Rethinking the Learning of Belief Network Probabilities

Ron Musick*

Advanced Information Technology Program
Lawrence Livermore National Laboratory
P.O. Box 808, L-419, Livermore, CA 94551
rmusick@llnl.gov

Abstract

Belief networks are a powerful tool for knowledge discovery that provide concise, understandable probabilistic models of data. There are methods grounded in probability theory to incrementally update the relationships described by the belief network when new information is seen, to perform complex inferences over any set of variables in the data, to incorporate domain expertise and prior knowledge into the model, and to automatically learn the model from data. This paper concentrates on part of the belief network induction problem, that of learning the quantitative structure (the conditional probabilities), given the qualitative structure. In particular, the current practice of rote learning the probabilities in belief networks can be significantly improved upon. We advance the idea of applying any learning algorithm to the task of conditional probability learning in belief networks, discuss potential benefits, and show results of applying neural networks and other algorithms to a medium sized car insurance belief network. The results demonstrate from 10 to 100% improvements in model error rates over the current approaches.

Introduction

Belief networks have been accepted as a tool for knowledge discovery in databases for several years now, and have been a growing focus of machine learning research for the past decade. Several uses have been demonstrated in the literature in domains as distinct as document retrieval, medical diagnosis, and telecommunications (D'Ambrosio 1994; Ezawa & Norton 1995; Park, Han, & Choi 1995). A common need across all of these application domains is for robust, flexible and powerful methods for the automatic induction of belief networks. Along with the obvious savings in time and effort, well-constructed automated techniques often lead to improved models, and can help data analysts develop deeper insight into the processes hidden in the data by making it easier to experiment with new ideas.

The main goal of this paper is to influence the current pattern of thought on the effective induction of belief network probabilities. This paper advocates the application of standard machine learning techniques to this problem together with, or in place of, the rote learning techniques that are most common today. We do not address the task of learning the structure of the network.

The conditional probability table (CPT) of a node (variable) in a belief network stores the probabilistic relation between that variable and its parents as a table of conditional probabilities. There is one CPT per node in the network. Inducing the CPTs is a learning problem. Each is a *potentially unique* function from the parent variables to the child, and thus should be open to a wide range of learning techniques. The current approach in most all cases is to learn the conditional probability tables with a simple statistical counting method ("bookkeeping") that can be likened to the rote learning done by chess and checkers programs back in the 60's (Samuel 1963). The bookkeeping approach is seductive because it is the easiest method to implement, is very understandable, and leads to Dirichlet distributions. Dirichlets have nice theoretical properties that can lead to effective measurements of accuracy during inference (Musick 1993). However, the power and flexibility of being able to apply any machine learning technique to CPT learning has advantages that can not be ignored. The following is a brief argument for *incorporating* machine learning techniques into the statistically oriented techniques that are currently in force. The rest of the paper backs these arguments up with results of an implementation of these concepts.

- **Unsupervised Generalization**: Generalization is the heart of the ability to learn and discover new knowledge. It can be argued that most machine learning algorithms generalize by assuming the existence of certain dependencies in the data and generalizing based on those. On the other hand, most statistical techniques (certainly bookkeeping) tend to assume independence in the data (unless specifi-

*This research was supported in part by the National Science Foundation under grant Nos. CDA-8722788, and IRI-9058427 while at the University of California, Berkeley.

cally stated with a distribution or a correlation matrix), and so do not generalize unless instructed to. The successes in both fields make it clear that each approach has its place in modeling data.

- **Sparse Data**: Sparse training data is an unavoidable condition in CPT learning. The size of a CPT is the number of unique parent instantiations (columns) times the number of child values (rows.. see Table 1), and in a practical application can be immense. The situation is aggravated by the fact that the training data will not be evenly distributed throughout a table. What often happens is that a small fraction of the columns in a table will together have a very high probability, and thus take in the bulk of the data. The set of low probability columns rarely see relevant data. The implication is that to "cover" a table with training data, the number of samples already likely to be needed must be multiplied by the inverse of the probability of the lowest probability parent instantiation. Bookkeeping further compounds the problem by requiring a significant amount of data supplied to each column in a CPT in order to produce viable estimates. Machine learning algorithms generalize across the data and often produce excellent results under sparse data conditions.

- **Flexibility**: Each CPT is a different learning problem, with unique characteristics. We can take advantage of the uniqueness by applying the algorithm that best fits the learning task. For tables where a linear relationship is expected between child and parent variables, apply linear regression. When the CPT is moderate sized and well covered by data, apply bookkeeping. When the data is sparse or the relation between child and parents is unknown, apply neural nets or decision trees. This ability to tailor the choice of algorithm to match problem characteristics can make a substantial difference in overall performance.

- **Problem Reduction**: In terms of the complexity of the learning task, the bookkeeping algorithm has a much more difficult job than other approaches. For example, if all variables have 5 values, and node X_i has four parents, then the CPT for X_i has 3125 cells or 625 columns for bookkeeping to learn. A effective neural network applied to the same problem (with a construction similar to what we use in Sections 2.2 and 3) need only learn 76 parameters.

The paper continues in Section 2 with a description of the algorithms that have been implemented and applied to the CPT learning problem. Section 3 explains the experimental methodology, and discusses the results of the implementation including comparisons between bookkeeping and other learning alternatives. Section 4 wraps up with a brief conclusion.

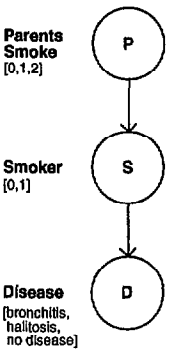

Figure 1: **A Smoker Belief Net**
This is a belief net showing a simplistic relation between smoking, bronchitis and having parents that smoke

Methods Used for Learning CPTs

This section describes the bookkeeping (BOOK), neural network (NN), and combination (COMB) algorithms that we have applied to the CPT learning phase of inducing a belief network from training data. We assume that the structure of the belief network is given. Formal details are kept to a minimum in this section; the interested reader can find in-depth descriptions of how these methods and others can be constructed and applied to CPT learning in (Musick 1994).

Bookkeeping

Bookkeeping is a simple matter of counting the training samples that are relevant to each cell in the CPT. Assuming uninformative priors and sampling without replacement, the application of standard statistical inference methodology (maximum likelyhood estimation) leads to the fact that the counts of relevant samples are actually the parameters that describe a beta distribution for each cell[1] of the CPT. A beta distribution $\beta(a, b)$ is similar in shape to a normal that is a bit skewed, and has a mean of $\frac{a}{a+b}$.

Table 1 contains the CPTs for the belief network in Figure 1, and depicts what happens on a bookkeeping update. The CPTs on the left are the original tables with a uniform prior, the CPTs on the right show what happens after updating for one example of Parents smoke = 0, Smoker = 1 and Disease = bronchitis. Each bullet shows where the sample "hits" in each particular table. When the sample hits a cell in the table, the a parameter is incremented, and the rest of the cells in that column have their b parameter incremented. Consider the $Pr(D/S)$ table, in particular the probability that a patient has bronchitis given that he smokes (this cell was "hit" by the sample). This probability starts at a prior of 1/3, and after the sample increases to 1/2, while the probability of Disease

[1]Or more generally, a Dirichlet for each column of the table.

$Pr(P)$		
P	0	$\beta(1,2)\bullet$
	1	$\beta(1,2)$
	2	$\beta(1,2)$

$Pr(S\|P)$				
		P		
		0	1	2
S	0	$\beta(1,1)$	$\beta(1,1)$	$\beta(1,1)$
	1	$\beta(1,1)\bullet$	$\beta(1,1)$	$\beta(1,1)$

$Pr(D\|S)$			
		S	
		0	1
D	b	$\beta(1,2)$	$\beta(1,2)\bullet$
	h	$\beta(1,2)$	$\beta(1,2)$
	n	$\beta(1,2)$	$\beta(1,2)$

$Pr(P)$		
P	0	$\beta(2,2)\bullet$
	1	$\beta(1,3)$
	2	$\beta(1,3)$

$Pr(S\|P)$				
		P		
		0	1	2
S	0	$\beta(1,2)$	$\beta(1,1)$	$\beta(1,1)$
	1	$\beta(2,1)\bullet$	$\beta(1,1)$	$\beta(1,1)$

$Pr(D\|S)$			
		S	
		0	1
D	b	$\beta(1,2)$	$\beta(2,2)\bullet$
	h	$\beta(1,2)$	$\beta(1,3)$
	n	$\beta(1,2)$	$\beta(1,3)$

Table 1: **Updating the CPTs**
This shows the update process from the original CPTs on the left to the new CPTs on the right, after a sample of P = 0, S = 1, D = b

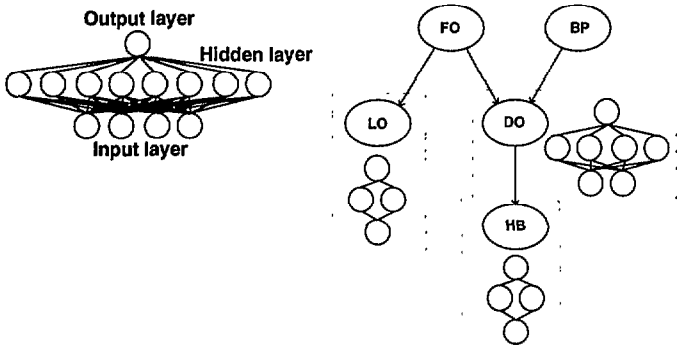

Figure 2: **Applying Neural Nets to Dog Out**
The belief network on the right is for the Dog Out problem, with variables FO = Family Out, BP = dog has Bowel Problems, LO = outside Lights Out, DO = the Dog is Outside, and HB = you can Hear the dog Barking The neural network on the left shows the structure of the networks in the belief network CPTs, and also happens to be the size of a network applied to the same problem The goal is to predict whether or not the dog is outside, based on whether the outside lights are on, if the dog is barking, and so on

= halitosis starts at 1/3 and drops to 1/4. Thus for bookkeeping each complete sample is relevant to an entire column in every CPT. Note also that bookkeeping makes a strong independence assumption; a sample relevant to one column in a CPT is independent of any other column in that CPT.

Neural Networks

The job of the neural network is to produce a density function for each column of the CPT, without any predetermined requirements on the family of functions that could be discovered. Figure 2 is a graphical example of applying neural networks to learn three of the tables in the Dog Out problem (Charniak 1991). Let T_i be the CPT for node X_i, and π_{ij} be a unique parent instantiation for the node. The general process for learning a CPT with NN is for each T_i:

1. Retrieve from the database the instances that are relevant to T_i.
2. Construct a neural network NN_{T_i} and initialize it.
3. Train NN_{T_i} on the samples, where the value of the variable X_i serves as the classification of each datum.
4. Apply input combination π_{i1} to the neural network and read off the distribution for the CPT column $Pr(X_i|\Pi_1)$ from the output units.
5. Normalize the output and write the values into the CPT for the belief network.
6. Repeat steps 4 and 5 for all input combinations π_{i2} through π_{im}.

The network used for the results described below is a 3-layer feed forward network constructed to learn density functions, and incorporates ideas of momentum and adaptive parameters. A network can be built for any combination of input/output variables, including nodes with binary, discrete, continuous, and nominal values. Details of the construction and mapping into CPT learning can be found in (Musick 1994). Note that while bookkeeping provides a distribution for each cell in the CPT, neural networks in general will only provide a point probability.

It should be made clear that this paper is *not* trying to promote this *particular* neural network construction as the best for the problem of CPT learning. In fact, we do not make that claim for any of the algorithms proposed here. Our claim is that effective CPT learning requires the ability to apply a wide range of learning techniques; this construction is used to demonstrate that point.

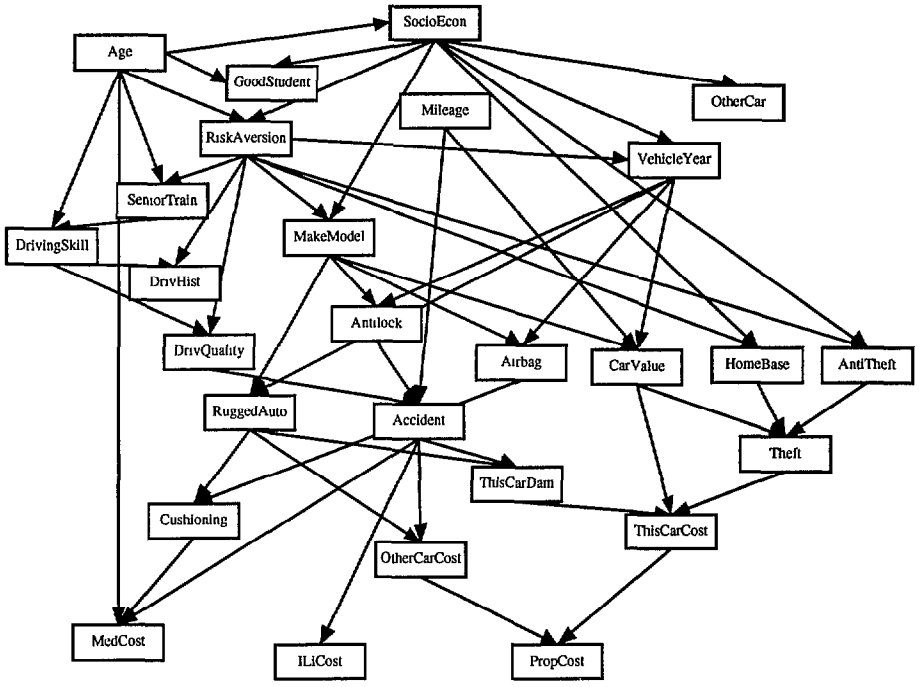

Figure 3: **Car Insurance Database**
This is the model of an insurance database, it includes binary, nominal, discrete and continuous variables.

COMB

COMB is probably the most interesting approach tested. COMB combines both BOOK and NN learning approaches *within one CPT*. The combination is done by choosing to learn data-rich columns with BOOK and the data-poor columns with NN. Specifically, columns with 20 samples or less are learned with NN, columns with more are learned with BOOK. Results are combined on a column by column basis instead of cell by cell since any one sample is relevant to exactly one column. The improvement in the overall model error rate with COMB is impressive.

Results and Comparisons

This section describes the results of an implementation of the above ideas.

Experimental Methodology

The belief network used in this paper is a realistic, moderately sized car insurance network (Russell *et al.* 1995) with 27 binary, discrete, nominal, and continuous nodes, 52 arcs and over 1400 conditional probabilities. The goal is to help predict how much financial risk is incurred from various policy holders given data about age, socio-economic class, and so on. Keep in mind that in high volume industries like insurance or finance, a 1% improvement in risk assessment could be quite valuable.

We need an objective measure of model error rates as a basis from which to compare the different approaches. To enable this, the training data has been *generated* in a very particular way. Figure 3 is taken to be "The" belief network B_S that represents the actual process occurring in the real world. This network is then used to generate a large database (typically on the order of one hundred thousand samples) which is used as the data seen by BOOK, NN, and COMB. The induced belief network B_D has the same structure as B_S, but the CPTs are constructed using a random sample from the large database. The learned belief net B_D is then compared to "The" belief net B_S for model evaluation.

Two methods are used to score the models, the *mean error*, and the *weighted error*. The mean error is the square root of the mean squared error between the predicted probability and the correct probability from all cells in all of the CPTs. The weighted error is the same thing, but with each error weighted by the probability of that related event occurring. Note that one implication of the mean error scoring system is that a data-rich column counts the same as a data-poor column. The mean error metric makes sense in domains where the data-poor columns of the CPT are valuable but scarse data points (for example, drug testing in the medical domain, where the cost of each sample might be a human life). The weighted error metric is more useful when the cost of an error is the same for high and low probability columns.

Finally, all of the results that follow stem from running this process 10 times: generate a random sample of the given size, pass the sample to three separate pro-

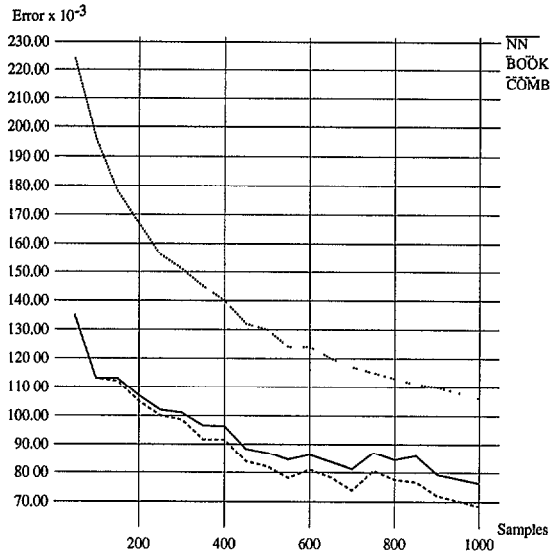

Figure 4: **Car Insurance Database, Mean Error Metric**
The results of applying NN, BOOK and COMB to the car insurance database as the sample size is increased to 1000 samples This graph uses the mean error metric BOOK is the topmost curve, COMB the bottommost

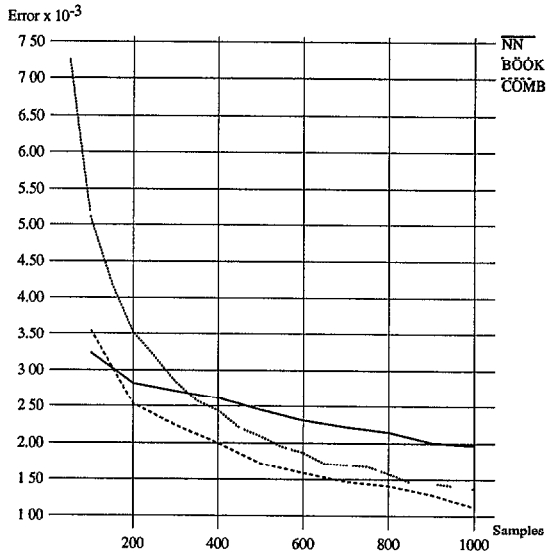

Figure 5: **Car Insurance Database, Weighted Probability Metric**
The results of applying NN, BOOK and COMB to the car insurance database as the sample size is increased to 1000 samples This graph uses the weighted error metric. The BOOK curve starts high and ends in the middle

cesses BOOK, NN and COMB, learn the CPTs, then generate the error measurements.

Experiments

Figures 4 and 5 show the learning performance as the total number of samples is varied from 50 to 1000. The learning curves in Figure 4 were built using the mean error metric, the curves in Figure 5 using the weighted error metric.

The most obvious basic trend is that no matter which error metric is used, for low numbers of samples NN significantly outperforms BOOK. In fact, based on other tests we have run (more than five different belief nets, and different learning algorithms including decision trees) it is fair to say that when the data is sparse relative to the size of the overall learning task, the more traditional machine learning algorithms like neural nets and decision trees significantly outperform BOOK.

The second noticeable aspect of these figures is that there seems to be very different stories being told by the two error metrics. The mean error metric shows NN doubling the performance of BOOK all the way out to 1000 samples and more, while the weighted error metric shows BOOK beginning to outperform NN at about 350 samples. The explanation of this is that the low-probability columns (data-poor columns) in all the CPTs far outnumber the medium and high probability columns. BOOK is expected to perform very well for CPT cells and columns for which there is alot of data, and it is exactly these columns that the weighted error

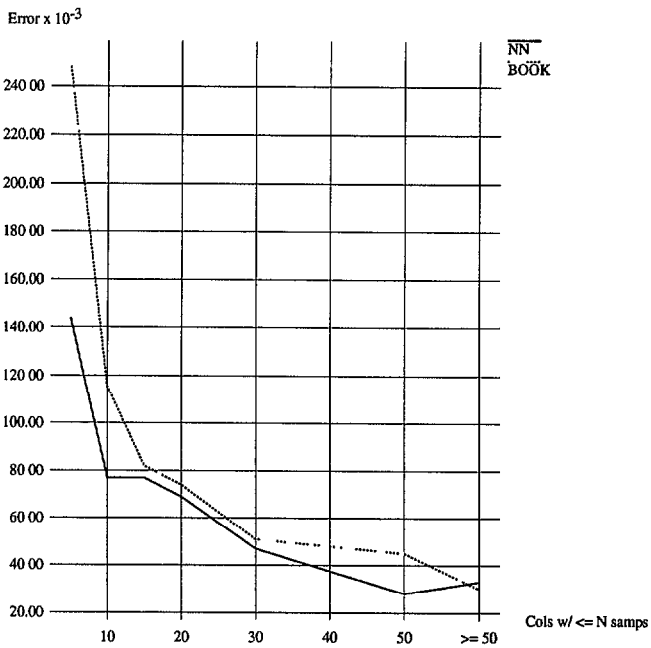

Figure 6: **Error As A Function Of Samples Per Column**
The results of applying NN and BOOK to the car insurance database on 100 samples.

metric is weighting more heavily.

The most interesting feature is that the new algorithm, COMB, consistently outperforms both NN and BOOK. COMB uses NN to learn low-probability columns, and BOOK to learn high probability columns. Even with the weighted error metric on the largest sample (1000 instances), the contribution from NN on the data-poor columns in COMB leads to more than a 10% improvement over the BOOK error rate.

In order to get a better feeling for where and how the improvements are taking place, we clustered columns in the CPTs according to the relative sparseness of the training data. Based on this, Figure 6 should be interpreted as follows: The X axis represents the columns in the CPTs that have seen ≤ 10 samples, 10 to 20 samples, and so on, up to the last point on the axis which represents all of the rest of the columns, all of which are data-rich. The Y axis is the mean error over those clusters. The general expectation is for high error on the data-poor columns to the left, and low error on the data-rich columns to the right. The error metric used for this graph is mean error.

What we see from this graph is that for the CPT columns with relatively little data (less than 50 samples per column), NN outperforms BOOK at all points along the curve.

Conclusion

We tested have several other belief networks and learning algorithms in addition to what is described here, and the results can all be boiled down to the same general conclusions. The nature of the CPT learning problem is that even in the face of massive data sets, data will be sparse for large CPTs. Bookkeeping requires large amounts of data in order to be effective. Machine learning oriented algorithms tend to be more accurate under sparse data conditions. Finally, the ability to freely tailor any learning algorithm to match the unique characteristics of each CPT has the potential to dramatically improve the predictive performance of any belief network model.

References

Charniak, E. 1991. Bayesian networks without tears. *AI Magazine* 12(4):50–63.

D'Ambrosio, B. 1994. Symbolic probabilistic inference in large bn2o networks. In *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*, 128–135.

Ezawa, K., and Norton, S. 1995. Knowledge discovery in telecommunication services data using bayesian networks. In *Proceedings of the First International Conference on Knowledge Discovery and Data Mining*, 100–105.

Musick, R. 1993. Maintaining inference distributions in belief nets. In *Proceedings of the Ninth Conference on Uncertainty in Artificial Intelligence.*

Musick, R. 1994. *Belief Network Induction.* Ph.D. Dissertation, UCB Tech Report CSD-95-863. University of California, Berkeley, Berkeley, CA.

Park, Y.; Han, Y.; and Choi, K. 1995. Automatic thesaurus construction using bayesian networks. In *Proceedings of the Fourth International Conference on Information and Knowledge Management*, 212–217. ACM Press.

Russell, S. J.; Binder, J.; Koller, D.; and Kanazawa, K. 1995. Local learning in probabilistic networks with hidden variables. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence.* Montreal, Canada: Morgan Kaufmann Publishers.

Samuel, A. 1963. Some studies in machine learning using the game of checkers. In Feigenbaum, E. A., and Feldman, J., eds., *Computers and Thought.* New York: McGraw-Hill.