Efficient Search for Strong Partial Determinations

Stefan Kramer and Bernhard Pfahringer

Abstract

Our work offers both a solution to the problem of finding functional dependencies that are distorted by noise and to the open problem of efficiently finding strong (i.e., highly compressive) partial determinations per se. Briefly, we introduce a restricted form of search for partial determinations which is based on functional dependencies. Focusing attention on solely partial determinations derivable from overfitting functional dependencies enables efficient search for strong partial determinations. Furthermore, we generalize the compression-based measure for evaluating partial determinations to n-valued attributes.

Applications to real-world data suggest that the restricted search indeed retrieves a subset of strong partial determinations in much shorter runtimes, thus showing the feasibility and usefulness of our approach.

1 Introduction

Functional dependencies [Mannila & Räihä, 1994] are a fundamental form of knowledge to be discovered in databases. In real-world databases, however, we have to face the effects of noise on functional dependencies: dependencies among attributes that would have been functional without noise are likely to have exceptions. Consequently, algorithms for inferring functional dependencies would not return those dependencies which are distorted by noise. More precisely and in machine learning terms, they would overfit the data, meaning that these algorithms would find too specific functional dependencies instead of the ones we would like to find.

In contrast to functional dependencies, *partial determinations* ([Russell, 1989], [Pfahringer & Kramer, 1995]) or *approximate functional dependencies* allow for exceptions. Partial determinations may reflect probabilistic dependencies among attributes, but they may also be "impure" functional dependencies, i.e. functional dependencies which are distorted by noise and have only a few exceptions. In this paper, as in [Pfahringer & Kramer, 1995], we deal with those partial determinations which help to compress a given database as much as possible. These highly compressive partial determinations will be called *strong partial determinations*.

In the next section we define and compare functional dependencies and partial determinations. In the third section we summarize the ideas from [Pfahringer & Kramer, 1995]. Subsequently, we define and explain a new compression-based measure for partial determinations ranging over n-valued attributes. Then we describe an efficient method to search for strong partial determinations. In section 6, we report on experimental results of our method in several "real-world" databases.

2 Functional Dependencies and Partial Determinations

Formally, functional dependencies ([Mannila & Räihä, 1994] and partial determinations ([Pfahringer & Kramer, 1995]) are defined as follows:

Given a relation schema (i.e. a set of attributes) R, a *functional dependency* over R is an expression $X \rightarrow Y$, where $X, Y \subseteq R$. To define the semantics of such expressions, let r be a relation (a table) over R, i.e. a set of tuples over R. We write $Dom(A)$ for the domain of an attribute $A \in R$ and $Dom(X)$ for the domain of a set of attributes $X \subseteq R$. The projection of a tuple t on a set of attributes X, denoted $t[X]$, is defined as the restriction of t on X (here $X \subseteq R$). Now $X \rightarrow Y$ holds in r, denoted $r \models X \rightarrow Y$, if all tuples $u, v \in r$ with $u[X] = v[X]$ also satisfy $u[Y] = v[Y]$.

Partial determinations are generalizations of functional dependencies. They are expressions of the form $X \rightarrow_d Y$. The index d is a number. The set of attributes X will be referred to as LHS, the left-hand side of the partial determination, and Y will be referred to as RHS, the right-hand side. In the following, we will restrict ourselves to RHSs consisting of single attributes.

Semantically, $X \rightarrow_d Y$ holds in r, denoted $r \models X \rightarrow_d Y$, if d is the *determination factor* $d(X, Y)$ as defined by [Russell, 1989]. The determination factor is the probability that two randomly chosen tuples have the same values of Y, provided they have the same values of X. Note that $d(X, Y)$ is defined without regard to a particular mapping from $Dom(X)$ to $Dom(Y)$.

Corresponding to an expression $X \rightarrow_d Y$, we define a mapping $pd_{X \rightarrow_d Y}$ in the following way. The domain of the mapping is defined by the LHS and the range is defined by the RHS of the partial determination. The tuples in r determine the mapping itself: for all tuples u that are equal under the projection X, we call the most frequently occurring $u[Y]$ the majority tuple. $pd_{X \rightarrow_d Y}$ maps tuples of $Dom(X)$ to their respective majority tuples in $Dom(Y)$. A tuple $u \in r$ is called an *exception* to the mapping $pd_{X \rightarrow_d Y}$, if $u[Y] \neq pd_{X \rightarrow_d Y}(u[X])$. Depending on the number of exceptions, the partial determination is more or less "functional".

In the following, the notion of a *partial determination* is used for both the statement $X \rightarrow_d Y$ and the corresponding function $pd_{X \rightarrow_d Y}$. This is not problematic, since there is a correspondence between the referents.

3 Compression-Based Evaluation of Partial Determinations

The basic decision is which partial determinations to search for in databases. If we are interested only in accuracy, we are likely to get overly complex partial determinations in the presence of noise: we will find partial determinations fitting the noise instead of the underlying dependencies. To avoid this, we also have to take into account how

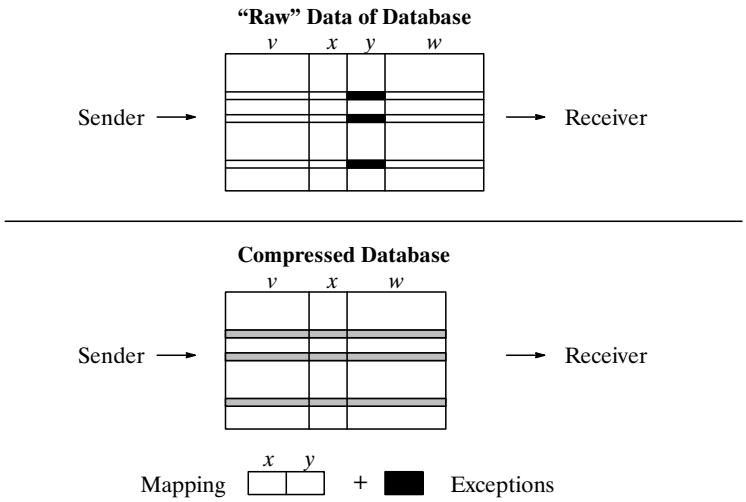

Figure 1: Transmission metaphor.

complex partial determinations are. Therefore, [Pfahringer & Kramer, 1995] proposes a compression-based measure based on the so-called *Minimum Description Length* (MDL) principle [Rissanen, 1978]. The MDL principle tries to measure both the simplicity and the accuracy of a particular theory (in our setting: a partial determination) in a common currency, namely in terms of the number of bits needed for encoding both a theory and the data given that theory. The theory with the *minimal* total message length (the most compressive partial determination) is also the *most probable* theory explaining the data [Rissanen, 1978].

To illustrate the key idea of the measure for partial determinations proposed in [Pfahringer & Kramer, 1995], we consider the hypothetical task of transmitting a given database as efficiently as possible (see fig. 1). If we can find a good partial determination $X \rightarrow_d Y$ for a given attribute Y, transmitting the partial determination instead of the raw data may improve efficiency considerably: we just have to transmit the mapping (the *model*) and transmit the information to correct the values of Y for the exceptions to the mapping $pd_{X \rightarrow_d Y}$ of $X \rightarrow_d Y$. Thus, we have to identify the exceptions (the "grey tuples" in figure 1) and transmit the corrections (the black box in figure 1). The values of Y need not be transmitted anymore. Thus we achieve some compression of the data, the degree of which is estimated by our measure based on the MDL principle.

The work reported in [Pfahringer & Kramer, 1995] has several limitations and problems: the paper only deals with partial determinations ranging over boolean attributes. Secondly, the problem of efficiently finding compressive partial determinations has not been solved. The paper compared three search methods: hill-climbing, exhaustive search up to a certain depth, and stochastic search. Implicitly, all these search methods restrict the scope of the search somehow, but the restrictions are not well-motivated. Thirdly, the method has not yet been applied to real-world databases. In the following sections, we propose extensions of our work that overcome these problems and limitations.

$$
\begin{aligned}
c(PD) &= c(Mapping) + \quad (1)\\
&\quad c(Exceptions)\\
c(Mapping) &= c_{choose}(|UsedAttrs|, |AllAttrs|) + \quad (2)\\
&\quad c_{string}(RHS\ Values\ Mapping)\\
c(Exceptions) &= c_{choose}(|Exceptions|, |AllExamples|) + \quad (3)\\
&\quad c_{string}(Corrections)\\
c_{choose}(E, N) &= N * entropy(\{E, N-E\}) \quad (4)\\
c_{string}(String) &= length(String) * entropy(char\_frequencies(String)) \quad (5)\\
&\quad + c_{max\_value\_order}(String)\\
entropy(\{F_1, \ldots, F_M\}) &= -(\sum_{i=1}^{M} plog(F_i/N)), \text{where } N = \sum_{i=1}^{M} F_i \quad (6)\\
c_{max\_value\_order}(String) &= \sum_{i=0}^{j-1} log(M-i), \quad (7)\\
&\quad \text{where } M \text{ is the cardinality of the alphabet,}\\
&\quad \text{and } j \text{ is the number of different chars in } String.\\
plog(P) &= \begin{cases} 0 & \text{if P=0} \\ P * log_2(P) & \text{otherwise} \end{cases} \quad (8)
\end{aligned}
$$

Figure 2: The definition of the coding length c of a partial determination.

4 A New Compression-Based Measure for Multi-Valued Partial Determinations

In figure 2, we define the new compression-based measure for partial determinations ranging over multi-valued attributes. The total coding length (1) is the sum of the coding length of the mapping and the coding length of the corrections to be made. The coding length for a mapping (2) is the sum of the coding length for specifying the LHS-attributes and of the coding length for the string of values for the RHS-attribute. The alphabet for this string is the domain of the RHS-attribute.

Exceptions to the mapping are encoded by (3). Whereas identifying the exceptions is enough for binary attributes, we also have to transmit the corrections for M-valued attributes, since we cannot conclude which of the remaining possible values is the correct one. Therefore, in addition to identifying the exceptions, we also have to encode a string of corrections for the identified exceptions to the mapping, where again the alphabet of this correction string is the domain of the RHS-attribute.

For estimating the cost c_{choose} of encoding the selection of E elements out of N possible elements (4) we just use the theoretical entropy-based bound provided by

[Shannon & Weaver, 1949].

Encoding a string over a multi-valued alphabet (5) is a straightforward generalization of (4). It can be thought of as a repeated binary selection encoding of what is the value occurring most frequently and which positions in the string are exceptions, i.e. have a different value. The resulting formula can be simplified to the definition (5) for c_{string}, which is just the sum of the entropy of the distribution of values in the string (6) and of an encoding of the values' identities sorted according to their frequencies (7).

5 Efficient Search for Strong Partial Determinations

Based on the view of partial determinations as functional dependencies which are distorted by noise, we propose an efficient search strategy to find strong partial determinations. Starting with overfitting functional dependencies, we propose to search for the best subset of left-hand side attributes according to our MDL-measure which avoids fitting the noise. In machine learning terminology again, we propose to *prune* the left-hand sides of overfitting functional dependencies. Pruning is achieved by a complete A*-like search over all subsets of LHS-attributes of the original functional dependency. In the following, we will refer to this two-level approach as *restricted search*. Limiting our attention to this highly interesting subset of partial determinations enables efficient search.

Even though both searching for functional dependencies and for partial determinations is exponential in the worst case, on average functional dependencies can be searched for much more efficiently due to much stronger pruning criteria. Additionally the evaluation function is much simpler: just counting the number of tuples in a projection vs. creating a mapping and computing its coding length for a projection. So in restricted search the expensive part has only to deal with a small number of attributes, namely those occurring in the left-hand side of the respective functional dependency. Furthermore, as a functional dependency is a partial determination with no exception, it also supplies an upper-bound on the coding length of possibly better partial determinations in the restricted search.

The next section will empirically show the effectiveness of restricted search in comparison to *full search*. Full search is implemented as an iterative-deepening best-first search for partial determinations. For pragmatic reasons, if the number of attributes is larger than 20, both search approaches must be limited to left-hand sides of a prespecified maximum length, which is usually 10 in our experiments.

6 Empirical Results

For a preliminary empirical comparison of both search strategies we have done experiments using various small to medium-sized "real-world" databases. `Breast cancer`, `Lymphography`, and `Mushroom` are taken from the UCI-repository. `Peace, Conflicts`, and `Attempts` are databases capturing mediation attempts in international conflicts and crises. Table 1 gives the sizes (number of attributes and number of tuples) of these databases and

Domain	# A.	# T.	# PDs Found Restr. Search	Compression Restr. Search (σ)	Compression Full Search (σ)	# Add. PDs Found Full Search	Compression Full Search Add. PDs (σ)
Br. Cancer	11	699	6	1.49 (0.78)	1.64 (0.73)	5	1.20 (0.07)
Lymph.	19	148	14	1.13 (0.09)	1.13 (0.09)	5	1.01 (0.01)
Mushroom	23	8124	7	95.16 (26.84)	100.03 (23.25)	15	5.49 (6.32)
Peace	50	1753	33	4.47 (2.76)	4.63 (2.70)	16	2.11 (0.85)
Conflicts	33	268	24	1.38 (0.47)	1.38 (0.47)	4	1.37 (0.24)
Attempts	17	1753	10	2.23 (1.00)	2.26 (0.99)	7	1.65 (0.48)

Table 1: Results in real-world domains: the meaning of columns is explained in section 6.

summarizes the search results. For the restricted search approach (pruning of overfitting functional dependencies) we list the number of attributes, for which a partial determination was found and the determinations' average compression factor. This compression factor for a given attribute is defined as the ratio between the coding length of the raw data and the coding length of the respective partial determination. Next we list the average compression factor achieved by a full search for partial determinations for the same RHS-attributes as those for which the restricted search found compressive partial determinations. Additionally we list the number of attributes for which *only the full search* was able to find compressive partial determinations, and again include their average compression factor. We can summarize as follows: for a subset of all attributes restricted search seems to find partial determinations which are on average almost as strong or compressive as those found by full search. Partial determinations missed by restricted search are on average significantly weaker (with the exception of the `Conflicts` database). We have not yet performed detailed runtime measurements, but even for the smallest database `Breast Cancer` the difference is already of two orders of magnitude (80 seconds vs. 1 hour). For the other domains the differences in the runtimes were even more drastic. Summing up, the restricted search indeed retrieves a subset of strong partial determinations in much shorter runtimes.

Inspecting the retrieved determinations for useful knowledge also provided a few insights. For instance the `Peace` database includes both a few raw numerical attributes and their respective discretizations. Clearly a discretization should be functionally dependent on the raw data, which was *not* the case in the actual database. But all those dependencies were rediscovered as partial determinations. So our method helped us discover several erroneously discretized values present in the database. Other findings included, amongst others, implicit database design conventions, and redundancies due to the flat table representation of what was actually structured knowledge. We discovered strong partial determinations that indeed reflect the implicit and hidden structure of the dataset.

7 Related Work

Functional dependencies [Mannila & Räihä, 1994] are essentially relational and do not allow for the possibility of exceptions. On the contrary, *association rules* [Agrawal & Srikant, 1994] not only allow for the possibility of exceptions, but are essentially probabilistic. *Partial determinations* ([Russell, 1989], [Pfahringer & Kramer, 1995]) can be viewed as generalizations of both functional dependencies and association rules, in that they are *relational* in nature and may have *exceptions*. As we have shown in [Pfahringer & Kramer, 1995], extending the measures used for evaluating association rules, namely *support* and *confidence*, to partial determinations leads to several problems.

The term "partial determination" has been introduced by Russell ([Russell, 1989]). He also described a method for evaluating partial determinations with respect to given facts. Briefly, Russell uses sampling to estimate the proportion of tuples in the database for which the determination holds. In other words, he estimates nothing else but the accuracy, and thus the overfitting argument applies.

[Shen, 1991] describes an algorithm that searches for three kinds of regularities in a large knowledge base. One of those regularities are determinations, but they are restricted to those having only a single attribute in the left-hand side. Since Shen is not looking for more complex dependencies, there is no need to avoid overfitting the data. Furthermore, the determinations of interest are like association rules in that they have binary attributes. The algorithm generates such simple determinations and returns them if the support is bigger than the counter-support and if a statistical test suggests their significance.

[Schlimmer, 1993] proposes an algorithm that returns every "reliable" partial determination with a complexity lower than a user-defined threshold. The reliability measure is supposed to measure the "functional degree" of the map given *subsequent* data. As we have argued in [Pfahringer & Kramer, 1995], this measure does not avoid overfitting the data, since it does not have a penalty for overly complex dependencies.

[Bell & Brockhausen, 1995] hint at a simple modification of their functional dependency search algorithm to cope with noise. But their modification can only take into account the number of "errors" *in the projection*, whereas a reliable estimate would need to assess the *global* number of "errors".

[Kivinen & Mannila, 1995] contains a thorough theoretical analysis of approximate inference of data dependencies. The paper discusses three different measures of approximate functional dependencies which can be used in a sampling framework. These three functions are related to accuracy, and therefore using them in a search framework would cause overfitting.

8 Conclusion and Further Work

Our contribution is twofold: firstly, we offer a solution to the problem of finding functional dependencies that are distorted by noise. Secondly, we tackled the problem of efficiently finding strong (i.e., highly compressive) partial determinations, which has not been solved

in [Pfahringer & Kramer, 1995]. Focusing attention on a highly interesting subset of partial determinations (those which could be functional dependencies distorted by noise) enables efficient search for strong partial determinations.

Note that basically any algorithm for inferring functional dependencies could be used in the first step of our method. It is also important to note that this approach is based on assumptions about real-world data, namely that there are strict functional dependencies which are distorted by noise.

In this paper we also generalize the compression-based measure proposed in [Pfahringer & Kramer, 1995] to partial determinations ranging over n-valued attributes. Applications to real-world data suggest the usefulness of our approach.

Our approach to searching for partial determinations has several limitations: clearly, we cannot find *all* partial determinations, but only a subset. However, experiments indicate that those partial determinations which are distorted functional dependencies are among the most compressive dependencies to be found. The second problem is that if the noise level is too high, no functional dependency might exist at all. So the pruning approach would not find partial determinations in these situations.

We plan to extend our work along the following lines: our method could be applied to inductive data engineering in the context of deductive databases, similar to INDEX [Flach, 1993]. The returned partial determinations could be used to restructure the database in order to minimize the required memory. Another application of the measure would be to evaluate and compare functional dependencies. Since usually a large number of functional dependencies are returned, our measure provides a means to distinguish between reliable functional dependencies and those which are due to chance. Another challenge would be a tight integration of the search for functional dependencies and the search for partial determinations. Finally, it would be interesting to compare and combine our approach with sampling techniques as proposed in [Kivinen & Mannila, 1993] and [Kivinen & Mannila, 1995].

Acknowledgements

This research is partly sponsored by the Austrian *Fonds zur Förderung der Wissenschaftlichen Forschung (FWF)* under grant number P10489-MAT. Financial support for the Austrian Research Institute for Artificial Intelligence is provided by the Austrian Federal Ministry of Science, Research, and Arts. We would like to thank Gerhard Widmer for valuable discussions. Both the lymphography and the breast cancer domain were obtained from the University Medical Centre, Institute of Oncology, Ljubljana, Slovenia. Thanks go to M. Zwitter and M. Soklic for providing the data. The mediation databases were obtained from the Department of Political Science at the University of Canterbury, Christchurch, N.Z. Thanks go to Dr. Jacob Bercovitch.

References

[Agrawal & Srikant, 1994] Agrawal R. and Srikant R.: Fast Algorithms for Mining Association Rules. *Proceedings of the* 20^{th} *VLDB Conference*, Santiago, Chile, 1994.

[Bell & Brockhausen, 1995] Bell S., Brockhausen P.: Discovery of Data Dependencies in Relational Databases, FB Informatik, Universitaet Dortmund, LS-8 Report 14, 1995.

[Flach, 1993] Flach P.A.: Predicate Invention in Inductive Data Engineering, in Brazdil P.B.(ed.), *Machine Learning: ECML-93*, Springer, Berlin, pp.83-94, 1993.

[Kivinen & Mannila, 1993] Kivinen J., Mannila H.: The power of sampling in knowledge discovery, University of Helsinki Department of Computer Science, Series C, No. C-1993- 66, 1993.

[Kivinen & Mannila, 1995] Kivinen J., Mannila H.: Approximate dependency inference from relations, *Theoretical Computer Science*, 149(1):129-149, 1995.

[Mannila & Räihä, 1994] Mannila H. and Räihä K.-J.: Algorithms for Inferring Functional Dependencies From Relations. *Data & Knowledge Engineering* 12 (1994) 83-99, 1994.

[Pfahringer & Kramer, 1995] Pfahringer B., Kramer S.: Compression-Based Evaluation of Partial Determinations, *Proceedings of the First International Conference on Knowledge Discovery and Data Mining*, AAAI Press, 1995.

[Rissanen, 1978] J. Rissanen: Modeling by Shortest Data Description. In: *Automatica*, 14:465-471, 1978.

[Russell, 1989] Russell S.J.: *The Use of Knowledge in Analogy and Induction.* Pitman Publishing, London, 1989.

[Schlimmer, 1993] Schlimmer J.C.: Efficiently Inducing Determinations: A Complete and Systematic Search Algorithm that Uses Optimal Pruning. *Proceedings of the* 10^{th} *International Conference on Machine Learning*, 1993.

[Shannon & Weaver, 1949] Shannon C.E. and Weaver W.: *The Mathematical Theory of Communication*, University of Illinois Press, 1949.

[Shen, 1991] Shen W.-M.: Discovering Regularities from Large Knowledge Bases. *Proceedings of the* 8^{th} *International Workshop on Machine Learning*, 1991.