequence of joins in a subgraph contains exactly the relations and joins that a background table must follow to calculate information with respect to the target classes. That is, each of such subgraphs describes a *very compact* linkage between a background relation and the target relation.

Two heuristic constraints are imposed on each constructed subgraph. The first is that each subgraph must start at the target relation. This constraint ensures that each subgraph will contain the target relation and, therefore, be able to calculate how much essential information possessed by a particular subgraph with respect to the target classes (details to be discussed in Section 3.2). The second constraint is for relations to be unique for each candidate subgraph. The intuition behind this strategy may be described as follows. Typically in a relational domain, the number of possible join paths given a large number of relations is usually *very* large, making it too costly to exhaustively search all join paths (Hamill and Martin, 2004). For example, consider a database that contains n tables. Suppose that all of the n tables contain at least one foreign key linking this table to the other tables in the database. In this case, the number of foreign key paths with length n will be $n!$. Also, join paths with many relations may decrease the number of entities related

¹ Further discussions regarding all the resulted join paths for this database will be presented in Example 1 in this section

Algorithm 2 Subgraph Construction

Input: Graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, Target relation $\mathcal{R}_t$ and background relations $\{\mathcal{R}_{b_1}, \dots, \mathcal{R}_{b_n}\}$, Maximum length of join path allowed $MaxJ$.
Output: Subgraph set $\{\mathcal{G}_{s_1}, \dots, \mathcal{G}_{s_n}\}$.

```

1: Let  $\mathcal{G}_s.add(\mathcal{R}_t)$ , current length of join  $\ell = 0$ , current subgraph set  $\mathcal{W} = \{\mathcal{R}_t\}$ ;
2: repeat
3:    $\ell++$ ; Current Search Subgraph Set  $\mathcal{S} = \mathcal{W}$ ;
4:    $\mathcal{W} = \emptyset$ ;
5:   for each subgraph  $s \in \mathcal{S}$ ,  $s = \{R_t \times \dots \times R_{b_k}\}$  do
6:     for each edge  $e$  in  $\mathcal{E}$  do
7:       if  $e = \{R_{b_k} \times R_{b_n}\}$  and  $R_{b_n} \notin s$  then
8:         append  $e$  to  $s$ , add  $s$  to subgraph  $\mathcal{W}$ 
9:       end if
10:    end for
11:  end for
12:  for each  $w \in \mathcal{W}$  do
13:     $\mathcal{G}_s.add(w)$ ,
14:  end for
15: until  $\ell \leq MaxJ$  or  $\mathcal{W} = \emptyset$ 
16: RETURN  $\mathcal{G}_s$ 

```

to the target tuples. Therefore, we propose this restriction for the SESP algorithm as a tradeoff between accuracy and efficiency (further discussion will be presented in Section 4.2.2). In fact, this heuristic also helps avoid cycles in a join path.

Using these constraints, the subgraph construction process, as described in Algorithm 2, proceeds initially by finding unique join paths with two relations. These join paths are progressively lengthened, one relation at a time. It keeps collecting join paths with one more relation. We use the length of the join path as the stopping criterion, preferring subgraphs with shorter length. The reason for preferring shorter subgraphs is that semantic links with too many joins are usually very weak in a relational database (Yin et al, 2006). Thus we specify a maximum length for join paths. When this number is reached, the entire join path extraction process stops. Note that a special subgraph, one that is comprised solely of the target relation, is created as well.

Example 1: Figure 2 shows all six (6) subgraphs (from $\mathcal{G}_{s_1}$ to $\mathcal{G}_{s_6}$) constructed from the sample database (in Figure 1). In this figure, $\mathcal{G}_{s_1}$ depicts the subgraph which consists of only the target relation. $\mathcal{G}_{s_2}$, $\mathcal{G}_{s_3}$, and $\mathcal{G}_{s_4}$ describe subgraphs with two involved relations. Subgraphs containing three tables are shown in $\mathcal{G}_{s_5}$ and $\mathcal{G}_{s_6}$. Also, the original database schema and the pruned schema are depicted in the leftmost and rightmost sides, respectively, of the figure.

Fig. 2 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, as discussed next.

3.2 Subgraph Evaluation

This procedure calculates the correlation scores of different subsets of the entire subgraph set created in Section 3.1. In order to compute the correlation between subgraphs, the SESP strategy first obtains each subgraph’s embedded knowledge with respect to the target classes (denoted as *SubInfo*). Next, it calculates the correlation scores between these *SubInfo*, in order to approximate the correlation information between the corresponding subgraphs. Specifically, it consists of the following three steps. First, each subgraph is used to construct a relational classifier. Second, the constructed models are used to generate a data set in which each instance is described by different subgraphs’ *SubInfo*. Finally, the correlation scores of different *SubInfos* are computed. Each of these steps is discussed next, in detail.

3.2.1 Subgraph Classifier

Each subgraph created in Section 3.1 may be used to build a relational classifier using traditional efficient and accurate single-table learning algorithms (such as C4.5 (Quinlan, 1993) or SVMs (Burges, 1998)). These methods require “flat” data presentations. In order to employ these “flat” data methods, aggregation operators are usually used to squeeze a bag of tuples into one attribute-based entity in order to “link” the relations together (Perlich and Provost, 2006, 2003). For example, in the sample database, the count function may be used to determine the number of transactions associated with a particular loan, and thus “link” the Loan (target) and Transaction (background) tables. For instance, Knobbe (2004) applied aggregate functions such as min, max, and count for propositionalization in the RollUp relational learning system. Neville et al (2003) used aggregation functions such as average, mode, and count for the relational probability tree system. Also, Reutemann et al (2004) developed a propositionalization toolbox called Proper in the Weka system (Witten and Frank, 2000). The Proper Toolbox implements an extended version of the RelAggs algorithm designed by Krogel and Wrobel (2003) 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 nominal features from multiple tuples. Following the same line of thought, the SESP algorithm deploys the same aggregation functions employed in the RelAggs algorithm as implemented in Weka. Through generating relational features, each subgraph may separately “flatten” into a set of attribute-based training instances. Traditional well-studied learning algorithm such as decision trees or SVMs may therefore be applied to learn the relational target concept, forming a number of subgraph classifiers.

3.2.2 SubInfo

SubInfo is used to describe the knowledge held by a subgraph with respect to the target classes in the target relation. The idea here was inspired by the success of meta-learning algorithms (Chan and Stolfo, 1993; Giraud-Carrier et al, 2004). In a meta-learning setting such as Stacked Generalization (Merz, 1999; Ting and Witten, 1999; Wolpert, 1990), knowledge of the base learners is conveyed through their predictions in the meta level. These predictions serve as the confidence measure made by a given individual learner (Ting and Witten, 1999). Following the same line of thought, we here use class probabilistic predictions generated by a given subgraph classifier as its corresponding subgraph’s *SubInfo*.

Fig. 3 Subgraph evaluation process

The *SubInfo*, as described in Algorithm 1, is obtained as follows. Let $\{F_1, \dots, F_n\}$ be n classifiers, as described in Section 3.2.1, each formed from a different subgraph of the constructed subgraph set $\{\mathcal{G}_{s_1}, \dots, \mathcal{G}_{s_n}\}$, as presented in Section 3.1. Let $\mathcal{T}_e$ be an evaluation data set with m classes. For each instance t (with label L) in $\mathcal{T}_e$, each classifier F_i is called upon to produce prediction values $\{V_i^1, \dots, V_i^m\}$ for it. Here, V_i^c ($1 \leq c \leq m$) denotes the probability that instance t belongs to class c , as predicted by classifier F_i . Consequently, for each instance t in $\mathcal{T}_e$, a new instance t' is created. Instance t' consists of n sets of prediction values, i.e. $\mathcal{A} = \{\{V_1^k\}_{k=1}^m, \dots, \{V_n^k\}_{k=1}^m\}$, along with the original class label L . By doing so, this process creates a new data set $\mathcal{T}'_e$. Each instance $t' \in \mathcal{T}'_e$ is described by n variable sets $\{\{V_1^k(t')\}_{k=1}^m, \dots, \{V_n^k(t')\}_{k=1}^m\}$. For example, variables $\{V_1^k(t')\}_{k=1}^m$ are created by classifier F_1 and the variable set $\{V_2^k(t')\}_{k=1}^m$ is created by classifier F_2 , and so on. We define $\{V_i^k(t')\}_{k=1}^m$ to be the *SubInfo variables* of classifier

Table 1 Sample *SubInfo* variable values generated by the six subgraphs

$\mathcal{G}_{s_1}$		$\mathcal{G}_{s_2}$		$\mathcal{G}_{s_3}$		$\mathcal{G}_{s_4}$		$\mathcal{G}_{s_5}$		$\mathcal{G}_{s_6}$		class label
$V_1^1(t)$	$V_1^2(t)$	$V_2^1(t)$	$V_2^2(t)$	$V_3^1(t)$	$V_3^2(t)$	$V_4^1(t)$	$V_4^2(t)$	$V_5^1(t)$	$V_5^2(t)$	$V_6^1(t)$	$V_6^2(t)$	
0.93	0.07	0.21	0.79	0.73	0.27	0.01	0.99	0.43	0.57	0.51	0.49	1
0.66	0.34	0.32	0.68	0.46	0.54	0.12	0.88	0.86	0.14	0.82	0.18	2
0.89	0.11	0.81	0.19	0.69	0.31	0.61	0.39	0.39	0.61	0.31	0.69	2
0.22	0.78	0.47	0.53	0.02	0.98	0.27	0.73	0.73	0.27	0.97	0.03	1

F_i , which corresponds to subgraph $\mathcal{G}_{s_i}$. Figure 3 depicts this process for two subgraphs, i.e. $\mathcal{G}_{s_1}$ and $\mathcal{G}_{s_2}$. There, each instance in the new created data set consists of two sets of values, namely $\{V_1^1, \dots, V_1^m\}$ and $\{V_2^1, \dots, V_2^m\}$. These two sets were constructed by subgraph classifiers F_1 and F_2 , respectively.

Example 2: Let us resume Example 1. Recall from this example that, the database is decomposed into six subgraphs, namely $\mathcal{G}_{s_1}$, $\mathcal{G}_{s_2}$, $\mathcal{G}_{s_3}$, $\mathcal{G}_{s_4}$, $\mathcal{G}_{s_5}$, and $\mathcal{G}_{s_6}$. That is, six subgraph classifiers (denoted as F_1, F_2, F_3, F_4, F_5 , and F_6) are constructed, each built from one of the six subgraphs. At this learning stage, since the target variable *Risk Level* in the target relation *Loan* has two values, namely *High Risk* (denoted as 1) and *Low Risk* (denoted as 2), two *SubInfo* variables are generated for each classifier F_i . For instance, subgraph $\mathcal{G}_{s_1}$ creates two such variables with values of 0.93 and 0.07 for the first instance, as described in Table 1. These two numbers indicate the classifier F_i 's confidence levels of assigning the instance into class 1 and 2, respectively. Similarly, against the first instance, two *SubInfo* variables with values of 0.21 and 0.79, respectively, are generated for subgraph $\mathcal{G}_{s_2}$. Table 1 shows the twelve *SubInfo* variables (the second row in Table 1, as highlighted in yellow), along with four (4) sample instances generated by the six subgraphs. Each of the four instances also contains the original class labels of the evaluation data. Consequently, a data set with *SubInfo* variable values is created. Next, the degree of correlation between these *SubInfo* variables is evaluated.

3.2.3 Correlation of Subgraphs

In this step, we aim to identify a subset of subgraphs which are highly correlated with the target concept, but irrelevant to one another. That is, we aim to measure the "goodness" of a given subset of subgraphs.

Methods for selection of a subset of variables have been studied by many researchers (Almuallim and Dietterich, 1991, 1992; Ghiselli, 1964; Hall, 1998; Hogarth, 1977; Kira and Rendell, 1992; Kohavi et al, 1997; Koller and Sahami, 1996; Liu and Setiono, 1996; Zajonic, 1962). 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, Koller and Sahami (1996) described a strategy for selecting a subset of features, which can aid to better predict the class. Their approach aims to choose the subset which retains 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, nevertheless, requires the user to specify the number of the final attribute subset.

Following Koller and Sahami's idea of finding Markov boundaries for feature selection, Margaritis (2009) proposes a feature selection strategy for arbitrary domains. The algorithm aims to find subsets of the provided variable set. Features within such a subset are conditionally dependent on the target variable, but all the remaining variables in the set are conditionally independent from the target variable. Using a user-specified size of m , the so-call GS method exhaustively examines all the possible variable subsets with size up to m , aiming to find the Markov boundaries. To cope with the computational difficulty of exhaustive search when given a feature set with a large number of variables, the author also presents a practical version of the GS algorithm. Unlike the original GS algorithm, this practical version evaluates only a randomly selected k sets from the large number of potential feature subsets with size less than m . Nevertheless, as for m , k also needs to be carefully selected beforehand.

In addition to research in the feature selection field, studies on identifying redundant graphs have also been introduced. Pearl (1988) describes the d-separation algorithm to compute the conditional independence relations entailed in a directed graph. However, in order to apply the d-separation strategy, we need to have a probabilistic graphical models such as a Bayesian Networks (Heckerman, 1998) to map the variables into a directed graphical model and to correctly capture the probability distributions among these variables; this work is not trivial (Heckerman et al, 1995).

The above methods, unfortunately, cannot fulfill our subgraph subset identification requirement. In other words, they are unable to automatically 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 contradictory requirements. That is, on the one hand, this subset of variables has to be strongly correlated with the class, so that it can aid us to better predict the class labels. Consider this correlation information (denoted as R_{cf}) is calculated by averaging correlation scores of all variable-to-class pairs. Thus, keeping expanding the number of variables in the subset increases the value of R_{cf} . On the other hand, the correlation score between variables within this subset is required to be as low as possible, so that they contain diverse knowledge. Suppose we calculate this score (denoted as R_{ff}) by averaging the correlation information between all variable-to-variable pairs. In such a scenario, the value of R_{ff} will be decreasing if we keep removing variables from the subset.

In order to measure the level of such "goodness" of a given subset of subgraphs, the SESP strategy adapts a heuristic principle from the test theory (Ghiselli, 1964):

$$Q = \frac{K\overline{R_{cf}}}{\sqrt{K + K(K-1)\overline{R_{ff}}}} \quad (1)$$

Here, K is the number of variables in the subset, $\overline{R_{cf}}$ is the average variable-to-class correlation, and $\overline{R_{ff}}$ represents the average variable-to-variable dependence. This formula has previously been applied in test theory to estimate an external variable of interest (Ghiselli, 1964; Hogarth, 1977; Zajonic, 1962). In addition, Hall has adapted it into the CFS feature selection strategy (Hall, 1998), where

this measurement aims to discover a subset of features which are highly correlated to the class.

To measure the degree of correlation between variables and the target class and between the variables themselves, we also adopt the notion of *Symmetrical Uncertainty* ($\mathcal{U}$) (Press et al, 1988) to calculate $\overline{R_{cf}}$ and $\overline{R_{ff}}$. This score is a variation of the *Information Gain* (*InfoGain*) measure (Quinlan, 1993). It compensates for *InfoGain*'s bias toward attributes with more values, and has been successfully applied by Ghiselli (1964) and Hall (1998). *Symmetrical Uncertainty* is defined as follows:

Given variables X and Y ,

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

where $H(X)$ and $H(Y)$ are the entropies of the random variables X and Y , respectively. Entropy is a measure of the uncertainty of a random variable. The entropy of a random variable Y is defined as

$$H(Y) = - \sum_{y \in Y} p(y) \log_2(p(y))$$

And the *InfoGain* is given by

$$\begin{aligned} \text{InfoGain} = & - \sum_{y \in Y} p(y) \log_2(p(y)) \\ & + \sum_{x \in X} p(x) \sum_{y \in Y} p(y|x) \log_2(p(y|x)) \end{aligned} \quad (2)$$

Note that, these measures need all of the variables to be nominal, so numeric variables are first discretized.

Next, $\mathcal{Q}$ may be applied in order to identify the set of uncorrelated subgraphs.

3.2.4 Subgraph Pruning

In order to identify a set of uncorrelated subgraphs, the evaluation procedure searches all of the possible *SubInfo variable* subsets, and constructs a ranking on them. The best ranking subset will be selected, i.e. the subset with the highest $\mathcal{Q}$ value.

To search the *SubInfo variable* space, the SESP method uses a best first search strategy (Kohavi and John, 1997). The method starts with an empty set of variables, and keeps expanding, one variable at a time. In each round of the expansion, the best variable subset, namely the subset with the highest "goodness" value $\mathcal{Q}$ is chosen. In addition, the SESP algorithm terminates the search if a preset number of consecutive non-improvement expansions occurs. Based on our experimental observations we empirically set the number to five (5).

Finally, subgraphs are selected based on the final best subset of *SubInfo variables*. If a subgraph has no *SubInfo variables* that are strongly correlated to the class, the knowledge possessed by this subgraph may 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 *SubInfo variables* appears in the final best ranking subset.

Table 2 Identified *SubInfo* variable set

$\mathcal{G}_{s_1}$		$\mathcal{G}_{s_2}$		$\mathcal{G}_{s_3}$		$\mathcal{G}_{s_4}$		$\mathcal{G}_{s_5}$		$\mathcal{G}_{s_6}$		class label
$V_1^1(t)$	$V_1^2(t)$	$V_2^1(t)$	$V_2^2(t)$	$V_3^1(t)$	$V_3^2(t)$	$V_4^1(t)$	$V_4^2(t)$	$V_5^1(t)$	$V_5^2(t)$	$V_6^1(t)$	$V_6^2(t)$	
0.93	0.07	0.21	0.79	0.73	0.27	0.01	0.99	0.43	0.57	0.51	0.49	1
0.66	0.34	0.32	0.68	0.46	0.54	0.12	0.88	0.86	0.14	0.82	0.18	2
0.89	0.11	0.81	0.19	0.69	0.31	0.61	0.39	0.39	0.61	0.31	0.69	2
0.22	0.78	0.47	0.53	0.02	0.98	0.27	0.73	0.73	0.27	0.97	0.03	1

Example 3: As an example, let's continue the running example, i.e. Example 2. Recall that, twelve *SubInfo* variables (as shown in the second row of Table 2) have been constructed for the six subgraphs, i.e. subgraphs $\mathcal{G}_{s_1}$, $\mathcal{G}_{s_2}$, $\mathcal{G}_{s_3}$, $\mathcal{G}_{s_4}$, $\mathcal{G}_{s_5}$, and $\mathcal{G}_{s_6}$. Consider the SESP algorithm identified a final *SubInfo* variable subset with two variables, namely, $V_2^1(t)$ and $V_4^2(t)$ (as highlighted in yellow cells in Table 2). That is, this subset has the highest Q value among all the visited *SubInfo* variable subsets. This subset implies that only knowledge from subgraphs $\mathcal{G}_{s_2}$ and $\mathcal{G}_{s_4}$ really contribute to building the final model. Thus, subgraphs $\mathcal{G}_{s_2}$ and $\mathcal{G}_{s_4}$ are selected by the subgraph evaluation method. All other subgraphs are pruned, because they are considered as either irrelevant or redundant. That is to say, subgraphs $\mathcal{G}_{s_1}$, $\mathcal{G}_{s_3}$, $\mathcal{G}_{s_5}$, and $\mathcal{G}_{s_6}$ either contain very similar knowledge (with respect to the classification task) to that of the subgraphs $\mathcal{G}_{s_2}$ and/or $\mathcal{G}_{s_4}$ or are uncorrelated to the target classes. In this way, the running example database as depicted in Figure 1 results in pruning relations *Order* and *Life Style*. The pruned schema is pictured at the right hand side of Figure 4.

Fig. 4 Subgraph evaluation and pruning

In summary, Algorithm 1 describes the SESP algorithm. 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 classifier. Thirdly, *SubInfo* variables are generated for subgraphs using corresponding subgraph classifiers. Subsequently, a best first search strategy is employed to select the best

subset of *SubInfo variables*. Finally, subgraphs are pruned if none of their *SubInfo variables* appear in the final *SubInfo variables* 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.

4 Experimental Results

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. In addition, we also present how the databases are pruned. We perform our experiments using the MRC (Guo and Viktor, 2006), RelAggs (Kroegel, 2005), TILDE (Blockeel and Raedt, 1998), and CrossMine (Yin et al, 2006) algorithms, with their default settings. The MRC and RelAggs approaches are aggregation-based algorithms where C4.5 decision trees (Quinlan, 1993) were applied as the single-table learner. The C4.5 decision tree learner was used due to its de facto standard for empirical comparisons. In contrast, the CrossMine and TILDE methods are two benchmark logic-based strategies. In addition, we set the maximum length of joins, namely *MaxJ* of the SESP strategy to *two (2)*. That is, we only consider join paths which contain less than four tables. This number was empirically determined and provides a good trade off between accuracy and execution time (further discussion is provided in Section 4.2.2). The C4.5 decision tree algorithm was used as the subgraph 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). Note that, we implemented the aggregation calculation within the MySQL database in order to take advantage of the aggregation techniques, memory allocation strategies, and computational power of the database management system to enhance the learning process.

4.1 Real Databases

4.1.1 ECML98 Database

Our first experiment uses the database from the ECML 1998 Sisyphus Workshop. This database was extracted from the customer data warehouse of a Swiss insurance company (Kietz et al, 2000). The learning task (ECML98) is to categorize the 7,329 households into class 1 or 2 (Kroegel, 2005). Eight background relations are provided for this 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 schema prepared in (Kroegel, 2005).

4.1.2 Financial Database

Our second experiment uses the financial database from the PKDD 1999 discovery challenge (Berka, 2000). The database was offered by a Czech bank and contains typical business data. This database consists of eight tables, including a class attribute which indicates the status of the loan, i.e. A (finished and good), B

Table 3 Accuracies obtained using methods MRC, RelAggs, TILDE, and CrossMine against the original and pruned schemas

Schema	MRC		RelAggs		TILDE		CrossMine	
	Original	Pruned	Original	Pruned	Original	Pruned	Original	Pruned
F682AC	93.4 %	93.4 %	92.1 %	92.9 %	88.9 %	88.8%	90.3 %	90.3 %
F400AC	88.0 %	88.0 %	89.0 %	86.8 %	81.3 %	81.0%	85.8 %	87.3 %
F234AC	92.3 %	92.3 %	90.2 %	90.2 %	86.8 %	86.8%	88.0 %	89.4 %
ECML98	88.2 %	87.5 %	88.0 %	86.2 %	53.7 %	52.0%	85.3 %	83.7 %

Table 4 Compression rates, in terms of the number of relations, tuples, and attributes, achieved by the SESP method for the four learning tasks, along with the maximum length of join paths in the original and pruned database schemas

Schema	Num. of Tables			Num. of Records			Num. of Attributes			Leng. of Join Path	
	Ori.	Pru.	Rate	Ori.	Pru.	Rate	Ori.	Pru.	Rate	Ori.	Pru.
F682AC	8	2	75.0%	76264	53586	29.74%	52	14	73.08%	5	1
F400AC	8	5	37.5 %	75982	12240	83.89%	52	16	69.23%	5	2
F234AC	8	3	62.5%	75816	65870	13.12%	52	28	46.15%	5	1
ECML98	9	4	55.5%	197478	62429	68.38%	123	24	80.48%	1	1

Table 5 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 Time
	Original	Pruned	Original	Pruned	Original	Pruned	Original	Pruned	
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	424.43	220.99	1703.58	1206.39	1108.60	167.76	570.90	366.78	356.24

(finished but bad), C (good but not finished), or D (bad and not finished). In order to test how different numbers of tuples in the target relation (with the same database schema) affect the performance of the SESP algorithm, we derived three learning tasks from this database. Each of these three tasks has a different number of target tuples but shares the same background relations. Our first learning task (F234AC) is to learn if a loan is good or bad from the 234 finished tuples. The second learning problem (F682AC) 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 (F400AC) uses the Financial database as prepared in (Yin et al, 2006), which has 400 examples in the target table.

4.1.3 Experimental Results and Discussion

The predictive accuracy we obtained, using MRC, RelAggs, TILDE, and CrossMine is presented in Table 3. The results obtained with the respective original and pruned schemas are shown side by side. The pruned schemas are the schemas resulting from the application of the SESP strategy as a data pre-processing step for the four tested algorithms. In table 5, we also provide the execution time of the pruning process, as well as the running time required for the four tested al-

gorithms against the original and pruned schemas. In addition, we present, in Table 4, the number of relations, tuples, and attributes before (denoted as Ori.) and after (denoted as Pru.) the pruning, along with their compression rates (denoted as Rate) achieved by the SESP approach. The compression rate considers the number of objects (either relation, tuple, or attribute) of the original schema ($N_{original}$) and the number of objects pruned (N_{pruned}), which is calculated as $(N_{original} - N_{pruned})/N_{original}$. In the last two columns of Table 4, we also show the maximum length of join paths in the original and pruned database schemas.

From Tables 3 and 4, 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 multirelational classification models in terms of the accuracy obtained. The results shown in Tables 3 and 4 provide us with two meaningful observations. The first is that the SESP strategy is capable of pruning the databases meaningfully. The results, as shown in Table 4, indicate that the compression rates for the number of relations for these four learning schemas are 75%, 62.5%, 37.5%, and 55.5%, respectively. The number of tables, originally eight, eight, eight, and nine, were pruned to two, three, five, and four for tasks F682AC, F234AC, F400AC and ECML98, respectively. In terms of the number of attributes, the compression rates for all four tasks are at least 45%. Promisingly, the number of records for the databases are also significantly reduced. For example, only 16.11% and 31.62% of the original tuples are needed to form accurate classifiers when against the databases F400AC and ECML98, respectively. In particular, our results, as shown in the last column of the Table 4, also demonstrate that the maximum length of join path needed to search for a multirelational classification algorithm is very small. These results suggest that, in three of the four test cases, join paths involving with two relations were sufficient for building an accurate relational classifier.

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, results as presented in Table 3 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 accuracies produced by this method against both the pruned and full schemas were also very close. In two cases (F400AC and F234AC) the predictive performance on the pruned schemas outperformed that of the original structures. One exception

Table 6 Parameters for the Data Generator

Parameter	Value
Number of relations	10, 20, 50, 80, 100, or 150
Min number of tuples in each relation	50
Expected number of tuples in each relation	1000
Min number attributes in each relation	2
Expected number of attributes in each relation	15
Min number of values in each attribute	2
Expected number of values in each attribute	10
Expected number of foreign keys in each relation	2

is the performance with the ECML98 database, where a slight decrease of 1.6% against the full schema was observed.

In terms of computational cost of the SESP method, results presented in Table 5 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. For example, for the TILDE algorithm, against all the four test cases, the learning time required on the pruned databases was less than 20% of that of learning against the original databases.

In short, these results imply that the SESP strategy can significantly reduce the size of the relational databases, while still maintaining predictive accuracy of the final classification model. Furthermore, our results suggest that relations close to the target relations should be paid more attention while building an accurate classifier.

4.2 Synthetic Databases

To further examine the pruning effect of the SESP algorithm, we generated six synthetic databases with different characteristics. The aims of these experiments were to further explore the applicability of the SESP algorithm when considering relational domains with a varying number of relations and tuples.

The database generator was obtained from Yin et al (2006). In their paper, Yin et al. used this database generator to create synthetic databases to mimic real-world databases in order to evaluate the scalability of the multirelational classification algorithm CrossMine. To create a database, the generator first generates a relational schema with a specified number of relations. Among them, the first randomly generated table was chosen as the target relation and the others were used as background relations. In this step, a number of foreign keys is also generated following an exponential distribution. These joins connect the created relations and form different join paths for the databases. Finally, synthetic tuples with categorical attributes (integer values) are created and added to the database schema. 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. Interested readers please refer to the paper presented by Yin et al (2006) for detailed discussions of the database generator.

For each database in this paper, we set the expected number of tuples and attributes to 1000 and 15, respectively. Default values for other parameters of the data generator were used. Table 6 listed some of the major parameters used in this paper. The six databases were generated with 10, 20, 50, 80, 100, and 150 relations (denoted as SynR10, SynR20, SynR50, SynR80, SynR100, and SynR150), respectively.

Against these six synthetic databases, we compare the accuracy of a relational classifier constructed from the original schema with that when training the corresponding pruned schema. We used the MRC and CrossMine methods as the relational learners. These two algorithms were chosen because our experiments conducted against real-world databases, as described in Section 4.1, show that they achieved a good balance between scalability and predictive accuracy. Again, all experiments were performed using ten-fold cross validation.

Fig. 5 Accuracies obtained by the MRC and CrossMine methods for original and pruned schemas against the six synthetic databases

Fig. 6 Execution time (seconds) required by the MRC and CrossMine methods for original and pruned schemas against the six synthetic databases

Table 7 Compression rates, in terms of the number of relations, tuples, and attributes, achieved by the SESP method for the six synthetic learning tasks, along with the maximum length of join paths in the original and pruned database schemas

Schema	Num. of Tables			Num. of Records			Num. of Attributes			Leng. of Join Path	
	Ori.	Pru.	Rate	Ori.	Pru.	Rate	Ori.	Pru.	Rate	Ori.	Pru.
SynR10	10	7	30.0%	10794	8708	19.33%	172	107	37.79%	9	2
SynR20	20	5	75.0%	22032	6125	72.20%	325	57	82.46%	10	2
SynR50	50	9	82.0 %	37293	5927	84.11%	763	143	81.26%	18	2
SynR80	80	11	86.2%	71974	11721	83.71%	1362	180	86.78%	>20	2
SynR100	100	12	88.0 %	100395	14521	85.54%	1472	195	86.75%	>20	2
SynR150	150	8	94.6 %	138148	3651	97.36%	2542	135	94.69%	>20	2

4.2.1 Pruning Effect

For each of the six synthetic databases, the accuracies obtained against both original and pruned schemas by the MRC and CrossMine methods are described in Figures 5(a) and 5(b), respectively. The sizes of the databases, along with their compression rates achieved, in terms of the number of relations, tuples, and attributes, obtained for each of the six databases are provided in Table 7. We also provide the execution time needed using the MRC and CrossMine algorithms against the original and pruned schemas in Figure 6.

Fig. 7 The original (left) and pruned schemas (right) of databases SynR10

From Figures 5 and 6 and Table 7, one can again deduce that the SESP algorithm not only significantly reduces the size of the relational databases, in terms of the number of relations, tuples, and attributes, but also produces very comparable classification models in terms of the accuracy obtained. The results also show that the accuracies are comparable regardless of 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 than 1.5%. In addition, results as presented in Figure 6

Fig. 8 The original (left) and pruned schemas (right) of databases SynR20

Fig. 9 The original (left) and pruned schemas (right) of databases SynR50

Fig. 10 The original (left) and pruned schemas (right) of databases SynR80

show that the execution time needed for constructing relational models using the two tested algorithms was meaningfully reduced when pruned.

In terms of reducing the size of the databases, the results obtained were quite significant. As shown in Table 7, the compression rates, in terms of the number of relations, the number of tuples, and the number of attributes were more than 80% for all 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. For example, for the SynR150 database, less than 6% of its original relations, tuples, and attributes was useful for building an accurate classifier.

Fig. 11 The original (left) and pruned schemas (right) of databases SynR100

Fig. 12 The original (left) and pruned schemas (right) of databases SynR150

Results, as shown in Table 7, also suggest another important observation. That is, although the maximum length of join paths are large (the number is 9, 10, 18 for databases SynR10, SynR20, and SynR50, respectively; and the number is larger than 20 for the other three databases), while building an accurate classifier, only join paths with length equal to two (2) were used. To visualize how these six synthetic databases were pruned, we provided the graphic schema results in Figures 7, 8, 9, 10, 11, and 12, where circles stands for relations and lines for joins. Also, we highlighted the target relation in each schema with *green*. The database schemas (before and after the SESP approach) for the experiments, i.e. SynR10, SynR20, SynR50, SynR80, SynR100, and SynR150 are provided in Figures 7, 8, 9, 10, 11, and 12, respectively. The original schemas are on the left hand side and the pruned structures on the right. These results further confirm our previous observations as discussed earlier this section. That is, the approach significantly reduced the size of the six databases. Importantly, the results visually suggest that more weight should be put on the relations closer to the target table in a multirelational classification task.

4.2.2 Impact of Join Path Length

Recall from Section 4 that, our previous experiments heuristically set the maximum length of join path, i.e. $MaxJ$ in the SESP algorithm to two (2). However, as can be seen from Table 7, the maximum length of join path in the above six synthetic databases is more than twenty. In order to further examine the impact of

this heuristic number, we test the performance of the MRC strategy with respect to the *MaxJ*. By doing so, we intended to verify if there was a need for relational learning algorithms to explore longer join paths from the databases provided. We chose the MRC approach since we can perfectly control its search depth within a database, in terms of the length of join path.

Against each of the six synthetic databases, the MRC strategy varied its *MaxJ* value from zero (0) to ten (10) (zero means using only the target relation to build the model). In other words, we here allow a join path to involve up to eleven (11) tables. We chose this number, because for complex structure databases (such as the SynR150 database) the training time required for the ten-fold cross validation was greater than 6600 seconds. In addition, the required execution time exponentially increased as the value for *MaxJ* extends. We provided the predictive accuracy obtained, the average running time required for each fold of the ten-fold cross validation, and the number of subgraphs used for building the model by the MRC methods in Figures 13(a), 13(b), and 13(c), respectively. Note that, to speed up the execution, we ran these experiments on a 2.66Ghz Intel Quad CPU with 4 GByte of RAM.

From Figures 13(a) and 13(b), one observes that when *MaxJ* equals two, the MRC method provided a good trade-off between predictive performance obtained and execution time needed. Against four of the six databases (each tested with 11 different *MaxJ* values), the SESP algorithm obtained the highest accuracy when the maximum length of join path was set to two. The two exceptions were against the databases SynR50 and SynR100, when the *MaxJ* value was set to five. In these two cases, the MRC approach with *MaxJ* of two achieved slightly lower accuracy (less than 1%), compared to that of setting the *MaxJ* to five. However, the results as provided in Figure 13(b) suggest that the execution time required for the two tested cases with the *MaxJ* value of five may double, compared to that of cases with value of two. As may be observed from Figure 13(b), the running time needed for most of the tested cases exponentially increases with respect to the maximum length of join path allowed for the search, i.e. *MaxJ*. These results imply that, in most of the tested cases, when *MaxJ* equals two the MRC algorithm not only obtained the best accuracy but also required reasonable execution time.

Figure 13(c) also demonstrates that, the number of subgraphs used for training the model increases very quickly when extending the length of join path allowed to search by the MRC algorithm. For example, as shown in Figure 13(c), when *MaxJ* equals 10, the number of subgraphs used by the MRC algorithm against the SynR50, SynR80, SynR100, and SynR150 databases was over 2000. However, when compared to that of setting the value of *MaxJ* to two (2), the large number of additional subgraphs used did not help improve the predictive accuracy of the constructed models, but dramatically increase the execution time required.

In short, setting the value of *MaxJ* to two (2) provided us with a good trade-off between accuracy achieved and execution time needed. On the one hand, if we allow the SESP algorithm to search join paths with less depth, we may be able to further prune objects from a database. However, this could significantly decrease the relational learning algorithm's predictive performance against the pruned databases when building classification models. On the other hand, if we force the SESP approach to search deeper join paths, it may not improve the accuracy of the constructed model, but dramatically increase the execution time required for the learning process, as shown in Figures 13(a) and 13(b).

Fig. 13 Predictive accuracy obtained, the average running time required for each fold of the ten-fold cross validation, and the number of subgraphs used for building the model w.r.t maximum length of Join Path

While we only chose the MRC algorithm to evaluate the SESP approach's heuristic number for $MaxJ$, we believe that such a setting provides us with a good testbed. Our research as reported in (Guo and Viktor, 2008) has shown that the MRC algorithm is able to conduct superior or very comparable accuracies, when compared with the other three examined algorithms as used in Section 4.1, namely, RelAgg, TILDE and CrossMine.

In summary, our experimental results on real and synthetic databases show that the SESP strategy may significantly reduce the size of the relational databases, while maintaining predictive accuracy for the final classification model. That is, one may build an accurate classification model with only a small subset of the original database. In other words, from a utility-based learning perspective, the small relevant part of a complex database may be identified for efficient learning, thus benefiting the learning's economic utility such as cost associated with acquiring the training data, cleaning the data, transforming the data, and managing the relational databases.

5 Conclusions and Discussions

Multirelational data mining applications usually involve a large number of relations, where each may come from a different party. Unfortunately, acquiring and managing such data is often expensive, in terms of data mining overheads. Also, the size of a database may pose severe scalability problem for multirelational classification tasks.

This article reports the SESP strategy, which aims to pre-prune uninteresting relations and tuples in order to reduce the scale of relational learning tasks. Our method creates a pruned subset of the original database and minimizes the predictive performance loss incurred by the final classification model. The experiments performed, against both real-world and synthetic databases, show that our strategy is able to, without sacrificing predictive accuracy, significantly reduce the size of the databases, in terms of the number of relations, tuples, and attributes, for multirelational classification. The approach prunes the size of databases by as much as 94%. Such reduction also results in decreasing computational cost of the learning process. The method improves the multirelational learning algorithms' execution time by as much as 80%.

This paper makes two chief contributions to the multirelational data mining community. First, a novel approach, namely the SESP strategy, is devised to reduce the learning scale for multirelational mining. One may use the SESP approach to retrieve a very compact database schema for building an accurate classification model, resulting in saving economic and computational cost in the knowledge discovery process. Another contribution is that, our research experimentally demonstrates that one may build an accurate classification model with only a small subset of the provided database.

Several future directions would be worth investigating. Firstly, while experimental results are promising, we intend to study the statistic sufficiency of using *Subinfo* to describe a subgraph for the subgraph evaluation element in the SESP method. Secondly, we plan to study how the structure of the database impacts the SESP method. Intuitively, the structure of the foreign keys and the functional dependencies among tuples in a database should play an important role on the shape of the pruned schema. We intend to further address these issues. For example, with synthetic databases, we can control the interconnections among relations as well as the correlations among the attributes across these relations. Therefore, we may obtain a better understanding about the SESP pruning strategy. Finally, as discussed in Section 3.2.3, the Markov boundaries approach for feature selection cannot be directly applied to our proposed strategy. Nevertheless, we consider this research line promising. We aim to integrate Markov boundaries algorithms within the SESP framework.

References

- Almuallim H, Dietterich TG (1991) Learning with many irrelevant features. In: AAAI '91, AAAI Press, Anaheim, California, vol 2, pp 547–552
- Almuallim H, Dietterich TG (1992) Efficient algorithms for identifying relevant features. Tech. rep., Corvallis, OR, USA

- Alphonse E, Matwin S (2004) Filtering multi-instance problems to reduce dimensionality in relational learning. *Journal of Intelligent Information Systems* 22(1):23–40
- Berka P (2000) Guide to the financial data set. In: A. Siebes and P. Berka, editors, PKDD2000 Discovery Challenge
- Bhattacharya I, Getoor L (2007) Collective entity resolution in relational data. *ACM Trans Knowl Discov Data* 1(1):5
- Blockeel H, Raedt LD (1998) Top-down induction of first-order logical decision trees. *Artificial Intelligence* 101(1-2):285–297
- Bringmann B, Zimmermann A (2009) One in a million: picking the right patterns. *Knowl Inf Syst* 18:61–81
- Burges CJC (1998) A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery* 2(2):121–167
- Burnside JDE, Ramakrishnan R, Costa VS, Shavlik J (2005) View learning for statistical relational learning: With an application to mammography. In: *Proceeding of the 19th IJCAI*, pp 677–683
- Ceci M, Appice A (2006) Spatial associative classification: propositional vs structural approach. *Journal of Intelligent Information Systems* 27:191–213
- Chan PK, Stolfo SJ (1993) Experiments on multistrategy learning by meta-learning. In: *CIKM '93*, ACM Press, New York, NY, USA, pp 314–323
- Chen BC, Ramakrishnan R, Shavlik JW, Tamma P (2009) Bellwether analysis: Searching for cost-effective query-defined predictors in large databases. *ACM Trans Knowl Discov Data* 3(1):1–49
- Cohen W (1995) Learning to classify English text with ILP methods. In: De Raedt L (ed) *ILP '95, DEPTCW*, pp 3–24
- De Marchi F, Petit JM (2007) Semantic sampling of existing databases through informative armstrong databases. *Inf Syst* 32(3):446–457
- De Raedt L (2008) *Logical and Relational Learning*. Cognitive Technologies, Springer
- Dehaspe L, Toivonen H, King RD (1998) Finding frequent substructures in chemical compounds. *AAAI Press*, pp 30–36
- Dzeroski S, Lavrac N (2001) Saso Dzeroski and Nada Lavrac, editors, *Relational Data Mining*. Springer, Berlin
- Frank R, Moser F, Ester M (2007) A method for multi-relational classification using single and multi-feature aggregation functions. In: *PKDD 2007*, pp 430–437
- Getoor L, Taskar B (2007) editors, *Statistical Relational Learning*. MIT Press, In press
- Ghiselli EE (1964) *Theory of Psychological Measurement*. McGrawHill Book Company
- Giraud-Carrier CG, Vilalta R, Brazdil P (2004) Introduction to the special issue on meta-learning. *Machine Learning* 54(3):187–193
- Guo H, Viktor HL (2006) Mining relational data through correlation-based multiple view validation. In: *KDD '06*, New York, NY, USA, pp 567–573
- Guo H, Viktor HL (2008) Multirelational classification: a multiple view approach. *Knowl Inf Syst* 17(3):287–312
- Guo H, Viktor HL, Paquet E (2007) Pruning relations for substructure discovery of multi-relational databases. In: *PKDD*, pp 462–470

- Guo H, Viktor HL, Paquet E (2011) Privacy disclosure and preservation in learning with multi-relational databases. *JCSE* 5(3):183–196
- Habrador A, Bernard M, Sebban M (2005) Detecting irrelevant subtrees to improve probabilistic learning from tree-structured data. *Fundamenta Informaticae: Special Issue on Mining Graphs, Trees and Sequences*
- Hall M (1998) Correlation-based feature selection for machine learning, Ph.D thesis, department of computer science, university of waikato, new zealand
- Hamill R, Martin N (2004) Database support for path query functions. In: *Proc. of 21st British National Conference on Databases (BNCOD 21)*, pp 84–99
- Han J, Kamber M (2005) *Data Mining: Concepts and Techniques*, 2nd Edition. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA
- Heckerman D (1998) A tutorial on learning with bayesian networks. In: *Proceedings of the NATO Advanced Study Institute on Learning in Graphical Models*, Kluwer Academic Publishers, Norwell, MA, USA, pp 301–354
- Heckerman D, Geiger D, Chickering DM (1995) Learning bayesian networks: The combination of knowledge and statistical data. *Machine Learning* 20(3):197–243
- Hogarth R (1977) Methods for aggregating opinions. In: H. Jungermann and G. de Zeeuw, editors, *Decision Making and Change in Human Affairs.*, Dordrecht-Holland
- Jamil HM (2002) Bottom-up association rule mining in relational databases. *Journal of Intelligent Information Systems* 19(2):191–206
- Jensen D, , Jensen D, Neville J (2002) Schemas and models. In: *In Proceedings of the SIGKDD-2002 Workshop on Multi-relational Learning*, pp 56–70
- Kietz JU, Zcker R, Vaduva A (2000) 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
- Kira K, Rendell LA (1992) A practical approach to feature selection. In: *ML92: Proceedings of the 9th International Workshop on Machine Learning*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp 249–256
- Knobbe AJ (2004) Multi-relational data mining. PhD thesis, University Utrecht
- Kohavi R, John GH (1997) Wrappers for feature subset selection. *Artificial Intelligence* 97(1-2):273–324
- Kohavi R, Langley P, Yun Y (1997) The utility of feature weighting in nearest-neighbor algorithms. In: *ECML '97*, Springer-Verlag, Prague, Czech Republic
- Koller D, Sahami M (1996) Toward optimal feature selection. In: *ICML '96*, pp 284–292
- Kroegel MA (2005) On propositionalization for knowledge discovery in relational databases. PhD thesis, Otto-von-Guericke-Universitt Magdeburg
- Kroegel MA, Wrobel S (2003) Facets of aggregation approaches to propositionalization. In: *ILP'03*
- Landwehr N, Kersting K, Raedt LD (2007) Integrating naive bayes and foil. *J Mach Learn Res* 8:481–507
- Landwehr N, Passerini A, Raedt LD, Frasconi P (2010) Fast learning of relational kernels. *Machine Learning* 78(3):305–342
- Lipton RJ, Naughton JF, Schneider DA, Seshadri S (1993) Efficient sampling strategies for relational database operations. *Theor Comput Sci* 116(1&2):195–226

- Liu H, Setiono R (1996) A probabilistic approach to feature selection - a filter solution. In: ICML '96, pp 319–327
- Margaritis D (2009) Toward provably correct feature selection in arbitrary domains. In: NIPS, pp 1240–1248
- Merz CJ (1999) Using correspondence analysis to combine classifiers. *Mach Learn* 36(1-2):33–58
- Neville J, Jensen D, Friedland L, Hay M (2003) Learning relational probability trees. In: Proceedings of the Ninth ACM SIGKDD, ACM Press, New York, NY, USA, pp 625–630
- Olken F, Rotem D (1986) Simple random sampling from relational databases. In: VLDB, pp 160–169
- Pearl J (1988) Probabilistic reasoning in intelligent systems: networks of plausible inference. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA
- Perlich C, Provost F (2006) Distribution-based aggregation for relational learning with identifier attributes. *Mach Learn* 62(1-2):65–105
- Perlich C, Provost FJ (2003) Aggregation-based feature invention and relational concept classes. In: KDD'03, pp 167–176
- Press WH, Flannery BP, Teukolsky SA, Vetterling WT (1988) Numerical recipes in C: the art of scientific computing. Cambridge University Press
- Quinlan JR (1993) C4.5: programs for machine learning. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA
- Quinlan JR, Cameron-Jones RM (1993) Foil: A midterm report. In: ECML '93, pp 3–20
- Reutemann P, Pfahringer B, Frank E (2004) A toolbox for learning from relational data with propositional and multi-instance learners. In: Australian Conference on Artificial Intelligence, pp 1017–1023
- Rückert U, Kramer S (2008) Margin-based first-order rule learning. *Mach Learn* 70:189–206
- Singh L, Getoor L, Licamele L (2005) Pruning social networks using structural properties and descriptive attributes. In: ICDM '05, pp 773–776
- Ting KM, Witten IH (1999) Issues in stacked generalization. *J Artif Intell Res (JAIR)* 10:271–289
- Witten IH, Frank E (2000) Data mining: practical machine learning tools and techniques with Java implementations. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA
- Wolpert DH (1990) Stacked generalization. Tech. Rep. LA-UR-90-3460, Los Alamos, NM
- Yin X, Han J, Yang J, Yu PS (2006) Efficient classification across multiple database relations: A crossmine approach. *IEEE Transactions on Knowledge and Data Engineering* 18(6):770–783
- Zajonc R (1962) A note on group judgements and group size. *Human Relations* 15:177–180
- Zhong N, Ohsuga S (1995) KOSI - an integrated system for discovering functional relations from databases. *Journal of Intelligent Information Systems* 5(1):25–50
- Zucker JD, Ganascia JG (1996) Representation changes for efficient learning in structural domains. In: ICML '96, pp 543–551