Using The Central Limit Theorem for Belief Network Learning

Ian Davidson[1], Minoo Aminian[1]

[1]Computer Science Dept, SUNY Albany
Albany, NY, USA, 12222.
davidson@cs.albany.edu

Abstract. Learning the parameters (conditional and marginal probabilities) from a data set is a common method of building a belief network. Consider the situation where we have many complete (no missing values), same-sized data sets randomly selected from the population. For each data set we learn the network parameters using only that data set. In such a situation there will be no uncertainty (posterior distribution) over the parameters in each graph, however, how will the parameters learnt differ from data set to data set? In this paper we show the parameter estimates *across* the data sets converge to a Gaussian distribution with a mean equal to the population (true) parameters. This result is obtained by a straight-forward application of the central limit theorem to belief networks. We empirically verify the central tendency of the learnt parameters and show that the parameters' variance can be accurately estimated by Efron's bootstrap sampling approach. Learning multiple networks from bootstrap samples allows the calculation of each parameter's expected value (as per standard belief networks) and also its second moment, the variance. Having the expectation and variance of each parameter has many potential applications. In this paper we discuss initial attempts to explore their use to obtain confidence intervals over belief network inferences and the use of Chebychev's bound to determine if it is worth gathering more data for learning.

Introduction

Bayesian belief networks are powerful inference tools that have found many practical applications [1]. A belief network is a model of a real world situation with each node ($N_1 \dots N_m$) in the network/graph (G) representing an event. An edge/connection ($e_1 \dots e_l$) between nodes is a directional link showing that an event influences or causes the value of another. As each node value can be modeled as a random variable ($X_1 \dots X_m$), the graph is effectively a compact representation of the joint probability distribution over all combinations of node values. Figure 1 shows a belief network modeling the situation that relates smoking, cancer and other relevant information. Building belief networks is a challenging task that can be accomplished by knowledge engineering or automatically producing the model from data, a process commonly known as learning [2].

Learning graphical models from a training data set is a popular approach where domain expertise is lacking. The training data contains the occurrence (or otherwise) of the events for many situations with each situation termed a training example, instance or observation. For example, the parameters for the Cancer network shown in Figure 1 could be learnt from many patient records that contained the occurrence or not of the events. The four commonly occurring situations of learning belief networks [3] are different combinations of a) learning with complete (or incomplete) data and b) learning with known (or unknown) graph structure. Learning with complete data indicates that the training data contains no missing values, while incomplete data indicates for some training examples, pieces of information of the world situation were unknown. With known graph structure the learning process will attempt to estimate the parameters of the random variables while with unknown structure the graph structure is also learnt. There exists a variety of learning algorithms for each of these situations [1][3]. In this paper we shall focus on the situation where the data is complete and graph structure is specified. In this simplest situation there is no posterior distribution over the model space as there is no uncertainty due to latent variables or missing values. This allows us to focus on the uncertainty due to variations in the data. In future work we hope to generalize our findings for other learning situations. Without loss of generality and for clarity we shall focus on Boolean networks.

Learning the parameters of the graph from the training data D of s records involves counting the occurrences of each node value (for parentless nodes) across all records to calculate the marginal probabilities or combination of values (for nodes with parents) and normalizing to derive the conditional probabilities. Now consider having many randomly drawn data sets ($D_1 \dots D_t$) all of size s from which we learn model $\theta_1 \dots \theta_t$. Each model places a probability distribution over the event space, which is all combinations of node values. How will these parameters vary from model to model? We shall answer this question by considering a probability distribution over the **parameters** for each node. We shall call these random variables $\{\boldsymbol{Q_1 \dots Q_m}\}$. Note that $\boldsymbol{Q_i}$ may be a block of random variables as the number of parameters per node can vary depending on its number of parents. In a Boolean network for a parentless node, $\boldsymbol{Q_i}$ is a single random variable whose value represents node parameter, P(X_i = *TRUE*). For nodes that have parents in a Boolean network, $\boldsymbol{Q_i}$ consists of $2^{(\#Parents)}$ random variables representing the parameters P(X_i = *TRUE* | $Condition_1$) … P(X_i = *TRUE* | $Condition_{2^\wedge(\#Parents)}$) with the different conditions being the various combinations of TRUE and FALSE values of the parents.

What can we say of these continuous random variables that represent the parameter estimates learnt from different random samples? In this paper we show that these continuous random variables form a Gaussian distribution that center on the population values (the parameters that would be learnt from the entire population of data) and whose variance is a function of the sample size. We show that $\boldsymbol{Q_i}$ ~ Gaussian(p_i^*, $p_i^*(1\text{-}\ p_i^*)$ /(ks)) where k is a constant, p_i^* is the relevant parameters of the generating mechanism that produced the data and s is the sample size. The standard deviation of this random variable approaches zero and the sample mean converges asymptotically to the population value as the value of s increases. We can now make two estimates for each parameter in the network: its expected value and its variance.

In most learning situations the actual probability distribution over the learnt parameters due to uncertainty/variability in the **observed data set** is unknown. Sometimes, generalities are known such as that decision trees are unstable learners or that iterative algorithms that minimize the distortion (vector quantization error) are sensitive to initial parameters. In this paper we make use of the probability distribution over the learnt parameters for two applications. These should be treated as initial attempts to make use of the mean and variance of each parameter and future work will refine these applications. Firstly, we use the variance to determine if gathering more data will necessarily change the mean value of the parameters by at least some user specified value. This has applications when collecting data is time consuming or expensive. We then illustrate how to create confidence intervals associated with each inference made from the network.

We begin this paper by describing the learning of belief networks, readers familiar with this topic may skip this section. The applicability of the central limit theorem to belief network learning with complete data is described next. We then empirically verify that the central tendency of the distribution of the learnt parameters by sampling the data to learn **without** replacement from a large collection of data (generated by a Gibbs sampler). Having such a large pool of data is an unusual luxury and we next illustrate approximating the variance associated with the learnt parameters using Efron’s bootstrap sampling (sampling with replacement) [5]. We next discuss how to use the estimated variance for our two proposed applications. Finally we conclude our work and describe future directions.

Learning Belief Networks

The general situation of learning a belief network can be considered as consisting of estimating all the parameters (marginal and conditional probabilities), $\boldsymbol{\theta}$, for the nodes and discovering the structure of the directed acyclic graph, G. Let $\theta_i = \{\boldsymbol{q_{i1}} \dots \boldsymbol{q_{im}}\}$ be the parameters of the m node graph learnt from data set i. Note that the corresponding upper-case notation indicates the random variable for these parameters. As the network is Boolean, we only need the probability of the node being one value given its conditions (if any). In later sections we shall refer to graph and network together as the model of the data, $H=\{\theta, G\}$.

This paper focuses on learning belief networks in the presence of complete data and a given graph structure. From a data set $D_i = \{d_{i1}, \dots , d_{is}\}$, $d_{il} = \{a_{i11} \dots a_{ilm}\}$ of s records, we find either the marginal or conditional probabilities for each node in the network after applying a Laplace correction. The term a_{ilj} refers to the j^{th} node’s value of the l^{th} instance within the i^{th} data set. For complete data, known graph structure and Boolean network the calculation for node i is simply:

If N_j has no parents, $q_{ij} = (1 + \sum_l a_{ilj} = T)/(s+2)$ (1)

If N_i has parents $Parents(N_i) = N_k$ (2)

$$q_{ijT} = \left(1 + \sum_l a_{ilj} = T \text{ and } a_{ilk} = T\right) / \left(2 + \sum_l a_{ilk} = T\right)$$

$$q_{ijF} = \left(1 + \sum_l a_{ilj} = T \text{ and } a_{ilk} = F\right) / \left(2 + \sum_l a_{ilk} = F\right)$$

Note the **subscript** T or F refers to the node’s parent’s value, the parameter q always refers to probability of the node taking on the value T. For example q_{ijF} refers to the probability learnt from the i^{th} data set that the j^{th}

node will be *T* given its parent is *F*. If a node has more than one parent then the expression for the parameter calculations involves similar calculations over all combinations of the parent node values.

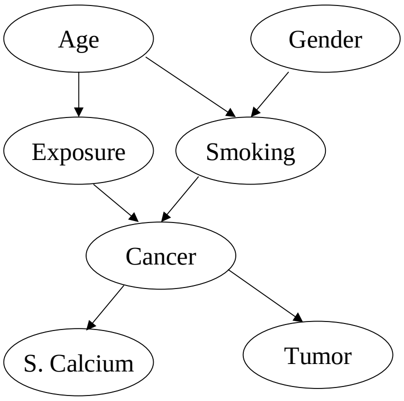

Figure 1. The Cancer Belief Network.

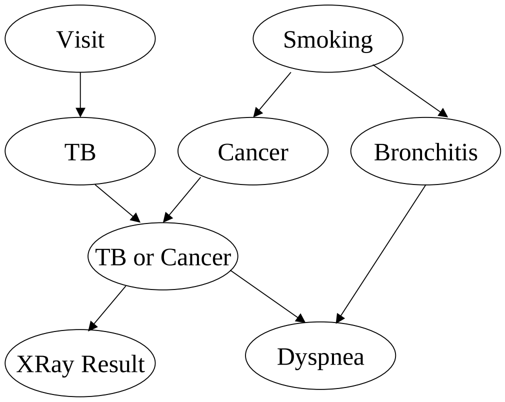

Figure 2. The Asia Belief Network

The Asymptotic Distribution of Learnt Parameters for Belief Networks

Consider as before a number of random samples of training data (D_1 … D_t) each of size *s*. We divide our graph of nodes into two subsets, those with no ancestors (*Parentless*) and those that have ancestors and are hence children (*Children*). We initially consider binary belief networks, whose nodes can only take on two values T (*TRUE*) and F (*FALSE*) and children that have only one parent, but our results generalize to multi-parent nodes. For each training data (D_k) set we learn a model θ_k as described earlier.

Under quite general condition (finite variance) the parameters of the nodes that belong to *Parentless* will be Gaussian distributed due to the central limit theorem [4]. We can formally show this by making use of a common result from the central limit theorem, the Gaussian approximation to the binomial distribution [4].

A similar line of argument that follows could have been established to show the central tendency around each and every joint probability value (P(N_1… N_m)) but since the belief network is a compact representation of the joint distribution, we show the central tendency around the parameters of the network.

Situation #1: ***$N_j \in$ Parentless***

Let p_j be the proportion of times in the entire population (all possible data) that the *j*th node (a parentless node) is TRUE. If our sampling process (creating the training set) is random then when we add/sample a record, the chance the *j*th node value is TRUE can be modeled as the result of a Bernouli trial (a Bernouli trial can only have one of two outcomes). As the random sampling process generates independent and identically data (IID) then the probability that of the *s* records in the sample that *n* records will have TRUE for the *j*th node is given by a Binomial distribution:

$$P(s,n,p_j)=\frac{s!}{n!(s-n)!}\left(p_j\right)^n\left(1-p_j\right)^{s-n} \qquad (3)$$

Applying the Taylor series expansion to equation (3) with the logarithm function for its dampening (smoothness) properties and an expansion point $p_j.s$ yields the famous Gaussian distribution expression:

$$P(s,n,p_j)=\frac{1}{\sqrt{2\pi\sigma}}e^{-(n-s.p_j)^2/(2\sigma^2)},\sigma^2=sp_j(1-p_j) \qquad (4)$$

If the sample size is *s* then the number of occurrences of X_j=T in any randomly selected data set will be given by a Gaussian whose mean is $s.p_j$ and variance $s.p_j(1- p_j)$ as shown in equation (4).

Therefore, Q_j ~ *Gaussian*(p_j, p_j(1- p_j)/*s*)) in the limit when $t \rightarrow \infty$ and $p_j > \varepsilon$, where p_j is the proportion of times X_j=T in the population.

Situation #2. $N_i \in$ Children and Parent(N_i) = N_j … N_k

A similar argument as above holds except that there are now as many Bernouli experiments as there are combinations of parent values. In total there will be 2^k Bernouli experiments. Therefore, Q_i ~ Gaussian($p_{i,j…k}$, ($p_{i,j…k}$)(1-$p_{i,j…k}$)) / (sc_k)). Note that sc_k is just the proportion of times that the condition of interest (the parent value combination) occurred.

A valid question to ask is how big should p_j (or $p_{i,j...k}$) and s (or sc_k) be to obtain reliable estimates. The general consensus in empirical statistics is that if $t >$ 100, s > 20 and p_j (or $p_{i,j...k}$) > 0.05 then estimates made from the central limit theorem are reliable [4]. This is equivalent to the situation of learning from one hundred samples, where each parental condition occurs at least twenty times and no marginal or conditional probability estimate less than 0.05. While these conditions may seem prohibitive, we found that in practice for standard belief networks tried (Asia, Cancer and Alarm) that the distribution of parameters was typically Gaussian. Table 1 shows for the Cancer data set that from 249 training data sets each of size 1000 training instances that the average of the learnt parameters passed a hypothesis test at the 95% confidence level when compared to the population (true) parameters. Similar results were obtained for the Cancer and Alarm data sets.

	n	Mean
Gender	249	0.4874
Hypothesised		0.5130

Difference between means	-0.0256
95% CI	-0.0313 to -0.0199

	n	Mean
Age	249	0.3022
Hypothesised		0.3030

Difference between means	-0.0008
95% CI	-0.0070 to 0.0055

	n	Mean
Ex.to.Tox.	249	0.4639
Hypothesised		0.4190

Difference between means	0.0449
95% CI	0.0342 to 0.0556

	n	Mean
Smoking	249	0.4670
Hypothesised		0.4850

Difference between means	-0.0180
95% CI	-0.0322 to -0.0039

	n	Mean
Cancer	249	0.2977
Hypothesised		0.3090

Difference between means	-0.0113
95% CI	-0.0227 to 0.0000

	n	Mean
S.C.	249	0.3210
Hypothesised		0.3170

Difference between means	0.0040
95% CI	-0.0025 to 0.0106

	n	Mean
Lung Tmr	249	0.2454
Hypothesised		0.2390

Difference between means	0.0064
95% CI	-0.0023 to 0.0151

Table 1. For the Cancer data set, hypothesis testing of the mean of the learnt parameter and the population (true value) of the parameter. In all cases the difference between the true value and the mean of the learnt parameters lied in the 95% confidence interval. Only a subset of the hypothesis tests are presented for space reasons.

Using Bootstrapping to Estimate the Variance Associated With Parameters

Having the ability to generate random independent samples from the population is a luxury not always available. What is typically available is a single set of data *D*. The bootstrap sampling approach [5] can be used to sample from *D* to approximate the situation of sampling from the population. Bootstrap sampling involves drawing *r* samples ($B_1 \dots B_r$) of size equal to the original data set by sampling **with replacement** from *D*. Efron showed that in the limit the variance amongst the bootstrap sample means will approach the variance amongst the independently drawn sample means. We use a visual representation of the difference between the probability distribution over the learnt parameter values (249 bootstrap samples each of size 1000 instances) and a Gaussian distribution with a mean and standard deviation calculated from the learnt parameter values as shown in Figure 3. As expected the most complicated node (smoking) had the greatest deviation from the normal distribution.

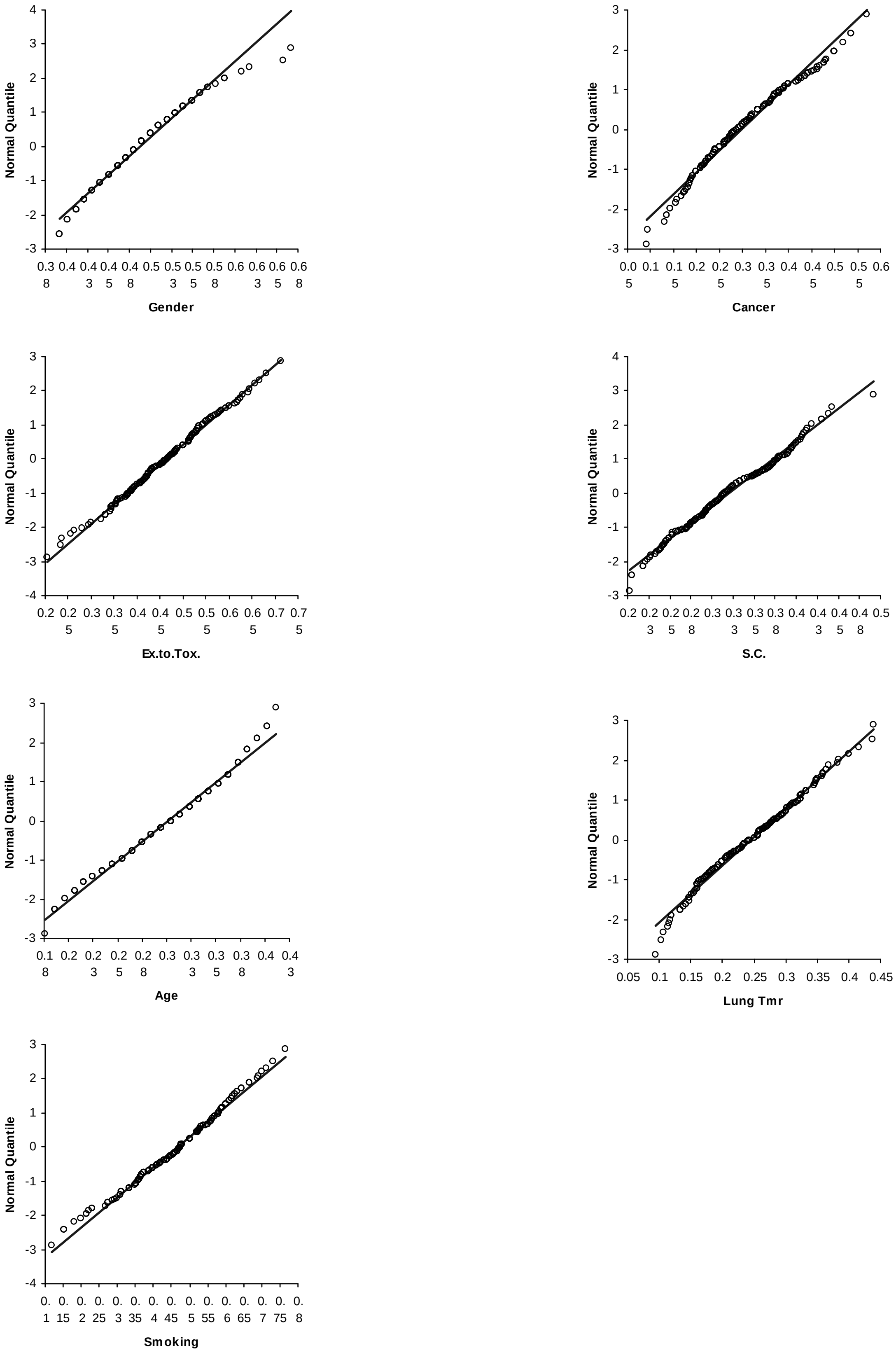

Figure 3. For the Cancer data set the difference between the distribution over the learnt parameter values and a Gaussian distribution with the mean and the standard deviation of the learnt parameter values. The straight line indicates the Gaussian distribution. The x-axis is the parameter value while the distance between the lines is the Kullback-Leibler distance between the distributions at that point.

Learning and Using the Variance of Learnt Parameters

We now describe an approach to learn the belief network parameters' mean and variance. Firstly, the parameters of a belief network are learnt from the available data using the traditional learning approach determined earlier. Secondly, bootstrap sampling is performed to estimate the variance associated with each learnt parameter. Estimating the variance involves taking a number of bootstrap samples of the available data, learning the parameters of the belief network from each and measuring the variance across these learnt parameters. Therefore, for each parameter in the belief network, we now have its expected value and variance. Estimates of the variance of the parameters could have been obtained from the expressions reported in situation#1 and situation#2 earlier. However, in practice we found that these were not as accurate as calculating variances from a large number (more than 100) bootstrap samples. This was because from our limited training data the learnt parameter was often different from the true value.

In this paper we propose two uses of these additional parameters which we describe next: determining if more data is needed for learning and coming up with confidence intervals over inferences.

Determining the Need for More Data

A central problem with learning is determining how much data is need (how big *s* should be). The computational learning theory literature [6] addresses this problem, unfortunately predominantly for learning situations where no noise exists, that is the Bayes error of the best classifier would be 0. Though there has been much interesting work studying the behavior of iterative algorithms [7][8] this work is typically for a fixed data set. The field of active learning attempts to determine what type of data to add to the initial training data set [9] (the third approach in the paper) [10] and [11]. However, to our knowledge this work does not address the question of when to **stop** adding data.

We now discuss the use of Chebychev's bound to determine if more data is needed for learning. Others [12] have used similar bounds (Chernoff's bound) to produce lower bound results for other pattern recognition problems. Since both the expectation and variance of the parameters of interest are known, we can make use of the Chebychev inequality.

The Chebychev inequality allows the definition of the sample size, *s*, required to obtain a parameter estimate ($\hat{p}$) within an error (ε) of the true value from a distribution with a mean of *p* and standard deviation of σ as shown below.

$$P\left[\left|\hat{p}-p\right|>\varepsilon\right]<\frac{\sigma^2}{s\varepsilon^2} \qquad (5)$$

This expression can be interpreted as an upper bound on the chance that the error is larger than ε. In-turn we can upper bound the right-hand side of this expression by δ which can be considered the maximum chance/risk we are willing to take that our estimate and true value differ by more than ε. Then rearranging these terms to solve for the sample size yields:

$$s \geq \frac{\sigma^2}{\varepsilon^2\delta} \qquad (6)$$

Typical applications of the Chebychev inequality use $\sigma^2=p(1-p)$ and produce a bound that requires knowing the parameter that is trying to be estimated! [13]. However, we can use our empirically (through bootstrapping) obtained variance and circumvent this problem.

The question of how much data is needed, is now answered with respect to how much error (ε) in our estimates we are willing to tolerate and the chance (δ) that this error will be exceeded. We can use equation (6) to determine if more data is needed if we are to tolerate learning parameters that with a chance δ differ from the true parameters by more than ε. The constants δ and ε are set by the user and effectively define the users definition of how "close" is close enough. For each node in the belief network we compute the number of instances of each "type" required. We then compare this against the number of actual instances in the data set of this type. *If for any node the required number is less than the actual number then more data needs to be added.* By instance "type", we mean instances with a specific combination of parent node values. There as many instance "types" as there are **needed** marginal or conditional probability table entries. For example, for the Cancer network (Figure 1) the Exposure node generates two types, one each for when its parent's (Age) value is T or F, the value of the remaining nodes are not specified.

To test this stopping criterion we need to empirically determine when adding new instances to the training set adds no information. This involves measuring the expectation (over the joint distribution) of the code word lengths for each combination of node values. We wish to only add instances to the training set if the instances reduce the mean amount of information needed to specify a combination of node values. As the training data size increases the parameters learnt stabilize and better approximate the chance occurrence in the population. Formally, the expected information is:

$$Average\#bits = -\sum_{i=1}^{2^m} H^*(E_i).\log(\theta(E_i)) \qquad (7)$$

where E_i is a particular combination of node values of which there are 2^m for a m binary node network. $H^*(.)$ is the probability distribution of the generating mechanism and $\theta(.)$ is the learnt distribution.

Figure 4 shows the expected information for different sized training sets for the Asia and Cancer networks averaged over 100 experiments. In each of these experiments the training data is incrementally added by sampling without replacement from the population (50,000 instances generated from a Gibbs sampler). Calculating this expected information requires a significant amount of time and cannot be used for determining when to stop adding instances to the training set.

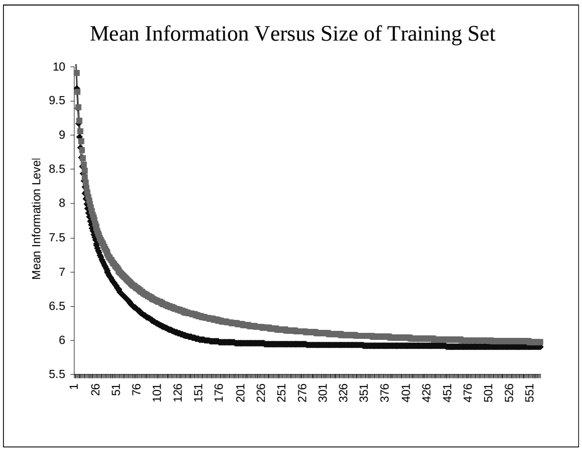

Figure 4. The mean information for the joint distribution for the Asia (top curve) and Cancer networks against training set size (x10). At approximately 5510 instances and 1510 instances the decrease in information due to adding instances are negligible for the Asia and Cancer data sets respectively.

For one hundred experiments we used the test described earlier in italics with $\varepsilon = 0.05$ and δ=0.025. Whether these results generalize to more complicated networks remains as future work.

	Stopping Point (From Figure 4)	**Mean Over All Trials**	**Standard Deviation Over All Trials**
Asia	5510	5203	129
Cancer	1510	1567	98

Table 2. For 100 trials, the mean number of instances in the training set before the test based on equation (6) specifies adding no more data points. The correct number of instances is visually determined from Figure 4.

Confidence Intervals Over Inferences

Typically with belief networks an inference is of the form that involves calculating a point estimate of $P(N_i=T \mid \{E\})$ where E is some set of node values that can be considered as evidence. For small networks exact inference is possible but since exact inference is NP-Hard for larger networks approximate inference techniques such as Gibbs sampling is used [2]. However, if the variance of each parameter is known we can now create a confidence interval associated with $P(N_i=T \mid \{E\})$. We now describe a simple approach to achieve this.

For a belief network parameter of value p and standard deviation σ we know that with 95% confidence that its true value is in the range $p \pm 1.96\sigma$. We can now create **three** versions of the belief network, one where all network parameters are set to p (the expected case) another where the network parameters are set to p-1.96σ

(lower bound) and another where they are set to $p+1.96\sigma$ (upper bound). Point estimation is performed in each of the three networks using whatever inference technique is applicable and the three values are then bought together to return an expected, lower and upper bound on the inference to provide a 95% confidence interval.

Though it may be thought that the changes amongst these networks will be trivial and not return greatly varying inferences, recent work [14] has shown that even small changes in network parameters can result in large changes in the inference/query probabilities.

Future Work and Conclusion

Our work posed the question: Given many independently drawn random samples from the population, if we learn the belief network parameter estimates from each, how will the learnt parameters differ from sample to sample. We have shown that for complete data and known graph structure that the learnt parameters will asymptotically distributed around the generating mechanism (true) parameters. This result follows from the central limit theorem and has many potential applications in belief network learning. This means each node in the belief network is described by the parameter's mean and variance. The variance parameter measures the uncertainty in the parameter estimates due to variability in the data.

However, having many random samples from which to estimate the variance is a great luxury. By using bootstrap sampling we can create an estimate of the variance by sampling from a single data set with replacement. Having a probability distribution over the learnt parameters due to **observed** data uncertainty has many potential applications. We describe two. We show how to create upper and lower bounds on the probabilities of inferences by creating three networks that represent the expected, lower bound and upper bound. Using the mean and variance associated with each parameter allows determining if more data is required so as to guarantee with a certain probability that the difference between the expected and actual parameter means are within a predetermine error.

We have limited our results in this paper to the simplest belief network learning situation to ensure the uncertainty in the parameters is only due to data variability. We intend to next extend our results to the situation where EM and SEM is used to learn from incomplete data and unknown graph structure.

References

[1] F. Jensen, An Introduction to Bayesian Networks, Springer Verlag, 1996

[2] J. Pearl, *Probabilistic Reasoning in Intelligent Systems*, Morgan Kaufmann

[3] Friedman, N. (1998), The Bayesian structural EM algorithm, in G. F. Cooper & S. Moral, eds., Proc. Fourteenth Conference on Uncertainty in Artificial Intelligence (UAI '98)',

[4] Devore, Peck,., Statistics: The Exploration and Analysis of Data, 3rd Ed, Wadsworth Publishing. 1997.

[5] Efron, B. 1979. Bootstrap methods: another look at the jackknife. *Annals of Statistics* 7: 1-26.

[6] Mitchell, T., *Machine Learning*, McGraw Hill, 1997.

[7] Bottou, L., and Bengio, Y., Convergence properties of the k-means algorithm. In G. Tesauro and D. Touretzky, editors, Adv. in Neural Info. Proc. Systems, volume 7, pages 585--592. MIT Press, Cambridge MA, 1995.

[8] Ruslan Salakhutdinov & Sam T. Roweis & Zoubin Ghahramani, Optimization with EM and Expectation-Conjugate-Gradient, International Conference on Machine Learning 20 (ICML03), pp.672--679

[9] D. MacKay. Information-based objective functions for active data selection. Neural Computation, 4(4):590--604,

[10] Simon Tong, Daphne Koller, Active Learning for Parameter Estimation in Bayesian Networks. NIPS 2000:

[11] Cohn, D.; Atlas, L.; and Ladner, R. 1994, Improving Generalization with Active Learning. Machine Learning 15(2):201-221.

[12] H. Mannila, H. Toivonen, and I. Verkamo. Efficient algorithms for discovering association rules. In AAAI Wkshp. Knowledge Discovery in Databases, July 1994

[13] M. Pradhan and P. Dagum. Optimal monte carlo estimation of belief network inference. In Proceedings of the 12th Conference on Uncertainty in Artificial Intelligence, pages 446-- 453, Portland, Oregon, 1996.

[14] Hei Chan and Adnan Darwiche. When do Numbers Really Matter? In JAIR 17 (2002) 265-287.