Pruning Relations for Substructure Discovery of Multi-relational Databases

Hongyu Guo[1], Herna L. Viktor[1], and Eric Paquet[2]

[1] School of Information Technology& Engineering, University of Ottawa, Canada
{hguo028, hlviktor}@site.uottawa.ca,
[2] National Research Council of Canada, Ottawa, Canada
eric.paquet@nrc-cnrc.gc.ca

Abstract. Multirelational data mining methods discover patterns across multiple interlinked tables (relations) in a relational database. In many large organizations, such a multi-relational database 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 multirelational classification, it follows that these subdivisions will express different interests in the data, leading to the need to explore various subsets of relevant relations with high utility with respect to the target class. The paper presents a novel approach for pruning the uninteresting relations of a relational database where relations come from such different parties and spans many classification tasks. We aim to create a pruned structure and thus minimize predictive performance loss on the final classification model. Our method identifies a set of strongly uncorrelated subgraphs to use for training and discards all others. The experiments performed demonstrate that our strategy is able to significantly reduce the size of the relational schema without sacrificing predictive accuracy.

1 Introduction

Knowledge discovery from relational databases poses a unique opportunity for the data mining community. In many large organizations, such a relational database spans numerous departments and/or subdivisions and these subdivisions will express different interests in the data, leading to the need to explore various subsets of relevant relations with high utility with respect to the target class. Furthermore, acquiring such data is often expensive. Also, it becomes harder to preserve data privacy when data in a database is from multiple sources. These problems can be mitigated by pruning uninteresting relations before constructing a model. Few attempts to address this issue have been made so far [5,12].

This paper presents the Subgraph Ensemble Structure Pruning (SESP) method for pre-pruning relational databases. The SESP approach aims to create a pruned relational schema that models only the most informative substructures, while maintaining satisfactory predictive performance. This is achieved

J.N. Kok et al. (Eds.): PKDD 2007, LNAI 4702, pp. 462–470, 2007.

by removing either *irrelevant* or *redundant* substructures from the database. The SESP algorithm assumes that strongly correlated substructures contain redundant information. Those which are weakly correlated to the class are said to be of low relevance. 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 each other, but correlated with the target class (*uncorrelated subgraphs*). All other subgraphs are discarded. We compare the classifiers constructed from the original schema with those constructed from the pruned database. Our experiments on both real-world and synthetic databases demonstrate that the SESP approach is able to significantly reduce the complexity of the relational schema without sacrificing the predictive accuracy.

The paper is organized as follows. Section 2 describes the problem setting. Next, a detailed discussion of the SESP algorithm is provided in Section 3. Section 4 presents a comparative evaluation. Section 5 concludes the paper.

2 Problem Setting

The problem setting for our SESP strategy is defined by a relational database schema $\Re$, which is described by a set of tables $\{R_1,\cdots,R_n\}$. Each table R_i consists of a set of tuples T_R, a primary key, and a set of foreign keys (in this paper, we refer to primary key and foreign keys as *key attributes*). Foreign key attributes[1] 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 $R_1 \bowtie \cdots \bowtie R_n$ describes a join path, where the length of the path is defined as the number of joins it contains.

In a relational classification setting a database $\Re$ contains a target relation R_t, a set of background relations R_b, and a set of joins (J). Each tuple $x \in T_{R_t}$ includes a unique primary key attribute $x.k$ (*tuple identifier*) and a categorical variable y. The task in this setting is to find a function $F(x)$ which maps each tuple x of the target table R_t to a category label y. That is, $y = F(x, R_t, R_b, J)$.

The SESP method aims to pre-prune the relational structure for the task of multirelational classification without loss of predictive accuracy. The SESP pruning strategy is closely related to our Multiple View Relational Classification (MRC) method presented by Guo and Viktor in [4]. The MRC method aims at constructing an accurate relational classifier directly from a relational database by using multiple views on the data.

3 The SESP Pruning Approach

The core idea 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: 1) to initially decompose the relational database schema into subgraphs (*subgraph construction*); and 2) to identify the subset of strongly uncorrelated subgraphs (*subgraph evaluation*).

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

Algorithm 1. The SESP Pruning Approach

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

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

3.1 Subgraph Construction

The *subgraph construction* process aims to build a set of subgraphs given a relational database schema where each subgraph corresponds to a unique join path. The construction process initially converts the relational database schema into an undirected graph, using the tables as the nodes and joins as edges.

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 construct a classification model (details to be discussed in Section 3.2). The second constraint is for relations to be unique for each candidate subgraph. 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 [7]. Also, join paths with many relations may decrease the number of entities related to the target tuples. Therefore, we propose to this restriction for the SESP algorithm as a tradeoff between accuracy and efficiency.

Using these constraints, the subgraph construction process proceeds initially by finding unique join paths with two relations, i.e. join paths with a length of one. These join paths are progressively lengthened, one relation at a time. 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 [13]. 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.

3.2 Subgraph Evaluation

In order to select which subgraphs will be best for classification, subgraphs are evaluated according to the following methodology (Algorithm 1). Firstly, each of the created subgraphs is used to construct a separate classifier (*subgraph-based relational classification*). Secondly, the constructed classification models are used to generate an evaluation data set in which each instance is described by sets of feature sets (*subgraph features*). Each feature set describes the knowledge

contained in one of the classification models. Thirdly, the set of *subgraph features* which are strongly uncorrelated with each other, but correlated with the target class, are identified. These subgraph feature sets *correspond*, therefore, to a set of subgraphs. Lastly, subgraphs which are not selected are discarded.

Subgraph-based Relational Classification. Each subgraph created in Section 3.1 can be used to build a relational classifier using traditional single-table learning algorithms. 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. Often, different aggregation functions are employed on different types of attributes. Here, for a *Nominal Attribute*, the aggregation *COUNT* function is applied to calculate the number of times the attribute occurs within the data set. For a *Binary Attribute* for which there are only two possible values, the *COUNT* function is applied separately on each of the two values. For a *Numeric Attribute*, the aggregation functions *SUM, AVG, MIN, MAX, STDDEV and COUNT* are applied. In this way, each subgraph separately creates a set of attribute-based training instances.

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

As an example, consider a two class (y_1 and y_2) problem which has two subgraph-based classifiers C^1 and C^2 constructed. For each tuple t in the evaluation data set, each classifier C^i assigns for each class y_i a prediction p^i_i. For each such prediction a new attribute is thus generated, which we refer to as a subgraph feature. In our example, for the subgraph corresponding to classifier C^1, each evaluation example contains two new attributes (subgraph features) p^1_1 and p^1_2. Similarly, two subgraph features p^2_1 and p^2_2 are generated for the subgraph corresponding to classifier C^2. This evaluation example also contains the original class label of the tuple t.

Correlation Measurement of Subgraphs. The SESP strategy uses a heuristic measure to calculate the correlation score of a set of subgraphs (represented by subgraph features). This measure considers the correlation information both between subgraphs and between those subgraphs and the class to be learned. A similar heuristic principle has previously been applied in test theory

by Ghiselli [3] and in feature selection by Hall [6]. This heuristic assigns each subset of features a level of "goodness." The score is formalized by Equation 1 [6]:

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

Here, K is the number of features in the subset, $\overline{R_{cf}}$ is the average feature-to-class correlation, and $\overline{R_{ff}}$ represents the average feature-to-feature dependence.

To measure the degree of correlation between features and the target class and between the features themselves, we use the notion of *Symmetrical Uncertainty* ($\mathcal{U}$) [10] to calculate $\overline{R_{cf}}$ and $\overline{R_{ff}}$. This score is a variation of the *Information Gain* (*InfoGain*) measure [11]. It compensates for *InfoGain*'s bias toward attributes with more values, and has been successfully applied by Ghiselli [10] and Hall [6]. *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 $H(X)$ and $H(Y)$ are the entropies of the random variables X and Y, perspectively. Note that, these measures need all of the features to be nominal, so numeric features are first discretized properly.

Subgraph Pruning. In order to identify a set of uncorrelated subgraphs, the evaluation procedure searches all of the possible subgraph feature 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 subgraph feature space, the SESP method uses a best first search strategy [8]. The method starts with an empty set of features, and keeps expanding, one feature at a time. In each round of the expansion, the best feature 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).

Subgraphs are selected based on the final best subset of subgraph features. If a subgraph has no features that are strongly correlated to the class, the knowledge possessed by this subgraph can be said to be unimportant for the task at hand. Thus, it makes sense to prune this subgraph. The SESP algorithm, therefore, keeps a subgraph *if* and *only if* any of its subgraph features appears in the final best ranking subset.

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.

We perform our experiments using the MRC, RelAggs [9], TILDE [2], and CrossMine algorithms. The MRC and RelAggs approaches are aggregation-based algorithms where C4.5 decision trees [11] 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 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. Also, C4.5 decision tree was applied as the subgraph-based classifiers of the SESP strategy. All experiments were conducted using ten-fold cross validation. We report the average running time of each fold (run on a 3 GHz Pentium4 PC with 1 GByte of RAM).

A) Real Databases

Financial Database: Our first experiment uses the financial database from the PKDD 1999 discovery challenge [1]. This database consists of eight tables. This database provides us with two different learning problems. Our first learning task (F234) is to learn if a loan is good or bad from the 234 finished tuples. The second experimental task (F400) uses the Financial database as prepared in [13], which has 400 examples in the target table.

ECML98 Database: Our second experiment uses the database from the ECML 1998 Sisyphus Workshop. The learning task (ECML) is to categorize the 7,329 households into classes 1 and 2 [9]. Eight background relations are provided for this learning task. We here used the new star schema prepared in [9].

Experimental Results and Discussion: The predictive accuracy we obtained, using MRC, RelAggs, TILDE, and CrossMine is presented in Table 1. The results obtained with the respective original and pruned schemas are shown side by side. We also present the compression rates achieved by the SESP approach in the last column of Table 1. The compression rate considers the number of relations of the original schema ($N_{original}$) and the number of relations pruned (N_{pruned}) and is calculated as $(N_{original} - N_{pruned})/N_{original}$. In table 2, we also provide the execution time of the pruning process, as well as the running time required for the four tested algorithms against the original and pruned schemas.

From Table 1, one can see that the SESP algorithm not only significantly reduces the size of the relational schema, but also produces compact pruned schemas that provide comparable multi-relational classification models in terms of the accuracy obtained. The results shown in Table 1 provide us with two

Table 1. Accuracies obtained using methods MRC, RelAggs, TILDE, and CrossMine against the original and pruned schemas, along with the compression rate

Schema	MRC		RelAggs		TILDE		CrossMine		Compress.
	Original	Pruned	Original	Pruned	Original	Pruned	Original	Pruned	Rate
F400	88.0 %	88.0 %	89.0 %	86.8 %	81.3 %	81.0%	85.8 %	87.3 %	50.0 %
F234	92.3 %	92.3 %	90.2 %	90.2 %	86.8 %	86.8%	88.0 %	89.4 %	25.0 %
ECML	88.2 %	87.5 %	88.0 %	86.2 %	53.7 %	52.0%	85.3 %	83.7 %	55.5 %

Table 2. Execution time (seconds) required using the four tested methods against the original and pruned schemas, along with the computational time of the SESP method

Schema	MRC		RelAggs		TILDE		CrossMine		Pruning
	Original	Pruned	Original	Pruned	Original	Pruned	Original	Pruned	Time
F400	2.83	2.25	60.00	51.83	650.00	132.32	8.10	6.76	1.97
F234	1.60	1.17	40.80	34.13	568.30	80.36	5.00	3.41	1.07
ECML	424.43	220.99	1703.58	1206.39	1108.60	167.76	570.90	366.78	356.24

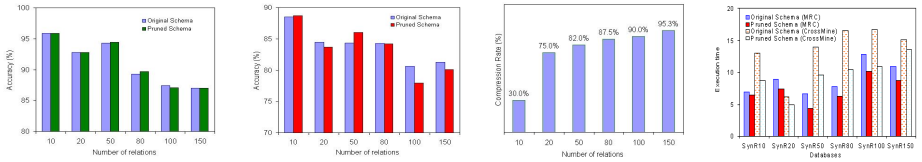
meaningful observations. The first is that the SESP strategy is capable of pruning the database schemas meaningfully. The compression rates for these three learning schemas are 50%, 25%, and 55.5%, respectively. The second finding is that the pruned schemas produce comparable accuracies, when compared to the results obtained with the original schemas. Results as obtained by the aggregation-based methods show that, for two of the three databases (F400 and F234), the MRC algorithm obtained the same predictive results when pruned. Only against the ECML database, did the pruned MRC algorithm obtain a slightly lower accuracy than the original (lower by only 0.7%). When considering the RelAggs algorithm, the accuracy produced by the RelAggs method against both the pruned and full schemas were comparable. Against the F234 data set, the RelAggs algorithm achieved the same predictive results. Only against the F400 and ECML data sets, did the RelAggs method yield slightly lower accuracy than the original.

When testing with the logic-based strategies, Table 1 shows that, the TILDE algorithm obtained almost the same accuracy against two of the three tested data sets (F400 and F234). Only against the ECML database, did the TILDE algorithm obtain a slightly lower accuracy than the original (lower by only 1.7%). When considering the CrossMine method, the accuracy produced by this method against both the pruned and full schemas was also very close.

In terms of computational cost of the SESP method, results presented in Table 2 show that the pruning processes were fast. The fast pruning time is especially relevant when considering the time required when training all four methods against the original schemas. This result is promising, especially when considering large, complex schemas where the cost to process the entire schema is prohibitive. Also, the results indicate that meaningful execution time reductions may be achieved when building the models against the pruned schemas.

B) Synthetic Databases

To further test the SESP algorithm, we generated six synthetic databases with 10, 20, 50, 80, 100, and 150 relations (denoted as SynR10, SynR20, SynR50, SynR80, SynR100, and SynR150), respectively. The database generator was obtained from Yin et al. in [13]. For each database in this paper, we set the expected number of tuples and attributes to 1000 and 15, respectively. The accuracies obtained by the MRC and CrossMine methods are shown in Figures 1(a) and 1(b), respectively. The compression rates obtained are provided in Figure 1(c). We also provide the execution time needed using the MRC and CrossMine algorithms against the original and pruned schemas in Figure 1(d).



(a) Accuracy (MRC) (b) Accu.(CrossMine) (c) Compression rate (d) Execution time

Fig. 1. Accuracies obtained and execution time (seconds) required by the MRC and CrossMine methods, as well as compression rates achieved by the SESP

From these Figures one can again see that the SESP algorithm not only significantly reduces the complexity of the structural schemas, but also produces very comparable classification models in terms of the accuracy obtained. 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. In terms of compression rate, the results in Figure 1(c) show that the compression rates were more than 80% for databases with more than 50 relations. In addition, results as presented in Figure 1(d) show that the execution time needed for constructing relational models using the two tested algorithms was meaningful reduced when pruned.

5 Conclusion and Future Work

Multirelational data mining application usually involves a large number of relations. This paper presents a novel algorithm to pre-prune uninteresting relations of relational learning tasks. Our experiments demonstrate that the strategy is able to significantly reduce the size of the relational schema while still maintaining the accuracy of the final model. This research suggests that one can build an accurate relational classification model using only a small subset of the original schema. Our future work will include research on extending the subgraph definition to include graphs with more than one slot chain. We also aim to empirically evaluate the optimum choice of maximum subgraph length.

References

1. Berka, P.: Guide to the financial data set. In: Siebes, A., Berka, P. (eds.) PKDD2000 Discovery Challenge (2000)
2. Blockeel, H., Raedt, L.D.: Top-down induction of first-order logical decision trees. Artificial Intelligence 101(1-2), 285–297 (1998)
3. Ghiselli, E.E.: Theory of Psychological Measurement. McGrawHill Company, New York (1964)
4. Guo, H., Viktor, H.L.: Mining relational data through correlation-based multiple view validation. In: KDD '06, pp. 567–573, New York, USA (2006)

5. Habrard, A., Bernard, M., Sebban, M.: Detecting irrelevant subtrees to improve probabilistic learning from tree-structured data. Fundamenta Informaticae: Special Issue on Mining Graphs, Trees and Sequences (2005)
6. Hall, M.: Correlation-based feature selection for machine learning, Ph.D thesis, department of computer science, university of waikato, new zealand (1998)
7. Hamill, R., Martin, N.: Database support for path query functions. In: Williams, H., MacKinnon, L.M. (eds.) Key Technologies for Data Management. LNCS, vol. 3112, pp. 84–99. Springer, Heidelberg (2004)
8. Kohavi, R., John, G.H.: Wrappers for feature subset selection. Artificial Intelligence 97(1-2), 273–324 (1997)
9. Krogel, M.-A.: On Propositionalization for Knowledge Discovery in Relational Databases. PhD thesis, Otto-von-Guericke-Universität Magdeburg (2005)
10. Press, W.H., Flannery, B.P., Teukolsky, S.A., Vetterling, W.T.: Numerical recipes in C: the art of scientific computing. Cambridge University Press, Cambridge (1988)
11. Quinlan, J.R.: C4.5: programs for machine learning. Morgan Kaufmann, San Francisco (1993)
12. Singh, L., Getoor, L., Licamele, L.: Pruning social networks using structural properties and descriptive attributes. In: ICDM '05, pp. 773–776 (2005)
13. Yin, X., Han, J., Yang, J., Yu, P.S.: Crossmine: Efficient classification across multiple database relations. In: ICDE '04, Boston (2004)