Using Data Mining to Support the Construction and Maintenance of Expert Systems

Geoffrey Holmes and Sally Jo Cunningham
Department of Computer Science
University of Waikato
Hamilton
New Zealand

email:
Geoffrey Holmes g.holmes@waikato.ac.nz
Sally Jo Cunningham sallyjo@waikato.ac.nz

Abstract: Many expert systems are constructed from and allied with a large collection of databases that are continually being updated. In this paper we address the issues of how a such a knowledge base can be constructed using tools that search the databases or significant, *** and present them to the knowledge engineer for review. Once a knowledge base has been constructed, there is the problem of keeping it consistent with changing conditions in the real world. Concepts which are embodied in the data may drift in time, and so some mechanism for updating the knowledge base must be developed. As it would prove too costly to periodically revisit the knowledge acquisition phase, we propose a method of monitoring of the domain database automatically for "significant" concept change.

1. Introduction

There are two general paradigms for knowledge base construction: knowledge acquisition from a human expert, supported by appropriate expertise transfer tools, and empirical induction of knowledge from collections of examples. As noted by Gaines (1991), these approaches are fundamentally related in that the examples used by machine learning programs have been carefully constructed by domain experts. While most machine learning techniques can handle a limited amount of noisy or irrelevant data, they quickly break down if given an unstructured, unselective collection of information. Further, much general or common-sense information is not stored in databases, and hence cannot be accessed by induction programs.

Expert system construction can clearly benefit from a combination of direct expert input and automated knowledge extraction. Human experts can provide structure, direction, and a common-sense understanding of the application domain. At the same time, however, the modeling rules formulated by human experts can include irrelevant and incorrect information. These biases and misconceptions can be mitigated by applying empirical *data mining* techniques to a domain database, to confirm, modify, or extend the human's domain model. These changes may then be incorporated into the expert system's knowledge base.

We use the term *data mining* to denote a shift from the machine learning paradigm. Data mining does not generally produce a knowledge base; rather, the term refers to a technique that explores ("mines") a rich, relatively unstructured set of data and retrieves from it unusual patterns, unexpected regularities, implicit information, etc. These interesting findings are generally not complete rules, but may suggest information that should be incorporated into a rule base.

Our approach to expert system construction is what Kerschberg (1992) refers to as "loosely coupled" in the sense that the database is not fully "understood" by the other components of the system. This is in contrast to an "expert deductive database system" (Schmidt 1991) which is tightly coupled to the database. Deductive databases comprise facts (database relations) and deductive rules which are general properties or constraining properties of the database relations. Such a system would, to all intents and purposes, be realised in a single program, perhaps in Prolog, with the database and rule-base working in union.

Production of an expert deductive database system from this scheme involves the separation of the deductive database (referred to as the object-level, defining the logical aspects of the problem) from the way in which the object-level is used to make deductions (referred to as the meta-level). An expert or an expert in conjunction with a knowledge engineer would produce the object-level of the system. The meta-level would be produced by the knowledge engineer without reference to the expert since this part is less application dependent and only affects the efficiency of the deduction process.

Figure 1 captures our model of the relationship between data mining and expert system construction. The construction process begins with an initial set of data, chosen by the human expert. As the expert system is constructed, the human expert may alter the composition of the databases by directing that additional or different data be collected. The human expert creates the knowledge base both through direct knowledge transfer and through mining one or more databases. Interactively guided by the expert, the mining tool efficiently organizes the search for findings in the database. These finding are then evaluated for inclusion in the knowledge base.

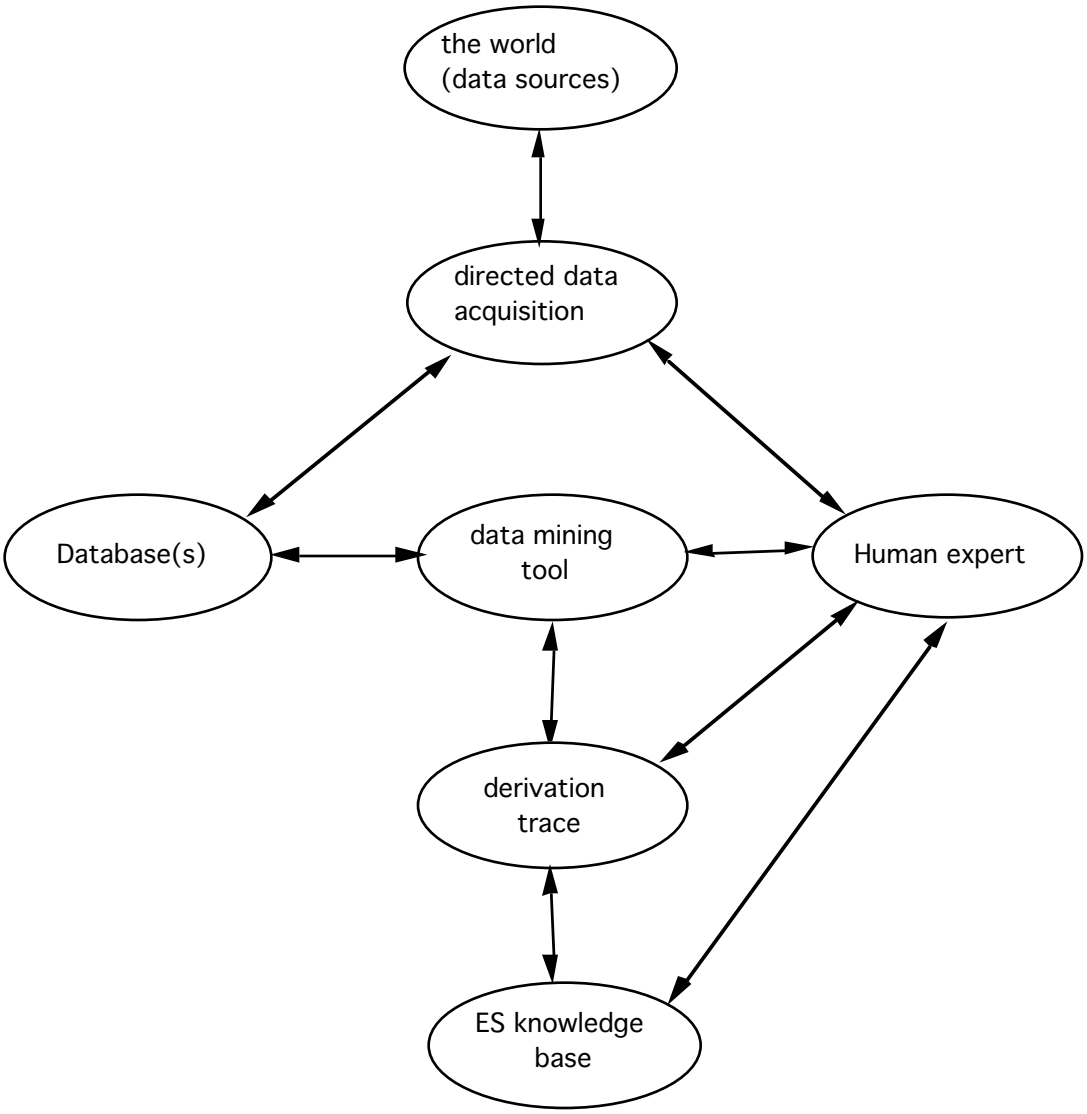

Figure 1. Architecture for using data mining to support expert system construction

The world, however, is not static; new types of data are acquired and stored, and old data is deleted or updated. These changes in the databases used to construct the expert system should be reflected by updates to the knowledge base (see Section 3.2). We accomplish this updating by maintaining a link between each rule in the knowledge base and the portion of the database that supports it (if any). The "derivation trace" stores the set of all such links. Significant changes in the data values that support a rule will signal that the rule should be re-evaluated by the human expert.

We do not attempt to update the knowledge base automatically. In a classic machine learning or deductive database approach, a carefully crafted database is tightly coupled to the knowledge base so that all information in the knowledge base is directly abstracted from the database. In this model, changes in the data can and indeed must be automatically reflected in the knowledge base.

In our model, one or more relatively unstructured, heterogenous databases are loosely coupled to a knowledge base. Changes in the underlying databases may only *potentially* require a corresponding modification of the knowledge base, since the changes may generate information irrelevant to the knowledge base. In addition, the information extracted from the database may have been augmented or modified by the human expert, and thus may not be amenable to automated modification in the knowledge base.

We illustrate the potential of this model through a case study utilizing the data mining tool "Explora", developed at the German National Research Center for Computer Science (Klosgen 1992, Klosgen 1993). The next section briefly describes this tool. Section 3 discusses knowledge base construction using it and addresses problems in knowledge base maintenance. Specifically, we discuss the importance of maintaining links between knowledge base rules and raw data, and examine an approach to determining when a "significant" change in the raw data occurs. Section 4 presents our conclusions.

2. The Explora data mining system

Database mining involves the extraction of implicit, "interesting" information from a database (Frawley et. al, 1991). The extraction process should be automated or semi-automated, and should support a high-level presentation of the information content of the data. Search through the space of candidate hypotheses can be made more efficient by the application of domain knowledge and statistical measurements of the database.

The Explora system is a statistically-based mining program described in detail in Klosgen (1992, 1993). Briefly, it identifies patterns of attribute values and value combinations that occur more or less frequently in the database than is expected. These selected patterns describe the dependency between the dependent and independent variables in a mining query. As an example: given a database of employee information, we may wish to determine how the age, sex, educational level, and race (all independent variables) impact an employee's salary (the dependent variable). The system may respond with relatively simple patterns such as "salaries are higher than average for males", or more complex pattern based on a logical conjunction of attributes and ranges, such as "salaries for college-educated employees are higher if the employee is a member of clerical staff and over the age of 40".

Standard statistical packages force the user to specify a hypothesis, select a statistical method to apply, and evaluate the measures produced. In contrast, Explora performs much of this process autonomously. Given a query (e.g., "find factors that tend to lead to high employee salaries"), it constructs an initial search space of hypotheses (candidate factors that may influence salary), selects appropriate tests for verification of these hypotheses, and evaluates the results of these tests.

Explora must search this large initial space of all potential attribute/range combinations for significant factors. The user provides domain knowledge that is used to prune this search space. Two types of domain description aid the pruning process: the aggregation of continuous variable values into meaningful clusters, such as grouping a continuous range of salaries into the categories high, middle, and low income; and the formation of taxonomies of variable values, expressing relationships such as "residents of Auckland are a subset of the residents of the North Island, which are a subset of residents of New Zealand". These clusters and taxonomies permit a partial ordering of patterns: e.g., the group "high-income people on the North Island" is more general than the group "high-income people in Auckland".

Redundancy filters are applied to the partial ordering to eliminate uninteresting portions of the statement space. There are two types[1]:

specificity filter: If a given factor is found to be significant or interesting, then prune away portions of the search space containing more specific factors.

For example: If employee salaries are higher for the general factor "male employee", then more specific factors (such as "clerical males" or "young males") are discarded as unlikely to prove interesting.

generality filter: If a given factor is found to be significant or interesting, then prune away portions of the search space containing more general factors.

For example: If more than 80% of the people with high salaries are men from Auckland, then this over-representation must hold true for men from the North Island and men from New Zealand. These more general areas of the search space may then be pruned.

Obviously the selection of a redundancy filter depends on the semantics of the domain, and the appropriate filter (if any) must be specified by the user.

3. Knowledge base construction and maintenance

The following discussion continues the example introduced in Section 2: we wish to construct an expert system that will predict an employee's expected salary range, given a description of the employee's work history, age, sex, etc. A database of information on current employees is available for mining.

3.1 Knowledge base construction

We take as our model an expert system that is constructed (in part) from concepts derived from one or more databases. The knowledge engineer guides the mining of these databases, and incorporates the concepts located into the expert system. This model differs from the classic machine learning model of expert system construction, in which the expert system is automatically derived from a single database. Our model is advantageous when there is no single database containing all information needed to construct the knowledge base, and when common sense or other unstored information must be added to the system.

We use data mining to suggest information for possible inclusion in rules. The data may yield rules suitable for direct inclusion in the knowledge base. For example, if we wish to determine the employment categories currently receiving the highest salaries, the rule "employees holding the rank of vice-president or above receive the highest salary" can be directly extracted from the database. In some cases the information extracted must be augmented with additional knowledge held by the expert, or combined with information mined from a different database. For example, we may extract the information that "graduates of the University of Texas receive higher than average salaries". The knowledge engineer may possess additional knowledge that it is the specialized training available at this university that provides a professional advantage, and that other universities are beginning to offer this training. A more general rule would be formed, then including a test for this training rather than a test for a University of Texas degree.

---

[1]The specificity and generality filters are referred to by Klosgen [1992, 1993] as probabilistic classification filter and strong necessary filters, respectively.

Alongside useful information, the data mining technique may also capture irrelevant information. One piece of information that can be mined from an employee database is that no one under the age of 16 receives a salary. A common sense understanding of the world allows a knowledge engineer to discard this fact, since the child labour laws guarantee that the company will never have an employee under 16, much less an under-aged employee earning a high salary.

3.2 Knowledge base maintenance

After the expert system is constructed, the employee database will continue to be updated, and as a consequence the knowledge base may have to be refined (Figure 2). These changes to the database may be insignificant, or they may indicate major shifts in the high-level information implicit in the data. A link must be maintained between the portions of the database used to construct a rule, the Explora query that extracted the information relevant to the rule, and the rule itself. We currently adopt a naive approach to maintenance: data updates are monitored, and when it appears likely that "significant" changes have occurred, the Explora query is re-executed. The current query results are compared with previous results (stored in a rule derivation trace). If the results differ, then the rule and the derivation trace are presented to the knowledge engineer to determine whether the knowledge base should be modified. The following discussion illustrates the problems inherent in maintenance under a loosely-coupled architecture. Further work is necessary to provide a more principled method of supporting knowledge base maintenance.

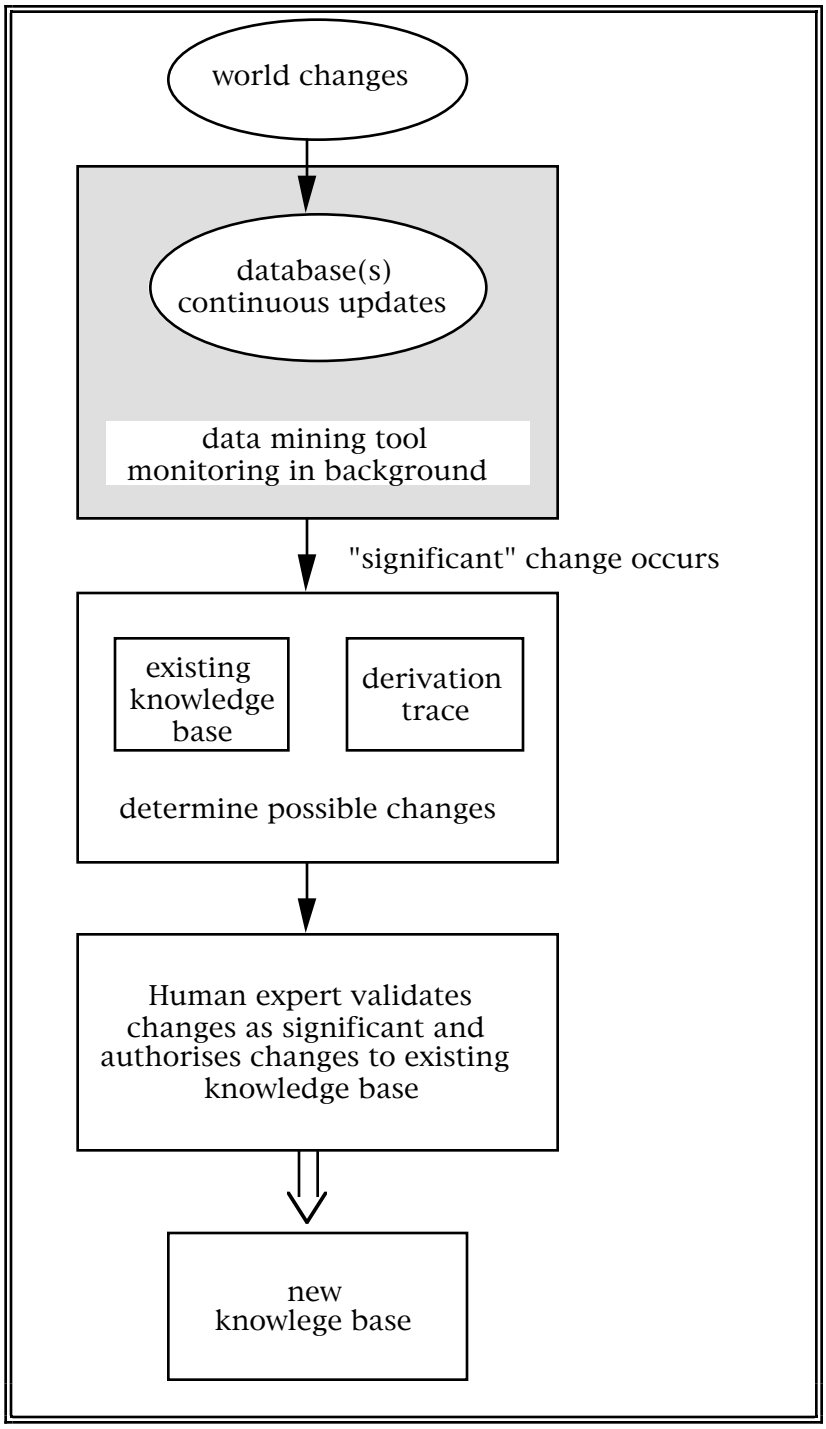

Figure 2. Knowledge base maintenance

The following example illustrates these concepts:

Our initial query to Explora is, “what factors indicate that the employee will receive a high salary?”, and the system’s response is that “males with a high school education receive the highest salaries”. We add the rule

male and high school education --> high salary

to the knowledge base. As the database is updated, changes in the data should be reflected in the rule base: our existing rule may need to be modified, for example by extending the range of education to encompass both high school and university; if the subpopulation supporting the rule is eliminated (i.e., if the employees forming the basis for the rule leave the company), then the rule may need to be deleted; and shifts in the employee database may indicate that a new, unrelated rule should be created (for example, that clerical workers on the North Island have low salaries). We do not address the problem of directly detecting the emergence of new rules, except in the general sense of monitoring the database for broad population shifts that indicate that a closer inspection of the data is necessary. Instead, we concentrate on the problem of tracking changes in the database that might lead to our rule no longer holding true. Our technique gives information that the knowledge engineer can use to modify the rule, but does not explore these possibilities autonomously.

, and tag those tuples in the employee database that have a salary in the range we have denoted to be "high". This tag is dynamically applied; new tuples with high salaries are tagged as well, and employees whose salaries fall have their tag removed.

How can our system recognize that it is likely that this rule may require modification? At present, we adopt the simple approach of storing a count of the initial number of tuples tagged as supporting the rule, and count the number of tagged tuple updates (modifications to any attribute in the tuple, tuple deletions, and tuple insertions). When the update count reaches a user-specified percentage of the initial tuple, count, then we re-run the original Explora query. If the search results differ, then this difference may indicate a need to update the knowledge base.

Different query results may occur because:

- erroneous data in the database is corrected. Suppose that a data entry error caused many female employees to be labelled as male. After the error has been corrected, a new run of the original query determines that the factor determining high salary is "high school education" alone. This indicates that the original rule should be modified as:
  high school education --> high salary

- formerly under-represented subpopulations grow. As an example, members of the subpopulation "age > 40 and a clerical worker" also tend to have high salaries. But because so few members of this population existed in the original database, Explora did not report this group as a statistically significant factor in predicting a high salary. If new tuples are added that are members of this sub-population, then Explora will detect this factor. In this case, the knowledge engineer would add an additional rule to the knowledge base:
  age > 40 and clerical worker --> high salary

- data changes reflect "concept drifts" in the world. Concepts implicit in a database will slowly change as the data itself changes, and this change must be reflected back into the knowledge base (Utgoff 1989). As an example, the past few years have seen an inflation in the qualifications required for many jobs. New employees are likely to have undergraduate or graduate degrees. As these new gain upper-level positions in the company, a new population description will emerge:
  university education --> high salary

This approach provides some solution to the update problem for the section of the rule base constructed via the Explora system. It produces a warning to the knowledge engineer that some of the knowledge has altered, who then determines whether to change the knowledge base to accommodate these changes in the data. Sending a signal in this way seems more reasonable than automatically updating the rule base. Consider a rule formed from only a few data points, for example, that all males of a certain minority earn a low salary. A new data point is entered into the

database with an erroneous salary field value that is greatly in excess of the correct figure. Let us suppose that this value is large enough to trigger a change in the rule

minority X and male --> low salary

to

minority X and male --> average salary

In this circumstance, the system we propose would probably trigger an investigation into the database to see why the situation has changed so radically.

4. Conclusion

It is attractive to loosely couple a set of databases with the knowledge base of an expert system. Information in the databases can support expert system construction by augmenting, confirming, or denying the domain model of the knowledge engineer. Since these databases may be large, complex, and relatively unstructured, data mining tools appear promising as aids to extracting implicit information from the databases. Further, as time passes the implicit information in the databases may be expected to diverge from the explicit knowledge in the expert system: erroneous values may be corrected, patchy ranges of variable values may be filled in, and concepts represented in the database may "drift" to reflect changing conditions in the world. We have described a model for maintaining links between portions of the database and the knowledge base rules these portions support. As the database is updated, these links are examined to determine whether the knowledge base should be updated as well.

The most important area for future work is to refine the mechanism for maintaining the database/knowledge base links. While the current technique appears adequate, experimentation with thresholds for "significant" concept change, efficient storage of database/knowledge base links, adding a greater semantic content to these links, etc. is indicated. In addition, we note that our technique for linking the knowledge base and the database through the derivation trace only supports the maintenance of the expert system's *current* rules. If changes occur in the database that would support the formation of new rules, then in most cases our technique will not detect the addition of this new implicit information. This problem must also be addressed, in order to fully support expert system maintenance.

References

Frawley, William, Piatetsky-Shapiro, Gregory, and Matheus, Christopher (1991). "Knowledge discovery in databases: an overview". In Piatetsky-Shapiro and Frawley, pp. 1-25.

Gaines, Brian (1991). "The trade-off between knowledge and data in knowledge acquisition". In Piatetsky-Shapiro and Frawley, pp. 490-505.

Piatetsky-Shapiro, Gregory, and Frawley, William J., eds. (1991). *Knowledge Discovery in Databases*. Menlo Park, CA: AAAI Press.

Kerschberg, L. (1992). Foreword section, *Expert Database Systems*. Keith Jeffery editor, Academic Press.

Klosgen, Willi (1992). "Problems for knowledge discovery in databases and their treatment in the statistics interpreter EXPLORA". *International Journal for Intelligent Systems*, vol 7, no. 7, pp. 649-673.

Klosgen, Willi (1993). "Discovery in databases: A typology of patterns". Proceedings of the Fourth International Workshop on Artificial Intelligence and Statistics, pp. 289-295.

Schmidt, H.(1991). "Meta-Level Control for Deductive Database Systems", *Lecture Notes in Computer Science no. 479*, Springer-Verlag, pp. 33-46.

Utgoff, P.E. (1989). “Incremental induction of decision trees.” *Machine Learning*, vol. 4, no. 2, pp. 161-186.