SINDBAD and SiQL: An Inductive Database and Query Language in the Relational Model

Jörg Wicker, Lothar Richter, Kristina Kessler, and Stefan Kramer

Technische Universität München, Institut für Informatik I12, Boltzmannstr. 3,
D-85748 Garching b. München, Germany
{joerg.wicker,lothar.richter,stefan.kramer}@in.tum.de

Abstract. In this demonstration, we will present the concepts and an implementation of an *inductive database* – as proposed by Imielinski and Mannila – in the relational model. The goal is to support all steps of the knowledge discovery process on the basis of queries to a database system. The query language SiQL (structured inductive query language), an SQL extension, offers query primitives for feature selection, discretization, pattern mining, clustering, instance-based learning and rule induction. A prototype system processing such queries was implemented as part of the SINDBAD (structured inductive database development) project. To support the analysis of multi-relational data, we incorporated multi-relational distance measures based on set distances and recursive descent. The inclusion of rule-based classification models made it necessary to extend the data model and software architecture significantly. The prototype is applied to three different data sets: gene expression analysis, gene regulation prediction and structure-activity relationships (SARs) of small molecules.

1 Introduction

Inductive databases are databases handling data, patterns and models, and supporting the complete knowledge discovery process on the basis of inductive query languages. Many of the recent proposals for inductive databases and constraint-based data mining are restricted to single pattern domains (such as itemsets or molecular fragments) or single tasks, such as pattern discovery or decision tree induction. Although the closure property is fulfilled by many of those approaches, the possibilities of combining various techniques in multi-step and compositional data mining are rather limited. In the demonstration, we present a prototype system, SINDBAD (structured inductive database development), supporting the most basic preprocessing and data mining operations such that they can be combined more or less arbitrarily. One explicit goal of the project is to support the complete knowledge discovery process, from pre-processing to post-processing, on the basis of database queries. The research extends ideas discussed at the Dagstuhl perspectives workshop "Data Mining: The Next Generation" [1], where a system of types and signatures of data manipulation and mining operators was proposed to support compositionality in the knowledge

W. Daelemans et al. (Eds.): ECML PKDD 2008, Part II, LNAI 5212, pp. 690–694, 2008.

discovery process. One of the main ideas was to use the simplest possible signature (mapping tables onto tables) as a starting point for the exploration of more complex scenarios. The relational model was chosen, as it possesses several desirable properties, from closure to convenient handling of collections of tuples. Moreover, it is possible to take advantage of mature and optimized database technology. Finally, systems supporting (variants of) SQL are well-known and established, making it easier to get users acquainted with new querying facilities.

2 SiQL and SINDBAD: Query Language and Demonstration Overview

SiQL (structured inductive database query language), the query language of the SINDBAD system, is a straightforward extension of SQL. Instead of just adding complicated data mining operators to SQL, we focused on incorporating small, but extensible and adjustable operators that can be combined to build more complex functions. The query language supports the knowledge discovery process by offering features for the successive transformation of data. As each pre-processing and data mining operator returns a table, the queries can be nested arbitrarily, and the kind of compositionality needed in multi-step data mining can be achieved easily. The mining operators were designed in analogy to relational algebra and SQL: For instance, we made heavy use of the extend-add-as operator and devised a feature-select clause in analogy to the select clause.

From each category of preprocessing/mining algorithms, we implemented the most fundamental representatives. For discretization, we included equal frequency and equal width, for feature selection a filter approach based on information gain or variance, for pattern mining, the computation of frequent itemsets using APriori, for clustering k-Medoids, and for classification k-nearest neighbor and rule induction (pFOIL, a propositional variant of FOIL [12]). External tools can be integrated via wrappers.

We adopted the `extend` [4] operator to add the results of various data mining operations as new attributes to a given relation. It computes a function for each tuple and adds the result as the value of a new attribute. In SINDBAD, the `extend` operator adds the results of clustering, instance- or rule-based predictions, and sampling to a table. For clustering/classification, the cluster/class membership is indicated by an additional attribute. In sampling, the sample membership determined by a random number generator is given in the new attribute. In this way, we can split datasets, for instance, into a training set and a test set. For clustering and instance-based learning (k-nearest neighbor), other methods for handling tuples and distances are provided as well.

One of the central concepts of SINDBAD is that of distances between objects. This is not restricted to tuples of a single relation. Using relational distance measures, it is possible to apply clustering and instance-based learning to multi-relational data [13]. Most relational distance measures are based on recursive descent and set distances, i.e., distances between sets of points. In the simplest case, the computation of a distance between two sets of tuples A and B boils

down to computing the minimum distance between two elements of each set (single linkage), $d_{SL}(A, B) = \min_{a \in A, b \in B} d(a, b)$.

One of the most recent additions is the inclusion of full-fledged predictive models in the form of rule sets. For simplicity, we chose pFOIL, a propositional variant of the traditional FOIL algorithm [12]. The addition of models required significant extensions of the data model of the system. Models can be composed of component models. The evaluations of *component models* (e.g. class predictions) can be aggregated via *combining functions*. Combining functions can be defined in terms of logical or arithmetic operators. In this way, rule sets, weighted rule sets, trees, linear classifiers, and ensembles can be handled conveniently. For details of the query language and the implementation, we have to refer to more comprehensive publications [9,14].

In the demonstration, we will highlight some of the main features of SINDBAD in three real-world applications. In the first application, we test it on the gene expression data from Golub et al. [6], which contains the expression levels of genes from two different types of leukemia. In the second application (see tables 1,2 and 3), the task is to predict gene regulation dependent on the presence of binding sites and the state of regulators [5]. The third application is to predict anti-HIV activity for more than 40,000 small molecules [10].

Table 1. Relational schema of gene regulation data. The relation gene is the main table and connects genes with experimental setups and expression levels. The fun_cat relation gives the functional category membership of a gene according to the FunCat database. The third relation, has_tfbs indicates occurrence of transcription factor binding sites in respective genes whereas in the regulators table experimental conditions and activated regulators are given. The last table p_p_interaction gives the gene product interaction data.

```
gene(gene_id,          fun_cat(gene_id,
    cond_id,                 fun_cat_id)
    level)
has_tfbs(gene_id,      regulators(cond_id,
    yaac3_01,                ybl005w,
    yacc1_01,                ycl067c,
    yacs1_07,                ydl214c,
    yacs2_01,                ydr277c,
    ...)                     ...)
p_p_interaction(gene1_id,
    gene2_id)
```

3 Related Work and Conclusion

In the demonstration, we present a new and comprehensive approach to inductive databases in the relational model. The main contribution is a new inductive query language in the form of a SQL extension, including pre-processing and data mining operators. The approach is similar to the ones by Meo et al. [11], Imielinski and Virmani [8], and Han et al. [7] in extending SQL for

Table 2. k-Medoids for gene regulation prediction. The resulting table shows in column 2 the gene identifiers, in column 3 the experimental conditions, followed by the change of expression level and the cluster membership in column 3 and 4.

```
(30)> configure kmedoids_k = 5;
(31)> ...
(32)> extend gene add k medoid membership of gene;
(33)> show table gene;
row|gene_id|cond_id                |level|cluster|
1  |YAL003W|2.5mM DTT 120 m dtt-1  |-1   |2      |
2  |YAL005C|2.5mM DTT 180 m dtt-1  |-1   |3      |
3  |YAL005C|1.5 mM diamide (20 m)  |+1   |5      |
4  |YAL005C|1.5 mM diamide (60 m)  |+1   |1      |
5  |YAL005C|aa starv 0.5 h         |-1   |2      |
...|...    |...                    |...  |...    |
```

Table 3. k-nearest neighbor for gene regulation prediction. This resulting table, column 2 and 3 are the same as in Table 2 followed by the predicted class label in column 4.

```
(40)> configure KNearestNeighbour_K = 10;
(41)> extend gene_test add knn prediction
     > of level from gene_train;
(42)> show table gene_test;
row|gene_id|cond_id                      |class|
1  |YBL064C|aa starv 1 h                 |+1   |
2  |YDL170W|YPD 3 d ypd-2                |-1   |
3  |YER126C|Heat shock 40 minutes hs -1  |-1   |
4  |YJL109C|dtt 240 min dtt-2            |+1   |
5  |YKL180W|Nitrogen Depletion 1 d       |+1   |
...|...    |...                          |...  |
```

data mining. Another approach of Blockeel et al. [2] focuses on the automatic generation of mining views, in which the relevant parts are materialized on demand when a model is queried. The focus of the work is on the storage model and evaluation logic of data mining results. However, SINDBAD differs from related work [3] in – among other things – the support of pre-processing features. Also, it is a real prototype, useful for exploring concepts and requirements on such systems. Since it is at the moment far from clear what the requirements of a full-fledged inductive database will be, it is our belief that we can only find this out by building such prototype systems. Regarding implementations, the most similar work seems to be MS SQL Server [15]. In contrast to MS SQL Server, SINDBAD focuses on the successive transformation of data. Moreover, the approach presented here seems to be less rigid, especially with respect to feature selection, discretization, and pattern mining. Oracle uses a similar approach to provide data mining functionality in the Oracle Database (see `http://www.oracle.com/technology/documentation/datamining.html`). In future work, we are planning to investigate a more elaborate system of signatures

and types. Type signatures would be useful to define admissible inputs and outputs of data manipulation and mining operations. Signatures of operators would enable first steps towards the optimization of inductive queries.

References

1. Agrawal, R., Bollinger, T., Clifton, C.W., Dzeroski, S., Freytag, J.C., Gehrke, J., Hipp, J.: Data mining: The next generation. In: Agrawal, R., Freytag, J.-C., Ramakrishnan, R. (eds.) Report based on a Dagstuhl perspectives workshop (2005)
2. Blockeel, H., Calders, T., Fromont, E., Goethals, B., Prado, A.: Mining views: Database views for data mining. In: Proc. IEEE ICDE (2008)
3. Boulicaut, J.F., Masson, C.: Data mining query languages. In: Maimon, O., Rokach, L. (eds.) The Data Mining and Knowledge Discovery Handbook, pp. 715–727. Springer, Heidelberg (2005)
4. Date, C.J.: An Introduction to Database Systems, 4th edn. Addison-Wesley, Reading (1986)
5. Fröhler, S., Kramer, S.: Inductive logic programming for gene regulation prediction. Machine Learning 70(2-3), 225–240 (2008)
6. Golub, T.R., Slonim, D.K., Tamayo, P., Huard, P., Gaasenbeek, M., Mesirov, J.P., Coller, H., Loh, M.L., Downing, J.R., Caligiuri, M.A., Bloomfield, C.D., Lander, E.S.: Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. Science 286(5439), 531–537 (1999)
7. Han, J., Fu, Y., Wang, W., Koperski, K., Zaiane, O.: DMQL: A data mining query language for relational databases. In: SIGMOD 1996 Workshop on Research Issues in Data Mining and Knowledge Discovery (DMKD 1996), Montreal, Canada (1996)
8. Imielinski, T., Virmani, A.: MSQL: A query language for database mining. Data Min. Knowl. Discov. 3(4), 373 (1999)
9. Kramer, S., Aufschild, V., Hapfelmeier, A., Jarasch, A., Kessler, K., Reckow, S., Wicker, J., Richter, L.: Inductive databases in the relational model: The data as the bridge. In: Bonchi, F., Boulicaut, J.-F. (eds.) KDID 2005. LNCS, vol. 3933, pp. 124–138. Springer, Heidelberg (2006)
10. Kramer, S., De Raedt, L., Helma, C.: Molecular feature mining in HIV data. In: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2001), pp. 136–143 (2001)
11. Meo, R., Psaila, G., Ceri, S.: An extension to sql for mining association rules. Data Mining and Knowledge Discovery 2(2), 195–224 (1998)
12. Quinlan, J.R.: Learning logical definitions from relations. Machine Learning 5, 239 (1990)
13. Ramon, J., Bruynoogh, M.: A polynomial time computable metric between point sets. Acta Informatica 37 (2001)
14. Richter, L., Wicker, J., Kessler, K., Kramer, S.: An inductive database and query language in the relational model. In: Proceedings of the 10th International Conference on Extending Database Technology (EDBT 2008), pp. 740–744. ACM Press, New York (2008)
15. Tang, Z.H., MacLennan, J.: Data mining with SQL Server 2005. Wiley, IN (2005)