Efficient Classification from Multiple Heterogeneous Databases*

Xiaoxin Yin[1] and Jiawei Han[1]

University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA.
{xyin1,hanj}@uiuc.edu

Abstract. With the fast expansion of computer networks, it is inevitable to study data mining on heterogeneous databases. In this paper we propose *MDBM*, an accurate and efficient approach for classification on multiple heterogeneous databases. We propose a regression-based method for predicting the usefulness of inter-database links that serve as bridges for information transfer, because such links are automatically detected and may or may not be useful or even valid. Because of the high cost of inter-database communication, MDBM employs a new strategy for cross-database classification, which finds and performs actions with high benefit-to-cost ratios. The experiments show that MDBM achieves high accuracy in cross-database classification, with much higher efficiency than previous approaches.

1 Introduction

The rapid growth of the number of data sources on the internet has brought great need for computation over multiple data sources, especially knowledge discovery from multiple data sources. For example, biologists need databases of genes, proteins, and microarrays in their research; a credit card company needs data from a credit bureau for building models for handling applications. Data integration approaches [5, 11] may be used to overcome the heterogeneity problem. However, perfect integration of heterogeneous data sources is a very challenging problem, and it is often impossible to migrate one whole database to another site. In contrast, distributed data mining [3, 7, 8, 12, 13] aims at discovering knowledge from a dataset that is stored at different sites. But they focus on a homogeneous dataset (a single table or a set of transactions) that is distributed to multiple sites, thus are unable to handle heterogeneous relational databases.

In this paper we study the problem of *cross-database classification*, which aims at building accurate classifiers based on multiple heterogeneous databases, because a single database often contains insufficient information for a classification task. For example, Yahoo shopping may want to build a model for predicting customers' behaviors (as in Figure 1), and thus needs important information

* The work was supported in part by the U.S. National Science Foundation NSF IIS-02-09199/IIS-03-08215. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the funding agencies.

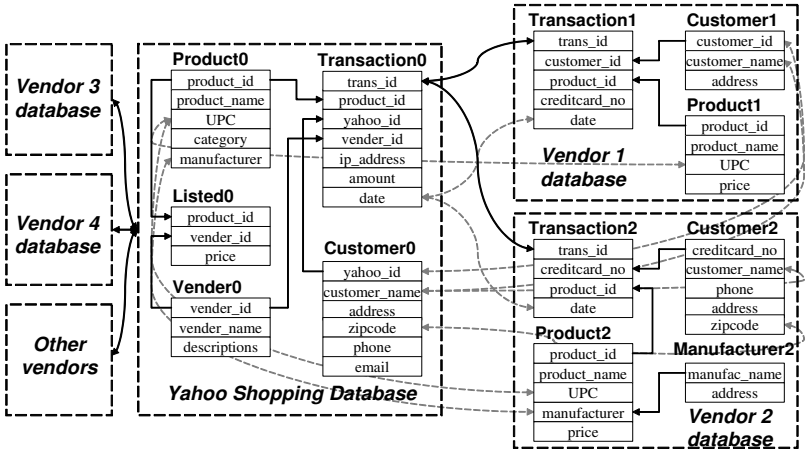

Fig. 1. Databases of Yahoo shopping and venders

from databases of different vendors. In this example the *Customer0* relation in the Yahoo shopping database is called *target relation*, whose tuples are *target tuples*. The goal of cross-database classification is to build an accurate classifier for predicting the class labels of target tuples.

There are two major challenges in cross-database classification. The first is the **data heterogeneity problem**. To transfer information across heterogeneous databases, one must detect *inter-database links*, which are links between matched attributes (as in Figure 1) and can serve as bridges for information transfer. There are many studies on this issue, such as schema mapping [5] and mining database structures [4]. However, some links detected may be vague and sometimes connect unrelated objects. For example, *Customer0.name* $\rightarrow$ *Customers1.name* may connect different persons with same name, and *Customer0.zipcode* $\rightarrow$ *Customer1.zipcode* may lead to an explosive number of joined tuples. The second challenge is the **efficiency problem**. It is often expensive to transfer information between two databases, which may be far from each other physically. Thus we must be able to build accurate cross-database classifiers with as low inter-database communication cost as possible. In this paper we propose *MDBM* (Multi-Database Miner), an efficient and accurate approach for classification across multiple heterogeneous databases.

The first contribution of this paper is to propose an approach for predicting the usefulness of links. As mentioned above, some links can lead to useful features, while some others may be useless and only add burdens to the classification procedure. We define the usefulness of a link as the maximum information gain of any feature generated by propagating information through this link. We propose a regression-based approach for building a model to predict usefulness of links based on properties of links. Our experiments show that this approach achieves reasonably high prediction accuracy.

Our second contribution is *economical classification*. As many approaches on relational (or first-order) classification [1, 9, 10, 14], MDBM also uses rule-based classification. All previous approaches build rules by searching for predicates (or literals) with highest information gain (or Foil gain), in order to build accurate

rules. Although this strategy is effective in single databases, it may lead to high inter-DB communication cost in multi-database classification. With the prediction model for gainfulness of links, MDBM can predict the gain and cost of each action of searching for predicates. The strategy of *economical classification* always selects the action with highest gain-to-cost ratio, i.e., the action of lowest price per unit of gain. It can achieve same total gain with much lower cost. Our experiments show that MDBM achieves as high accuracy as previous approaches, but is much more efficient in both running time and inter-DB communication.

The rest of the paper is organized as follows. We discuss related work in Section 2. Section 3 describes the approach for building prediction models for usefulness of links. We describe the strategy of economical cross-database classification in Section 4. We present empirical evaluation in Section 5, and conclude this study in Section 6.

2 Related Work

The traditional way of mining multiple databases is to first integrate the databases [5, 11], then apply data mining algorithms. However, it is often hard to integrate heterogeneous databases or to migrate one whole database to another site, because of both efficiency and privacy concerns. Thus in multi-database mining we need efficient approaches that can produce good mining results with low inter-database communication cost.

Distributed data mining received much attention in the last several years, which aims at discovering knowledge from a dataset that is distributed at different sites. There are two types of distributed data: (1) horizontally partitioned data, in which data about different objects with same attributes are owned by different sites; (2) vertically partitioned data, in which different attributes of the same set of objects are stored at different sites. Either way of distribution divides the rows or columns of a table into different parts. Distributed data mining approaches for horizontally partitioned data include meta-learning [3] that merges models built from different sites, and privacy preserving techniques including decision tree [8] and association rule mining [7]. Those for vertically partitioned data include association rule mining [12] and k-means clustering [13]. Distributed data mining works on a well-formatted data table stored at different sites. It is fundamentally different from cross-database data mining, which works on multiple heterogeneous databases, each containing a set of interconnected relations.

There are many studies on relational (or first-order) classification [1, 9, 10, 14], which aims at building accurate classifiers in relational databases. Such algorithms search among different relations for useful predicates, by transferring information across relations. They either build rules by adding good literals (or predicates), or build decision trees recursively. Such approaches have proven to be efficient and accurate in single-database scenarios. However, in multi-database classification, they may have high inter-database communication cost, because they only focus on finding gainful literals but not on how much data needs to be transferred. MDBM follows their main philosophy of classification (rule-based, greedy classification), but adopts a new strategy called *economical classification* which can achieve as high accuracy with much lower cost.

3 Predicting Usefulness of Links

3.1 Propagating Information Across Databases

In [4] an efficient approach is proposed to identify joinable attributes in a relational database. Its main idea is to compute the set resemblance of sets of values of different attributes, and it achieves good scalability with a sampling technique. MDBM uses this approach to find all joinable attributes across databases. For two attributes A_1 and A_2 in different databases, if a significant portion (at least 25%) of values of A_1 are joinable to A_2, or those of A_2 are joinable to A_1, then MDBM assumes there is a link between A_1 and A_2. This approach has the limitation that it can only detect simple links. A recent schema matching approach [5] that can detect complex links (e.g., "$firstname + lastname \rightarrow name$") can also be easily integrated into MDBM.

During cross-database mining, large amounts of data needs to be exchanged across databases frequently, and we need an approach that transfers minimum required information to enable effective data mining. In [14] an approach called *Tuple ID Propagation* is proposed, which propagates the unique IDs of target tuples and their class labels across different relations. Tuple ID Propagation is a method for virtually joining relations, and the propagated IDs can be used to identify useful features in different relations. As shown in Figure 2, the IDs can be propagated freely across different relations and databases. As shown in Figure 1, there are usually a large number of inter-database links. Some links serve as good bridges for cross-database mining, such as links of trans_id. While some other links are weak or even incorrect, such as links of zipcode and date.

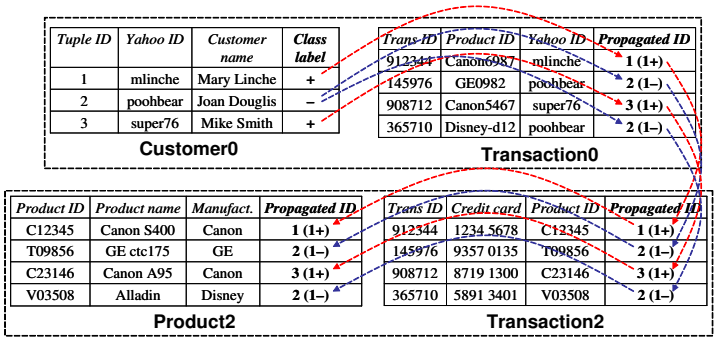

Fig. 2. Example of Tuple ID Propagation

3.2 Gainfulness of Links

As previous approaches of relational classification [1, 9, 10, 14], MDBM uses rule-based classification. A rule consists of a list of predicates and a class label. Suppose the class label is whether a customer will buy photo printers. An example rule is "[*Customer0* $\rightarrow$ *Transaction0*, *Transaction0.amount* $\geq$ 500], [*Transaction0* $\rightarrow$ *Product0*, *Product0.category*='digital camera'] $\Rightarrow$ +". It contains two predicates: the first one is that the customer buys some product of at least $500, and the second one is that this product is a digital camera. As in [10, 14], we use *Foil gain*, a variant of information gain, to measure the usefulness of a predicate.

Foil gain measures how many bits can be saved in representing class labels of positive tuples, by adding a predicate to the current rule.

Definition 1. (Foil gain). *For a rule r, we use $P(r)$ and $N(r)$ to denote the numbers of positive and negative target tuples satisfying r. We use $r+p$ to denote the rule constructed by appending predicate p to r. Suppose the current rule is $\hat{r}$. The Foil gain of predicate p is defined as*

$$Foil\_gain(p) = P(\hat{r}+p) \cdot \left[\log \frac{P(\hat{r}+p)}{P(\hat{r}+p)+N(\hat{r}+p)} - \log \frac{P(\hat{r})}{P(\hat{r})+N(\hat{r})}\right] \quad (1)$$

A link is considered to be a useful one if it brings significant Foil gain, and vice versa. To build a model for predicting the gainfulness of links, we need to first define the gainfulness of links in a predictable way. This definition must indicate the potential gain we can get from a link, but should not be significantly affected by the problem settings (e.g. usage of different classification goals) other than the properties of the link itself.

The definition of Foil gain mainly depends on two factors that vary greatly for different classification goals, even on same dataset. If there are a large number of positive target tuples, the Foil gain of each link is likely to be large. If the number of positive tuples is very small compared to that of negative tuples, then the entropy difference for each positive tuple is large, and Foil gain is likely to be large. Although these factors are not related to the links, they may affect their Foil gain greatly. Therefore, we eliminate the influences of these factors in the definition of gainfulness of links. We define the *gainfulness of a link as the maximum Foil gain we get from it, divided by the number of positive target tuples, and the maximum possible entropy gain for each positive tuple*, as follows.

Definition 2. (gainfulness of link). *Suppose there are P positive target tuples and N negative ones. Suppose p_l is the predicate with highest Foil gain that is found by propagating through link l. The gainfulness of l is defined as*

$$gainfulness(l) = \frac{Foilgain(p_l)}{P \cdot (-\log \frac{P}{P+N})} \quad (2)$$

3.3 Building Prediction Model

In order to build a model for predicting gainfulness of links, we need to select a good set of properties of links that are related to their gainfulness. The first property of a link is the type of its source and destination attributes. Each attribute can be a *key*, a *foreign-key*, or a *semi-key* (an attribute that can almost distinguish every tuple in a relation). Links between other attributes are not considered because they seldom convey strong semantic relationships.

Besides the types of links, the following three properties are selected: *coverage, fan-out, and correlation*. For a link $l = R_1.A \rightarrow R_2.B$, they are defined as follows. The *coverage* of link l is the proportion of tuples in R_1 that are joinable with R_2 via l. Propagating information through a link with high coverage is likely to generate predicates covering many positive tuples. The *fan-out* of link l is the average number of tuples in R_2 joinable with each tuple in R_1 via l.

Low fan-out usually indicates stronger relationships between linked objects. The *correlation* of link l is the maximum information gain of using any attribute of R_2 to predict the value of any attribute of R_1[1]. It indicates whether link l brings correlation between some attributes of R_1 and R_2. For example, the link *Product0.UPC* $\rightarrow$ *Product2.UPC* has high correlation because *category* of *Product0* can be predicted by *manufacturer* and some specifications of *Product2*.

The coverage, fan-out, and correlation of each link can be computed when searching for matching attributes between different databases. These properties can be roughly computed by sampling techniques in an efficient way.

Based on the properties of links, we use regression techniques to predict their gainfulness. Regression is a well studied field, with many mature approaches such as linear or non-linear regression, support vector machines, and neural networks. We finally choose neural networks [6], because it has high scalability and accuracy, and can model arbitrary functions. A neural network learns to predict values by keeping adapting itself when training examples are fed into it.

We perform multi-relational classification on some datasets to get properties and gainfulness of links, in order to get training data and build models. Our experiments show that these models achieve reasonably high accuracy when predicting for gainfulness of links on other datasets.

4 Economical Cross-Database Classification

4.1 Classification Algorithm

The procedure of rule-based classification consists of a series of actions of searching for gainful predicates. It keeps performing the following action: *propagating information across a link between two relations, and searching for good predicates based on propagated information.* For each action, there is a certain cost (of inter-database communication, computation, etc.), and a certain benefit (in predicting class labels of target tuples). The goal of economical cross-database classification is to achieve high accuracy, with as low cost as possible.

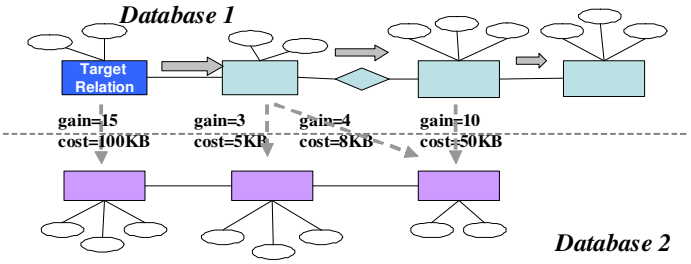

Fig. 3. Economical cross-database classification

For example, the estimated costs and benefits of four actions are shown in Figure 3. The main philosophy of economical classification is to always select the "cheapest" action, i.e., the action with highest benefit-to-cost ratio (the second one in Figure 3). In a cross-database classification process, the most gainful

[1] Numerical attributes are discretized when computing correlation.

action is usually not the cheapest one, and vice versa. By selecting the cheapest actions instead of the most gainful ones, MDBM can achieve similar total gain with a much lower "average price", thus achieves high accuracy with low cost.

In general MDBM uses the sequential covering algorithm as in [14] to build rules. At each step of searching for gainful predicates, instead of evaluating all possible predicates as in [14], it uses the strategy of economical cross-database classification to select a gainful predicate. At each step, there is a set of candidate links for propagation, each having an estimated benefit-to-cost ratio. MDBM conducts the action with highest benefit-to-cost ratio. If the real benefit of this action mostly meets our expectation, MDBM stops and moves to the next step. If the real benefit is much lower than estimation, and there is another action with higher estimated benefit-to-cost ratio, then MDBM will conduct that action.

The benefit of a propagation is defined as the maximum Foil gain of any feature found by this propagation, which can be estimated by the prediction model. Suppose there are P positive and N negative tuples satisfying the current rule. The estimated maximum Foil gain of propagation through a link l is

$$est\_Foilgain(l) = gainfulness(l) \cdot P \cdot \left(-\log \frac{P}{P+N} \right) \tag{3}$$

We use the communication overhead of a propagation as its cost, which can be estimated by the properties of link l and statistics of the source relation R_s of propagation. Suppose there are $|R_s|$ tuples in R_s, and each tuple is associated with I tuple IDs on average. [2]

$$est\_cost(l) = l.coverage \cdot |R_s| \cdot I \tag{4}$$

Now we describe the MDBM classification algorithm, which follows the main principles of previous relational classification approaches [10, 14]. MDBM builds a set of rules for each class. For a certain class, it builds rules one by one, and removes all positive tuples that are correctly classified by each rule, until more than a proportion of $(1 - \epsilon)$ of positive tuples are covered by any rule. To build a rule, it keeps searching for gainful predicates and adding them to the current rule. At each step, MDBM considers all links from the target relation or any relation used in the current rule. MDBM also utilizes some idea of beam search [15]. Suppose it builds a rule "$p_1, p_2 \Rightarrow +$", and this rule only covers a small portion of the positive tuples covered by p_1, then MDBM will try to build another rule based on those uncovered tuples satisfying p_1. By using the idea of beam search, MDBM tries to utilize all Foil gain of p_1, which saves some inter-database communication cost compared with starting from another empty rule.

4.2 Analysis of the Search Strategy

The strategy of previous rule-based classification algorithms [10, 14] is to try every possible action at each step and select the most gainful one. While our strategy is to select the cheapest action at each step. Using cheap actions will

[2] Because each propagation leads to some computational cost, the estimated cost of a propagation is set to MIN_COST if it is less than this. This threshold prevents MDBM from selecting many extremely cheap actions with very low gain.

lead to the generation of predicates and rules with less Foil gain. For our strategy to be effective, we need to prove that *many cheap actions can achieve similar classification accuracy as a smaller number of "expensive" actions, if their total gain are similar.*

Theorem 1. *Suppose a rule set S contains L rules $r_1, r_2, \ldots, r_L$. Each rule r_i covers p_i positive and n_i negative tuples, which are not covered by previous rules $(r_1, \ldots, r_{i-1})$. For another rule r that covers $(\sum_{i=1}^{L} p_i)$ positive and $(\sum_{i=1}^{L} n_i)$ negative tuples,*

$$\sum_{i=1}^{L} Foil\_gain(r_i) \leq Foil\_gain(r).$$

Corollary 1. *Suppose a rule set S contains L rules $r_1, r_2, \ldots, r_L$. Each rule r_i covers p_i positive and n_i negative tuples, which are not covered by previous rules $(r_1, \ldots, r_{i-1})$. If S has higher total Foil gain than a single rule r, then r either covers less positive tuples or more negative tuples than S.*

Theorem 1 and Corollary 1 show that, (1) if a rule set S covers identical numbers of positive and negative tuples as any single rule r, S will have less total gain, and (2) if S has total Foil gain of g, then for any single rule r with Foil gain less than g, r must cover either less positive or more negative examples than S. Although it cannot be strictly proven, we believe that in most cases if a rule set S has higher total gain than a rule r, S will have higher classification accuracy or at least cover more tuples with similar accuracy.

As mentioned before, in cross-database classification we want to achieve high classification accuracy with as low inter-database communication cost as possible. Let us compare MDBM with an existing rule-based multi-relational classification approach (e.g., [10] and [14]). MDBM always selects actions with high gain-to-cost ratios. Thus if both approaches build rule sets with similar total gains, MDBM will usually be much more efficient. On the other hand, our experiments show that MDBM achieves similar accuracies as the approach in [14], which means that MDBM probably builds a rule set with less total gain (according to Corollary 1), and is thus more efficient. The efficiency and accuracy of MDBM is also verified in our experiments.

Although MDBM usually builds more rules, it uses the same thresholds to control the complexity of each rule (by limiting the length of rule and minimum Foil gain of each predicate). Therefore, MDBM will not build overly complex rules, and overfitting is not a big concern.

5 Empirical Evaluation

We perform comprehensive experiments on both synthetic and real databases. The experiments are run on a 2.4GHz Pentium 4 PC with 1GB memory, running Windows XP Pro. The algorithms are implemented with Visual Studio.Net. The following parameters are used in MDBM: MIN_COST=0.5KB, MIN_GAIN=6.0, and $\epsilon = 0.1$. MDBM is compared with CrossMine [14], a recent approach for relational classification that is order of magnitude more efficient than previous approaches. We keep the implementation details and parameters of CrossMine,

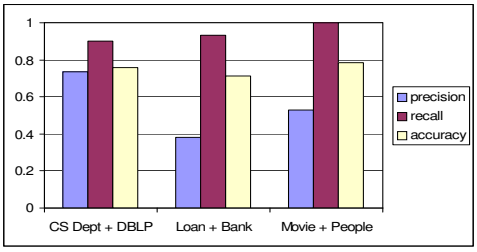

Fig. 4. Accuracy of predicting gainfulness of links

and reimplement it to make it capable of performing cross-database classification. We use the code of neural networks at
http://www-2.cs.cmu.edu/afs/cs.cmu.edu/user/mitchell/ftp/faces.html.

5.1 Experiments on Predicting Gainfulness of Links

We perform experiments on three real datasets to test the accuracy and efficiency of MDBM. The first one is CS Dept + DBLP dataset. CS Dept dataset[3] was collected from the web sources of Dept. of CS, UIUC. It contains eight relations: *Student*, *Advise*, *Professor*, *Registration*, *OpenCourse*, *Course*, *WorkIn*, and *ResearchGroup*. DBLP dataset is retrieved from DBLP web site and contains three relations: *Author*, *Publication*, and *Publish*. The target relation is *Student*, and the class labels are their research areas, which are inferred from their research groups, advisors, and recent publications.

The second dataset is Loan application + Bank dataset. This is from the financial dataset used in PKDD CUP 99, and is split into two datasets. One of them contains information about loan applications and has three relations: *Loan*, *Account*, and *District*. The other is about bank transactions and records and has five relations: *Client*, *Disposition*, *Card*, *Order*, and *Transaction*. The target relation is *Loan*. It stores the loan applications and their results (approved or not), which are used as class labels.

The third dataset is Movie + People dataset. It is from the Movies dataset in UCI KDD archive, and is split into two databases. One of them contains information of people and has three relations: *Actor*, *Director*, and *Studio*. The other contains information about movies and has four relations: *Movie*, *MovieCategory* (a movie may belong to multiple categories), *Cast*, and *Award*. The target relation is *Director* and the class label is whether a director is old or new (whether she started her career before or after 1970). All temporal information is removed from *Director* relation before training.

We first test the accuracy of predicting gainfulness of links. Cross-validation is used in this experiment as well as others, which means that a model is built based on the links from two datasets, and is used to predict the gainfulness of links in the third dataset. In this experiment a link is considered as gainful if its gainfulness is greater than 0.25. The precision, recall, and accuracy of prediction on each dataset is shown in Figure 4. Recall is more important than precision because important features may be missed if a gainful link is predicted

[3] http://dm1.cs.uiuc.edu/csuiuc_dataset/

as gainless, but it does not hurt much to predict a gainless link as gainful. It can be seen that we achieve high recall and overall accuracy for predicting gainfulness of links. This training process only takes about one second.

5.2 Experiments on Classification Accuracy

For each of the three datasets, we compare the accuracy, running time, and inter-database communication of three approaches: (1) *Single-DB CrossMine*—the CrossMine algorithm for single database; (2) *Multi-DB CrossMine*—the CrossMine algorithm that is able to propagate information and search for features across databases; (3) *MDBM*: our cross-database classification algorithm.

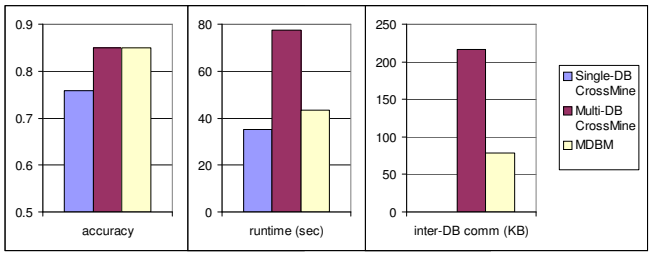

Fig. 5. Accuracy, runtime, and inter-DB communication on CS Dept + DBLP dataset

The results on CS Dept + DBLP dataset are shown in Figure 5. It can be seen that using multi-database information can significantly increase classification accuracy. MDBM achieves much higher efficiency in both running time and inter-database communication, which shows the effectiveness of our approach.

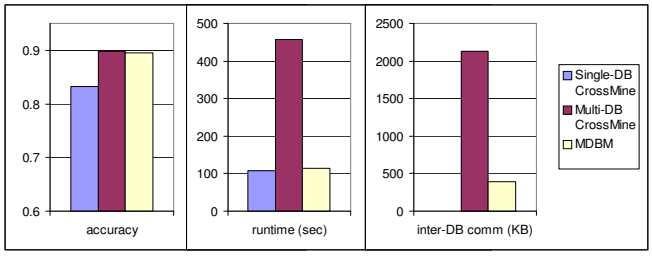

Fig. 6. Accuracy, runtime, and inter-DB communication on Loan + Bank dataset

The results on Loan application + Bank dataset are shown in Figure 6. One can see that both Multi-DB CrossMine and MDBM achieve high accuracy, and MDBM is much more efficient in inter-DB communication and running time.

The results on Movie + People dataset are shown in Figure 7. It can be seen that MDBM achieves higher accuracy than Multi-DB CrossMine. Again MDBM is much more efficient in inter-database communication (about 10% of that of Multi-DB CrossMine) and running time. Single-DB CrossMine runs fast because it cannot generate any meaningful rules.

5.3 Experiments on Scalability

We test the scalability of MDBM w.r.t. number of databases and number of tuples on synthetic datasets. We use the data generator for CrossMine [14], which

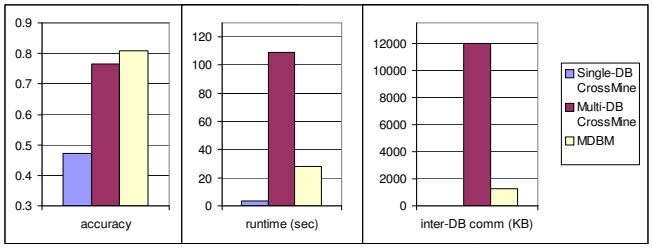

Fig. 7. Accuracy, runtime, and inter-DB communication on Movie + People dataset

can randomly generate a relational database with $|R|$ relations, each having N tuples on average. The target tuples are generated according to a set of randomly generated rules that involve different relations. After a dataset is generated, we randomly partition it into several databases, and use database structuring tool to identify inter-database links.

We first test the scalability of MDBM and Multi-DB CrossMine w.r.t. the number of databases. Five datasets are generated, with number of databases being one to five. Each database has five relations, and the expected number of tuples in each relation is 1000. The accuracy, runtime and inter-database communication of two algorithms are shown in Figure 8. It can be seen that their accuracies are close, but MDBM achieves much higher efficiency and scalability than CrossMine, especially in inter-database communication.

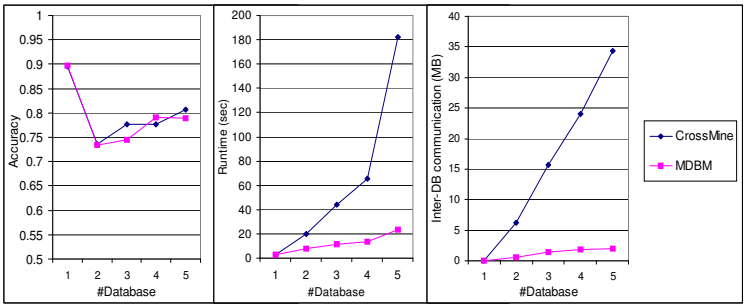

Fig. 8. Scalability w.r.t. number of databases

We also test the scalability of MDBM and Multi-DB CrossMine w.r.t. the number of tuples. Five datasets are generated with identical schemas, each having two databases and five relations in each database. The expected number of tuples in each relation grow from 200 to 5,000. The accuracy, runtime and inter-database communication of two algorithms are shown in Figure 9. It can be seen that both algorithms are linear scalable in runtime and inter-database communication, and MDBM is much more efficient than CrossMine.

6 Conclusions

In this paper we present MDBM, a new approach for cross-database classification. MDBM can perform accurate classification with data stored in multiple heterogeneous databases, with low inter-database communication. It builds a prediction model for usefulness of links from cross-database mining processes on

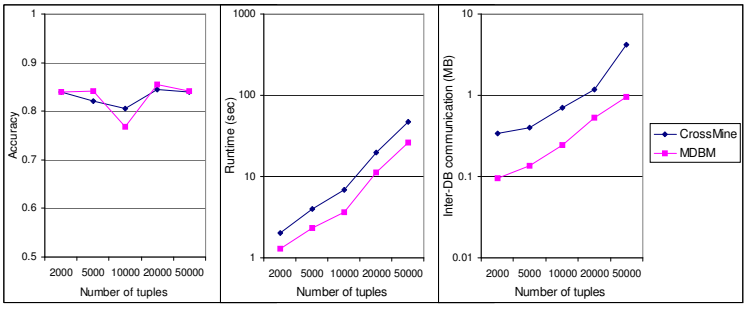

Fig. 9. Scalability w.r.t. number of tuples

available datasets that can guide the mining tasks. To achieve high classification accuracy with as low cost as possible, MDBM adopts an economical strategy for cross-database mining, which selects actions with high gain-to-cost ratio. It is shown by experiments that MDBM achieves both high accuracy and high efficiency (especially in inter-database communication) on classification tasks on both real and synthetic datasets.

References

1. H. Blockeel, L.D. Raedt. Top-down induction of logical decision trees. *Artificial Intelligence*, 1998.
2. P. Clark and R. Boswell. Rule induction with CN2: Some recent improvements. In *European Working Session on Learning*, 1991.
3. D. W. Cheung, V. T. Ng, A. W. Fu, Y. Fu. Efficient Mining of Association Rules in Distributed Databases. *TKDE*, 1996.
4. T. Dasu, T. Johnson, S. Muthukrishnan, V. Shkapenyuk. Mining Database Structure; Or, How to Build a Data Quality Browser. *SIGMOD*, 2002.
5. R. Dhamankar, Y. Lee, A. Doan, A. Halevy, P. Domingos. iMAP: Discovering Complex Semantic Matches between Database Schemas. *SIGMOD*, 2004.
6. J. Hertz, R. Palmer, A. Krogh. Introduction to the Theory of Neural Computation. Addison-Wesley, 1991.
7. M. Kantarcioglu, C. Clifton. Privacy-preserving Distributed Mining of Association Rules on Horizontally Partitioned Data. *TKDE*, 2004.
8. Y. Lindell, B. Pinkas. Privacy Preserving Data Mining. *CRYPTO*, 2000.
9. S. Muggleton. Inverse entailment and progol. In *New Generation Computing, Special issue on Inductive Logic Programming*, 1995.
10. J. R. Quinlan and R. M. Cameron-Jones. FOIL: A midterm report. In *European Conf. Machine Learning*, 1993.
11. E. Rahm, P.A. Bernstein. A Survey of Approaches to Automatic Schema Matching. *VLDB Journal*, 2001.
12. J. Vaidya, C. Clifton. Privacy Preserving Association Rule Mining in Vertically Partitioned Data. *KDD*, 2002.
13. J. Vaidya, C. Clifton. Privacy-Preserving K-Means Clustering over Vertically Partitioned Data *KDD*, 2003.
14. X. Yin, J. Han, J. Yang, P.S. Yu. CrossMine: Efficient Classification Across Multiple Database Relations. *ICDE*, 2004.
15. W. Zhang. Search techniques. *Handbook of data mining and knowledge discovery*, Oxford University Press, 2002.