Learning Right Sized Belief Networks by Means of a Hybrid Methodology*

Silvia Acid and Luis M. De Campos

Departamento de Ciencias de la Computación e I.A.
E.T.S.I. Informática, Universidad de Granada
18071 - Granada, SPAIN
{acid,lci}@decsai.ugr.es

Abstract. Previous algoritms for the construction of belief networks structures from data are mainly based either on independence criteria or on scoring metrics. The aim of this paper is to present a hybrid methodology that is a combination of these two approaches, which benefits from characteristics of each one, and to introduce an operative algoritm based on this methodology. We dedicate a special attention to the problem of getting the 'right' size of the belief network induced from data, i.e. finding a trade-off between network complexity and accuracy. We propose several approaches to tackle this matter. Results of the evaluation of the algorithm on the well-known Alarm network are also presented.

1 Introduction

Graphical models such as belief networks BN [13] have become very attractive tools because of their ability to represent knowledge with uncertainty and to efficiently perform reasoning tasks. The knowledge that they manage is in the form of dependence and independence relationships, two basic notions in the human reasoning. Both relationships are coded by means of (1) a qualitative component of the model, i.e., the directed acyclic graph (dag) and (2) a collection of numerical parameters, usually conditional probabilities which measure the strength of the dependencies displayed in the model.

In recent years, many BN learning algorithms from databases have been developed, coming from different approaches and different principles. Generally, they can be grouped in two main approaches: methods based on conditional independence tests [5,15], and methods based on a scoring metric [8,10,11,12].

The algorithm we are going to describe in this paper do not fall clearly in any of these two categories but it utilizes a hybrid methodology: it uses a specific metric and a search procedure (so, it belongs to the group of methods based on scoring metrics), although it also explicitly makes use of the conditional independences embodied in the topology of the network to elaborate the scoring metric (hence it has also strong similarities with the algorithms based on independence

* This work has been supported by the Spanish Comisión Interministerial de Ciencia y Tecnología (CICYT) under Project n. TIC96-0781.

D.A. Zighed, J. Komorowski, and J. Żytkow (Eds.): PKDD 2000, LNAI 1910, pp. 309–315, 2000.

tests). The only precedents we know about hybrid learning algorithms are the works in [9,16,14]. The rest of the paper is organized as follows. In Section 2 we outline our methodology and we propose an algorithm, BENEDICT-*dsep* for recovering the graph with the restriction that the ordering between the variables is given. Section 3 discusses different approaches to get right sized models from the learning process. In Section 4, we carry out a comparative study of the results obtained by the algorithm using the well-known Alarm network. Finally, Section 5 contains the concluding remarks and some proposals for future works.

2 Learning Belief Networks with BENEDICT-*dsep*

The algorithm we are going to describe, BENEDICT-*dsep*, works under the assumption that the total ordering of the variables is known. This assumption, although somewhat restrictive, is quite frequent for learning algorithms. The algorithm is part of a family of algorithms [3,4] that share a common methodology [1] for learning belief networks, which we have called BENEDICT. The name is an acronym of *BE*lief *NE*tworks *DI*scovery using *C*ut-set *T*echniques, and is motivated by the use of d-separating sets or cut-sets to define the metric.

Let us briefly describe the BENEDICT methodology. The basic idea is to measure the discrepancies between the conditional independencies (d-separation statements) represented in any given candidate network G and the ones displayed by the database D. The lesser these discrepancies are, the better the network fits the data. The aggregation of all these (local) discrepencies will result in a measure $g(G, D)$ of global discrepancy between the network and the database.

To measure the discrepancy of each one of the independencies in the graphical model and the numerical model (the database), BENEDICT uses the Kullback-Leibler cross entropy:

$$Dep(X, Y|Z) = \sum_{\mathtt{x},\mathtt{y},\mathtt{z}} P(\mathtt{x}, \mathtt{y}, \mathtt{z}) \log \frac{P(\mathtt{x}, \mathtt{y}|\mathtt{z})}{P(\mathtt{x}|\mathtt{z})P(\mathtt{y}|\mathtt{z})},$$

where `x, y, z` denote instantiations of the sets of variables X, Y and Z respectively, and P is a probability estimated from the database.

As the number and complexity of the d-separation statements in a dag G may grow exponentially with the number of nodes, we cannot use all the d-separations displayed by G, but we have to focus on some selected subset of 'representative' d-separation statements and ignore the remainders. Given any candidate network G, BENEDICT calculates, from the database, the conditional dependence degrees of any two non-adjacent single variables, x_i and x_j (assuming that $x_j <_\theta x_i$) given the set of minimum size, $S_G(x_i, x_j)$, that d-separates x_i and x_j in G, $Dep(x_i, x_j|S_G(x_i, x_j))$. Of course, finding this set takes some additional effort, but it is compensated by a decreasing computing time of the corresponding dependence degree and more reliable results. The method BENEDICT uses for efficiently finding the sets $S_G(x_i, x_j)$ is described in detail in [2][1].

In order to give a score to a specific network structure G given a database D, BENEDICT uses the aggregation (the sum) of the local discrepancies, as

[1] In [7], approximate minimum cut-sets are used instead of exact ones.

measure of global discrepancy $g(G, D)$ (which has to be minimized). Finally, the type of search method used by BENEDICT is a simple greedy search that allows to insert into the structure the candidate arc that produces a greater improvement of the score (removal of arcs is not permitted).

Let us describe more especifically the algorithm BENEDICT-*dsep*. Initially, it starts out from a completely disconnected network. The current set L of candidate arcs to be introduced in the current network is composed by all the arcs coherent with the ordering. The scoring of this 'empty' graph G_0 is then calculated, as $g(G_0, D) = \sum_{i=2}^{n} \sum_{x_j <_l x_i} Dep(x_i, x_j|\emptyset)$. Next, the algorithm looks for the arc whose addition to the graph results in a greater decrease of $g(., D)$, thus obtaining a new graph, G_1, containing only one arc. The process continues in this way, adding at each step, k, the single arc, say $x_j \to x_i$, which verifies

$$g(G_{k-1} \cup (x_j \to x_i), D) = \min_{\substack{x_h \to x_l \in L \\ x_h \to x_l \notin G_{k-1}}} g(G_{k-1} \cup (x_h \to x_l), D).$$

In this way arcs are added in a stepwise process until a stopping rule is satisfied.

3 Stopping Rules

In the description of the algorithm BENEDICT-*dsep* we have not specified the way in which we stop the learning process, i.e., how to decide whether the size of the graph is adequate and then do not add more arcs to the current structure. The extent to which the resultant network is a good description of the domain strongly depends on this question.

For example, if we let the graph grow until no more arcs can be included (thus obtaining a complete graph), this network has a zero global discrepancy with every database, since it does not explicitly represent any independence relationship. However, this network is far to be descriptive (although it may represent any independence relationship, that remains hidden in the conditional probability tables). In general, if we stop the learning process too late the resultant networks are very complex to be understood, estimated and used for inference purposes (moreover, an overfitting phenomenon can appear). On the other hand, if we stop too early the growing process, in order to get a simpler structure, we may lose the chance to improve it (i.e., to get a better score). So, the network displays more independences than those they are really supported by the data, thus resulting in a bad approximation of the distribution underlying the data. Therefore, we have to focus on finding appropiate stopping rules, that is, on finding a criterion for determining a good trade-off between a good description of the mechanisms which generated the data, and the simplicity of the model.

The first stopping rule we tried consisted in setting a threshold and deciding not to include an arc if the global discrepancy with the data was beneath it. A second alternative, also using thresholds, was not to include an arc if no significant improvement is achieved by adding the best arc to the current graph.

After having studied the behaviour of our algorithm using the two aforementioned stopping rules [3] (and using also a combination of both rules), by

means of several databases, we get clear that these rules produced in general unsatisfactory results. The problem has two aspects: First, the threshold values are critic to the density of the learned networks. Second, it is quite difficult to determine automatically, from the data distribution, the appropriate values of these thresholds. Thus, we decided to tackle the problem of finding the optimum size of the learned network from other perspectives:

- Using independence tests to remove arcs from the set of candidate arcs, and thus avoiding to include them in the structure regardless of the value of the discrepancy reached by the current network.
- Pruning instead of stopping. Grow a network in a not very restrictive manner and then prune it by revisiting each arc (dependency) established in the building process, in order to check whether it was introduced prematurely. This is inspired by the common practise of using methods for pruning decision trees, which often produce excelent results.

3.1 Independence Tests

The basic idea for using independence tests is to consider as candidate, at every step of the learning process, only the arcs between pairs of (non-adjacent) variables which have not been found independent. In the case that the test establishes an independence relationship, then the arc linking the independent variables is deleted from the set of candidate arcs L. Therefore, the learning process will stop naturally when the set of candidate arcs becomes empty (the arcs are removed from this set either because they are inserted into the current graph or because their extreme nodes are found to be independent by the test).

In order to define the kind of test, we will take advantage of the fact that the statistic $2 * N * Dep(x_i, x_j|Z)$ is approximately χ^2-distributed with $\|Z\|(\|x_i\| - 1)(\|x_j\| - 1)$ degrees of freedom, where N is the number of samples in D, and $\|.\|$ represents the number of different states in the corresponding set of variables.

Let us see how to use the independence tests in our algorithms: At each step, once the best candidate arc $x_j \rightarrow x_i$ has been selected, it is included in the current structure G (and removed from the set of candidate arcs); then the remaining candidate arcs in L are reexamined in order to see if some may be removed because their extreme nodes are found independent by the test. The newly introduced arc changes the connectivity of the network, and can turn into independent variables that were not yet determined as independent. All these independence tests are based on values that have been already calculated, so they entail almost null additional computational cost.

3.2 The Pruning Process

The first step is to grow the network by letting the building process to work with a non very restrictive stopping criterion. In the resultant structure G, some right arcs will be introduced and also some additional (incorrect) ones. Then a review of the arcs included in the structure may reveal that some of them are superfluous and thus can be removed from the network.

An arc in G may be considered superfluous if the nodes it connects are conditionally independent (i.e., the dependence degree between the pair is very low). Perhaps this fact was not discovered before because, due to the greedy nature of the process, the d-separating set used previously to try to separate these nodes was not correct. However, the information obtained in subsequent steps (new arcs have been introduced) changes the connectivity of the network and therefore may also change the relationships between the variables. So, to verify the possible independence relationship between the linked variables, the arc is removed momentarily and the *Dep* value is calculated to be used into an independence test. If the independency is true, the arc is removed definitively, otherwise it is put back in the network.

The ordering to applying these tests is precisely the ordering in which the arcs were introduced in the structure: in this way we can reexamine the established dependences in the light of the information obtained after any arc $x_j \rightarrow x_i$ is introduced (so, the d-separating set that, at the moment in which $x_j \rightarrow x_i$ was inserted, was not able to make independent x_i and x_j, could change, thus being able to render x_i and x_j independent).

4 Experimental Results

In this section, we are going to present the results obtained by our algorithm using different databases in the learning process. All the experiments were carried out on a sun4m sparc station at 100Mhz, to reconstruct the so-called Alarm belief network. The Alarm network contains 37 variables and 46 arcs. The input data commonly used are subsets of the Alarm database, which contains 20,000 cases that were stochastically generated using the Alarm network, specifically, we use the first 500, 1000, 2000 and 3000 cases for our experiments.

The information we will show about each experiment with our algorithm is the following: *Time.*- The time, measured in minutes and seconds, spent in learning the Alarm network. $g(G, D)$.- The value of the measure of global discrepancy with respect to the database D it comes from. *Ham.*- The Hamming distance, number of different arcs (missing or added) in the learned network with respect to the original model. *N.Arc.*- Number of arcs in the learned network. *Kullback.*- The Kullback distance (cross-entropy) between the probability distribution, P, associated to the database and the probability distribution associated to the learned network, P_G, to measure how closely the probability distribution learned approximates the empirical frecuency distribution. Actually, we have calculated a decreasing monotonic linear transformation of the Kullback distance, because this one has exponential complexity and the transformation can be computed in a very efficient form [12]. The interpretation of our transformation of the Kullback distance is: the higher this parameter is the better is the network.

Our first experiment concentrates around the stopping rules, aiming to determine the quality of the different rules proposed: using thresholds, independence tests, and pruning. In the first case we use a conjunction of thresholds (low discrepancy and no significant improvement). Both thresholds were set 'ad hoc' as a fraction of the initial discrepancy calculated on the empty graph. We use the

χ^2 test for the conditional independence tests and the pruning process with a fixed confidence level of 0.99. For this experiment we used the database containing 3000 cases, and the results are displayed in table 1. From these results it is

Table 1. Comparing stopping rules: Results using BENEDICT-*dsep*.

	Time	$g(G,D)$	Kullback	Ham.	N.Arc
Thresholds	20:58	1.3754	8.2329	17	31
Indep.	7:10	0.6995	9.2213	10	50
Indep.+ pruning	7:12	0.6818	9.2036	4	44

clear that the use of independence tests improves remarkably the performance of the algorithm with respect to the use of thresholds, as much in the quality of the results as in the efficiency. They present lower discrepancy values and also higher values of the objective Kullback measure. On the other hand, we can observe that the times spent are markedly lower, than using thresholds. This gives us an idea of the savings in evaluating candidate networks obtained by removing some arcs from the candidate set L. We can also see that final process of pruning, using almost null additional time, simplifies the final structures, although slightly decreases the Kullback distances; anyway, the results are comparable. In the light of these results, in the following experiments we always will use the combination of independence tests and pruning as the stopping rule.

In the second experiment our interest is focused in evaluating the robustness of the algorithm. The results of this experiment are displayed in table 2. Several

Table 2. Using different sample sizes: Results with BENEDICT-*dsep*.

Sample Size	Time	$g(G,D)$	Kullback	Ham.	N.Arc
500	2:22	2.7586	8.9759	11	43
1000	3:49	2.4367	9.0927	9	45
2000	5:06	0.9605	9.1345	5	45
3000	7:12	0.6818	9.2036	4	44

conclusions may be drawn from this experiment. First, it seems that the learned networks approach the original one as the sample size increases, as shown by the Hamming distances and the Kullback measures. The time employed is almost linear with respect to the sample size. Specifically, for the bigger training set, the mistaken arcs are 3 missing arcs, where two of them are not strongly supported by the data. The third missing arc makes to introduce an additional arc.

5 Concluding Remarks

We have proposed a hybrid methodology for learning bayesian belief networks from databases, and an algorithm based on it. Our algorithm uses conditional independence relationships of order as low as possible, thus gaining in efficiency and reliability. The algorithm recovers reasonably well complex belief network structures from data in polynomial time, under the assumption that the ordering among the variables is known. Several possible extensions of this work, could be:

- Avoiding the requirement of an ordering on the nodes. One approach is to leave the search procedure to also cope with the directions of the arcs to be added. Another option is to perform a two-steps process: the first one is to run an algorithm for learning a good ordering [6], the second one would be carried out by our algorithm.
- Modification of the pure greedy search used, other search techniques such as branch and bound or simulated annealing, could be used in combination with our scoring metric. We could modify the search process by using arc addition as well as arc removal (or even arc reversal).

References

1. S. Acid. *Métodos de aprendizaje de redes de creencia. Aplicación a la clasificación.* PhD thesis, Universidad de Granada. Spain, 1999 (in Spanish).
2. S. Acid, L.M. de Campos. An algorithm for finding minimum d-separating sets in belief networks. in: *Proceedings of the 12th Conf. on Uncertainty in Artificial Intelligence*, 1996, 3–10.
3. S. Acid, L.M. de Campos. Benedict: An algorithm for learning probabilistic belief networks. in: *Proceedings of the 6th IPMU Conf.*, 1996, 979–984.
4. S. Acid, L.M. de Campos. A Hybrid Methodology for Learning Belief Networks: BENEDICT. *Int. Journal of Approximate Reasoning* Submitted.
5. L.M. de Campos, J.F. Huete. A new approach for learning belief networks using independence criteria. *Int. Journal of Approximate Reasoning* 24 (2000) 11–37.
6. L.M. de Campos, J.F. Huete. Approximating causal orderings for Bayesian networks using genetic algorithms and simulated annealing. in: *Proceedings of the IPMU-2000 Conf.*, to appear.
7. J. Cheng, D.A. Bell, W. Liu. An algorithm for Bayesian belief network construction from data. in: *Proceedings of the Seventh Int. Workshop on Artificial Intelligence and Statistics*, 1997, 83–90.
8. G.F. Cooper, E. Herskovits. A bayesian method for the induction of probabilistic networks from data. *Machine Learning* 9 (4) (1992) 309–348.
9. D. Dash, M.J. Druzdzel. a hybrid anytime algorithm for the construction of causal models from sparse data. in: *Proceedings of the 15th Conf. on Uncertainty in Artificial Intelligence*, 1999.
10. D.Heckerman, D.Geiger, D.M. Chickering. Learning bayesian networks: The combination of knowledge and statistical data. *Machine Learning* 20 (1995) 197–243.
11. E. Herskovits, G.F. Cooper. Kutató: An entropy-driven system for the construction of probabilistic expert systems from databases. in: P. Bonissone (Ed.), *Proceedings of the 6th Conf. on Uncertainty in Artificial Intelligence*, Cambridge, 1990, 54–62.
12. W. Lam, F. Bacchus. Learning bayesian belief networks. an approach based on the mdl principle. *Computational Intelligence* 10(4) (1994) 269–293.
13. J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann, San Mateo, 1988.
14. M. Singh, M. Valtorta. Construction of Bayesian network structures from data: A brief survey and an efficient algorithm. *International Journal of Approximate Reasoning* 12 (1995) 111–131.
15. P. Spirtes, C. Glymour, R. Scheines. *Causation, Prediction and Search.* Lecture Notes in Statistics 81. Springer Verlag, New York, 1993.
16. P. Spirtes, T. Richardson, C. Meek. Learning Bayesian networks with discrete variables from data. in: *Proceedings of the First Int. Conf. on Knowledge Discovery and Data Mining*, 1995, 294–299.