Techniques for Improving the Efficiency of Inductive Logic Programming in the Context of Data Mining

PhD Thesis by Jan Struyf
Katholieke Universiteit Leuven

Data Mining
Companies and other organizations collect vast amounts of data in the hope to obtain from these data the knowledge that would help them improve the processes that have generated the data. A pharmaceutical company could for example collect various properties about the chemical compounds they use in drugs. Such data contains implicit knowledge about the compounds, which can, once extracted, be of great value to the company. Data mining systems are able to automatically extract such useful knowledge from the available data. The extracted knowledge can be represented as rules or more general models. For example, a data mining system could extract a rule predicting that a chemical compound is active with regard to a given disease if its molecular structure includes particular features. Having such knowledge can significantly reduce the development cost of new drugs.

Inductive Logic Programming
Attribute-value data mining (AVDM) systems are frequently used in practice because they are efficient and easy to use. In AVDM, the data is represented in one single table where each instance is described by a fixed number of attributes. Many practical databases however require a relational representation with several tables. For example, to store the molecular structure of chemical compounds, one would use three tables: ’compound’, ’atom’, and ’bond’. To be able to mine relational data with AVDM, the different tables have to be summarized into one table, which may result in a loss of important information. Relational data mining (RDM) systems do not have this disadvantage as they can process relational data in its original format.

Most research concerning RDM has been situated in the field known as inductive logic programming (ILP), in which mining systems are developed that use first order predicate logic to represent the relational input data. ILP systems are more expressive, but also considerably slower than most AVDM systems. There are two main reasons for this: the hypothesis space searched for suitable models is larger in ILP than in AVDM and evaluating a single candidate model is computationally more expensive as well.

Contribution
The thesis describes and evaluates a range of techniques that make ILP systems faster and more scalable. More specifically, it considers (1) query optimization techniques, (2) techniques for optimizing data representation and access, and (3) techniques for optimizing particular mining algorithms. Although these categories are rather orthogonal, similar ideas can be applied in each one of them.

Most ILP systems build models by searching a large space of candidate models. For example, to construct a suitable rule, an ILP system searches a large space of candidate rules, e.g., using a greedy search strategy. For each rule encountered during the search, a heuristic such as accuracy is computed. Rules with a high heuristic value are refined further in subsequent search steps, while clearly sub-optimal rules are discarded. To compute the accuracy of a given candidate rule, the system has to test the condition of the rule for each instance in the database. In ILP, the condition of a rule is a conjunction of predicate calls (a query). Answering such queries is known to be NP-complete, i.e., the runtime of all exact query execution algorithms grows exponentially with the rule length. For long rules, which are for example typical for data sets storing molecules, using straightforward query execution algorithms quickly becomes computationally infeasible.

The thesis therefore evaluates several optimization techniques that speed up query execution. In a first chapter on this topic, four static query transformations are discussed: the theta, cut, once and smartcall transformation. Query transformations replace a given query by a new one that can be executed faster. E.g., by removing redundant components from the query and by avoiding unnecessary backtracking over independent components. For each transformation, both the reduction in query execution time and the transformation overhead are analyzed theoretically as well as experimentally. The thesis also investigates how the transformations can be best combined. An experimental analysis on artificial data sets with different complexity parameters shows that the combination theta + smart + once has the best overall performance.

A second chapter is devoted to a reordering transformation. This transformation places the components of a query, based on their selectivity, in the most efficient order. The chapter describes how the parameters of the transformation can be estimated, discusses a greedy approximation that works faster for long queries, and investigates how the reordering transformation can be combined with the four transformations discussed previously. An experimental evaluation on biochemical and other data sets shows that combining the trans-
formations can result in a speedup of several orders of magnitude.

A third chapter discusses multi-query optimization techniques. Such techniques optimize the execution of a set of queries sharing common subexpressions. The chapter describes how these techniques can be combined with the query transformations discussed before. This is non-trivial because the transformations may have a negative effect on the size of the common sub-expressions. In particular, the so-called query-pack multi-query technique is extended so that it can efficiently include transformed queries.

During model generation, ILP systems typically generate redundant candidate models, e.g., models that are logically equivalent to previously generated ones or models that can be proved to be sub-optimal apriori. Equivalence and subsumption tests can be used to remove such redundant candidates. However, these tests are computationally expensive and must be repeated for each candidate. The thesis therefore proposes a number of algorithms that speed up subsumption and equivalence tests by combining methods similar to the multi-query optimization techniques with hashing. An experimental evaluation shows that the proposed methods can significantly reduce the runtime of ILP pattern mining systems.

In some applications, the available data does not fit in main memory. It is then important to have (1) a space efficient representation of the data, and (2) algorithms that optimize data access on disk. The thesis proposes a new framework for representing data that has both properties. Specific to mining applications is the notion of ‘instances’ (e.g., the molecules, etc.). The ILP system will process each instance in turn while evaluating candidate models. A possible optimization therefore is to create a data block on disk for each instance including all relevant data. The blocks can then be processed one by one: load one block from disk, execute the queries on the block, remove the block from memory and proceed with the next one. A drawback is that such a representation leads to redundant storage if the blocks are not disjoint. The framework therefore extends this mechanism to include auxiliary blocks storing data that is common to several examples. It also provides methods to efficiently process data sets that store complex objects (e.g., molecules) and state based data sets (e.g., data sets occurring in agent technology). An experimental evaluation shows that the framework allows a compact representation of the data that can be mined efficiently by ILP systems.

Finally, a technique is proposed that reduces the computational overhead of cross-validation. N-fold cross-validation is a mining technique that is frequently used to estimate the predictive performance of a model. It is however computationally expensive because the mining algorithm has to be executed N times. The thesis proposes an efficient cross-validation algorithm for decision trees that is inspired on the multi-query optimization techniques discussed before and avoids repeated construction of common parts of the cross-validation trees. For some data sets, the method significantly reduces the overhead of cross-validation. It can also be extended to rule set cross-validation and to bagging.

Conclusion

Knowledge becomes increasingly important in the information age. Data mining techniques can extract such knowledge automatically from given data. Data sets with a complex, relational structure require the expressivity offered by ILP systems. This expressivity comes however at the price of an increased computational cost, resulting in some cases in excessive runtimes. Jan Struyf therefore investigates in his PhD thesis a number of techniques that are able to reduce the runtime of ILP systems. The work includes query optimization techniques, techniques optimizing data access and a case study of a technique optimizing a data mining algorithm. The experimental evaluation shows that the proposed techniques significantly reduce the execution time of ILP systems so that it becomes practically acceptable, even for complex biochemical mining tasks.

The thesis can be downloaded at:

This research was funded by the Fund for Scientific Research (FWO) of Flanders.