Applied Mathematics & Information Sciences
An International Journal

Study of Bayesian Network Structure Learning

Wei Xiong[1], *Yonghui Cao*[1,2] *and Hui Liu*[3]

[1]School of Management, Zhejiang University,Hangzhou 310058 P. R. China
[2]School of Economics and Management, Henan Institute of Science and Technology,Xinxiang 453003 P. R. China
[3]School of Computer and Information Technology, Henan Normal University, Xinxiang 453007 P. R. China

Received: 25 Jul. 2012, Revised: 18 Sep. 2012, Accepted: 1 Oct. 2012
Published online: 1 Feb. 2013

Abstract: There are different structure of the network and the variables, and the process of learning Bayesian networks has a lot of different forms. The structure of the network can be unknown or known, and the variables can be observable or hidden in some or all of the data points. Consequently, there are four cases of learning Bayesian networks from data: known structure and observable variables, unknown structure and observable variables, known structure and unobservable variables and unknown structure and unobservable variables. In this paper, we focus on known structure and observable variables, unknown structure and observable variables.

Keywords: Bayesian networks, Network Structure, Observable Variables

1 Introduction

A Bayesian network, Bayes network, belief network or directed acyclic graphical model is a probabilistic graphical model that represents a set of random variables and their conditional dependencies via a directed acyclic graph (DAG). [1,2] For example, a Bayesian network could represent the probabilistic relationships between diseases and symptoms. Given symptoms, the network can be used to compute the probabilities of the presence of various diseases.

Formally, Bayesian networks are directed acyclic graphs whose nodes represent random variables in the Bayesian sense: they may be observable quantities, latent variables, unknown parameters or hypotheses. Edges represent conditional dependencies; nodes which are not connected represent variables which are conditionally independent of each other. Each node is associated with a probability function that takes as input a particular set of values for the node's parent variables and gives the probability of the variable represented by the node. For example, if the parents are m Boolean variables then the probability function could be represented by a table of 2m entries, one entry for each of the 2m possible combinations of its parents being true or false.

Efficient algorithms exist that perform inference and learning in Bayesian networks. Bayesian networks that model sequences of variables (e.g. speech signals or protein sequences) are called dynamic Bayesian networks. Generalizations of Bayesian networks that can represent and solve decision problems under uncertainty are called influence diagrams.

Because a Bayesian network is a complete model for the variables and their relationships, it can be used to answer probabilistic queries about them. For example, the network can be used to find out updated knowledge of the state of a subset of variables when other variables (the evidence variables) are observed. This process of computing the posterior distribution of variables given evidence is called probabilistic inference. The posterior gives a universal sufficient statistic for detection applications, when one wants to choose values for the variable subset which minimize some expected loss function, for instance the probability of decision error. A Bayesian network can thus be considered a mechanism for automatically applying Bayes' theorem to complex problems.[3]

How can Bayesian networks be learned from data? The process of learning Bayesian networks takes different forms in terms of whether the structure of the network is known and whether the variables are all observable. The structure of the network can be known or unknown, and the variables can be observable or hidden in all or some of the data points. The latter distinction can also be expressed as complete and incomplete data. Consequently, there are four cases of learning Bayesian

* Corresponding author e-mail: wxiong@zju.edu.cn,caoyonghui2000@126.com

networks from complete data: known structure and observable variables, unknown structure and observable variables, known structure and unobservable variables, and unknown structure and unobservable variables. In this paper we discussed two of them.

2 Known Network Structure and Observable Variables

This is the easiest and the most studied case of learning Bayesian networks in the literature [1, 2]. The network structure is specified, and the inducer only needs to estimate the parameters. The problem is well understood and the algorithms are computationally efficient. Despite its simplicity, this problem is still extremely useful, because numbers are very hard to elicit from people. Additionally, it forms the basis for everything else in Bayesian learning.

Because every variable is observable, each data case can be pigeonholed into the CPT entries corresponding to the values of the parent variables at each node. The pigeonhole principle essentially states that if a set consisting of more than $k \cdot n$ objects is partitioned into n classes, then some classes receive more than k objects. [4,5]Therefore, estimations will be highly accurate since every variable is observable.

Learning is achieved simply by calculating conditional probability table (CPT) entries using estimation techniques such as Maximum Likelihood Estimation (MLE) and Bayesian Estimation. For simplicity, MLE and Bayesian estimators will be explained by employing parameter learning for a single parameter.

Assume that an experiment was conducted by flipping a thumbtack in the air. The thumbtack comes to land as either heads or tails. As usual, the different tosses are assumed to be independent, and the probability of the thumbtack landing heads is some real number θ.Therefore, the goal is to estimate θ. Assume that we have a set of instances $d[1],\cdots,d[2]$ such that each instance is sampled from the same distribution and independently from the rest. The goal is to find a good value for the parameter θ. A parameter is good if it predicts the data well. In other words, if data are very likely given the parameter, the parameter is a good predictor. The likelihood function is defined as

$$L(D|\theta) = P(D|\theta) = \prod_{m=1}^{M} P(d[m]\theta) \tag{1}$$

Thus, the likelihood for a sequence H,T,T,H,H is

$$L(D|\theta) = \theta(1-\theta)(1-\theta)\theta\theta \tag{2}$$

To calculate the likelihood we need to know number of heads N_h and the number of tails N_t.These are the sufficient statistics for this learning problem. A sufficient statistic is a function of the data that summarize the relevant information for computing the likelihood.

The Maximum Likelihood Estimation (MLE) principle tells us to choose θ that maximizes the likelihood function. The MLE is one of the most commonly used estimators in statistics. For the above problem, the estimation of the parameter is as expected.

$$\hat{\theta} = \frac{N_h}{N_h + N_t} \tag{3}$$

The MLE estimate seems plausible, but is overly simplistic in many cases. Assume that the experiment with the thumbtack is done and 3 heads out of 10 are recorded. It may be quite reasonable to conclude that the parameter θ is 0.3. On the other hand, what if the same experiment is done with a dime and also 3 heads are recorded. We would be much less likely to jump the conclusion that the parameter of the dime is 0.3 because we have a lot more experience with tossing dimes. Thus, we have a lot more prior knowledge about their behavior.

Using MLE, we cannot make the following distinctions: between a thumbtack and a dime, and between 10 tosses and 1,000,000 tosses of a dime. On the other hand, there is another method recommended by Bayesian statistics. The MLE is a frequentist approach since it relies on the frequency in the data. Another approach is the Bayesian approach that assumes that there is unknown but fixed parameter θ.It estimates the parameter with some confidence, i.e., it calculates a range such that, if the parameter is out of this range, the probability of the data is very low.

The Bayesian approach deals with uncertainty over anything that is unknown by putting a distribution over it. In other words, the parameter θ is treated as a random variable and a distribution $P(\theta)$ is defined over it. Therefore, we can tell how likely the parameter is to take on one value versus another. In other words, we now have a joint probability space that contains both the tosses and the parameter. This joint probability is easy to find given our prior distribution over θ. Let $X[1],\cdots,X[M]$ be our coin tosses. The conditional probabilities $P(X[M]\theta)$are according to θ, i.e., $p(X[M] = H|\theta) = \theta$. Now, the value of the next toss $X[M+1]$can be predicted by

$$P(X[M+1]|X[1],\cdots,X[M]) = \int P(X[M+1]|\theta)P(\theta|D)d\theta \tag{4}$$

where

$$P(\theta|D) = \frac{P(D|\theta)p(\theta)}{P(D)} \tag{5}$$

The first term in the numerator is the likelihood, the second is the prior over parameters, and the third is a normalizing factor, which is the marginal probability of the data. If we reconsider the thumbtack problem again with a uniform prior over θ in the interval $[0,1]$then $P(D|\theta) = \theta^{N_k}(1-\theta)^{N_t}$ is proportional to the likelihood . After plugging this into the integral and doing all the

math and normalizing, it can be shown that the following equation holds.

$$P(X[M+1]|D) = \frac{N_h + 1}{N_h + N_t + 2} \tag{6}$$

Clearly, as the number of samples grows, the Bayesian estimator and the MLE estimator converge to each other. This result depends on the use of uniform prior. In the Bayesian networks literature, the most commonly used class of priors are the Dirichlet priors because it turns out that most of the interesting calculations can be done in closed form. The conjugacy of the Dirichlet priors allows us to have the posterior probabilities in the same form as prior probabilities. Therefore, we can do sequential updating within the same representations and the closed form solution can found both for the update and the prediction problem in many cases.

Recall that a multinomial is parameterized via a set of parameters $\theta_1 \cdots, \theta_k$ such that $\sum_i \theta_i = 1$; θ_i corresponds to the probability of ith outcome. A Dirichlet distribution over this set of parameters $\alpha_1, \cdots, \alpha_k$ is defined via a set of hyper parameters .Then, the generalization can be written as

$$Dir(\theta|\alpha_1, \cdots, \alpha_k) = \frac{\Gamma(\alpha)}{\prod_i \Gamma(\alpha_i)} \prod_i \theta_i^{\alpha_i - 1} \tag{7}$$

All of the results regarding prediction and computing the posterior extend in the obvious way. That is, if θ is distributed as in (7), then

$$P(x_i) = \frac{\alpha_i}{\sum_j \alpha_j} \tag{8}$$

To generalize these results for a Bayesian network, we need to define the sufficient statistic as $N(x,u)$ for the event $X = x$ and the parents $U = u$. In the MLE case, the estimation of the parameters can be calculated as

$$\hat{\theta}_{x|u} = \frac{N(x,u)}{N(u)} \tag{9}$$

Similarly, in the Bayesian case, the parameter estimation is calculated as

$$\hat{\theta}_{x|u} = Dir(\alpha_1 + N(x_1,u), \cdots, \alpha_k + N(x_k,u)) \tag{10}$$

If the data were actually generated from the given network structure, then both methods converge asymptotically to the correct parameter setting. If not, then they converge to the distribution with the given structure that is closest to the distribution from which the data were generated. Both estimations can be implemented online by accumulating sufficient statistics.

The process above is the method by which Bayesian network parameters are learned when the network topology is known and all variables are fully observable. The next section provides an overview of some proposed methods in the literature if the structure of the network is not known in advance.

3 Unknown Network Structure and Observable Variables

In this case, the inducer is given the set of variables in the model, and needs to select the arcs between them and estimate the parameters. This problem is very useful for a variety of applications; in general, when we are given a new domain with no available domain expert, and want to get all of the benefits of a BN model. It is also useful for data-mining style applications, where there are masses of data available and we would like to interpret them. In addition to providing a model that will allow us to predict behavior of cases that we have not seen, the structure also gives the expert some indication of what attributes are correlated. The algorithms for this problem are combinatorially expensive. They basically reduce to a heuristic search over the space of BN structures.

There has been some attention given to the problem of unknown network structure in the literature. The key aspect of the problem is to reconstruct the topology of the network from fully observable variables. In the literature, this is considered as a discrete optimization problem solved by a greedy search algorithm in the space of structures. Some examples of the greedy search algorithm can be found in [5, 6].

A MAP (Maximum a Posterior) analysis of the most likely network structure has been studied in [5] and [6] when the data are fully observable. The resulting algorithms are capable of recovering fairly large networks from large data sets with a high degree of accuracy. On the other hand, they usually adopt a greedy approach to choosing the set of parents for a given node because the problem of finding the best topology is intractable.

There are two main approaches to structure learning in *BNs*:

Constraint based: Perform tests of conditional independence on the data, and search for a network that is consistent with the observed dependencies and independencies.

Score based: Define a score that evaluates how well the (in) dependencies in a structure match the data, and search for a structure that maximizes the score.

Constraint-based methods are more intuitive. They follow the definition of a *BN* more closely. They also separate the notion of the independence from the structure construction. The advantage of score-based methods is that they less sensitive to errors in individual tests. Compromises can be made between the extent to which variables are dependent in the data and the cost of adding the edge.

The score-based methods operate on the same principle: a scoring function is defined for each network structure, representing how well it fits the data. The goal is to find the highest-scoring network structure. The space of Bayesian networks is a combinatorial space, consisting of a super exponential number of structures. Thus, it is not clear how one can find the highest-scoring network even with a scoring function. In general, the problem of finding

the highest-scoring network structure is NP-hard. On the other hand, the problem of searching a combinatorial space with the goal of optimizing a function is very well studied in AI literature. Consequently, the answer is to define a search space, and then do heuristic search.

In light of the above statements, a *BN* structure learning algorithm requires the following components be determined:

i) Scoring function for different candidate network structures.

ii) The definition of the search space: operators that take one structure and modify it to produce another.

iii) A search algorithm that does the optimization search.

Each component will be discussed separately. The three main scoring functions commonly used to learn Bayesian networks are the log-likelihood [13], the one based on the principle of minimal description length (MDL) [11]which is equivalent to Schwarz' Bayesian information criterion (BIC), and Bayesian score [3,13].

The log-likelihood function is simply the log of the likelihood function. That is,

$$l(D|B,\theta_B) = logL(D|B,\theta_B) \tag{11}$$

The log-likelihood is easier to analyze than the likelihood, because the logarithm turns all the products into sums. Therefore,

$$L(D|B,\theta_B) = \prod_m P(d[m]|B,\theta_B) \tag{12}$$

and, the following equation can be written:

$$L(D|B,\theta_B) = \sum_m logP(d[m]|B,\theta_B) \tag{13}$$

There are a couple of important things to note about the log-likelihood. The log-likelihood increases linearly with the length of data, M. The higher scoring networks are those where the node and the parents are highly correlated. Adding a node to the networks always increases the log-likelihood. As a result, the network structure that maximizes the likelihood is often the fully connected network. This is the deficiency of the log-likelihood score and is not desired. Thus, a score that makes it harder to add edges is necessary. In other words, we would like to penalize structures with too many edges.

One possible formulation of this idea is called the MDL score. It is defined as:

$$Score_{MDL}(B:D) = l(D|B,\hat{\theta}_B) - \frac{logM}{2}Dim(B) - DL(B) \tag{14}$$

Where $Dim(B)$ is the number of independent parameters in B and $DL(B)$ is the number of bits (the description length) required to represent the structure of .The abbreviation MDL stands for minimum description length. The MDL score is a compromise between fit to data and model complexity. Adding a variable as a parent causes the log-likelihood term to increase, but so does the penalty term.[6] There will be an edge addition if its increase to the likelihood is worth it.

Another commonly used score is called Bayesian score. In this case, the network score is evaluated as the probability of the structure given the data. The Bayesian score has the following form:

$$Score_{BDE}(B:D) = P(B|D) = \frac{P(D|B)P(B)}{P(D)} \tag{15}$$

As usual $P(D)$ is constant, so it can be ignored when different structures are compared. Therefore, the model maximizes $P(D|S)P(S)$, where S represents a structure. The ability to ascribe a prior over structures gives us a way of preferring some structures to others. Here, the probability $P(D|B)$ can be calculated as

$$P(B|D) = \int P(D|\theta_B,B)P(\theta_B|B)d\theta_B \tag{16}$$

From Equation (16), one can see that the more parameters we have the more variables we are integrating over. As a result, each dimension causes the value of the integral go down because the "hill" of the likelihood function is a smaller fraction of the space. Therefore, this idea gives preference to networks with fewer parameters. It can be shown that the Bayesian score is a general form of MDL score. The MDL score can be viewed as an approximation of the Bayesian score. Therefore, the Bayesian score is also a compromise between the model complexity and fit to the data.

Several ways of scoring different Bayesian network structures have been explained. Different scores have been explored in terms of the network complexity and how the network fits to the correlation in the data. Now, the goal is to find the network that has the highest score. In other words, training data D, the scoring function, and a set of possible structures are the inputs of the search algorithm while the desired output is a network that maximizes the score. It can be shown that finding maximal scoring network structures where nodes are restricted to having at most k parents is NP-hard for any $k > 1$. Therefore, a heuristic search is resorted to for this optimization problem. A search space is defined, where the states in the space are possible structures and the operators denote the adjacency of structures. This space is traversed looking for high-scoring functions to complete the optimization. The obvious operators in the search spaces are add an edge, delete an edge, and reverse an edge. The search starts with some candidate network, which may be the empty one, or one that some expert has provided as a starting point. [7,8]Then, applying the operators, the high-scoring network is searched in the space. The parameters of the network are calculated by using training data D.

The most commonly used algorithm for optimization search is simple greedy hill climbing. Even though the

hill-climbing method is commonly used, it has several key problems such as local maxima where all one-edge changes reduce the score and plateaus where a large set of neighboring networks that have the same score. There are some clever tricks that avoid some of these problems such as TABU-search, random restart, and simulated annealing. In general, greedy hill climbing with random start works quite well in practice. In a world, we examined methods for learning a Bayesian network from fully observable data in this section.

Beside this, unknown structure and unobservable variables is the most difficult case to resolve because the structure of the networks is unknown and the variables are not fully observable. There is no significant amount of research for this case. When some variables are sometimes or always unobserved. There are two recently developed methods that recover the Bayesian network structure with unobserved variables.

The first algorithm was proposed by Russell and is called structural EM(SEM) algorithm. The algorithm combines the standard EM algorithm, which optimizes the network parameters, with structure search for model selection. The main idea of this method is that it attempts to maximize the expected score of models instead of their actual scores at each iteration. Russell proves a theorem that the SEM algorithm makes 56 progress in each iteration on finding the better scoring network. Then, he states that if one chooses a model that maximizes the expected score at each iteration, then a better choice is provably made in terms of the marginal score of the network. The SEM algorithm is exciting since it attempts to directly optimize the true Bayesian score within EM iteration rather than an asymptotic approximation.

The most problematic aspect of SEM is that it might converge to a sub-optimal model. This could happen if the model generates a distribution that causes other models to appear worse when the expected score is examined. This difficulty becomes more obvious when the ratio of missing information is higher. Russell suggests that, in practice, the algorithm needs to be run from several starting points to get a better estimate of the MAP model. Another restriction of the SEM is that it focuses on learning a single model. In practice, several high scoring models is necessary for better prediction.

Additional to this deficiency, the algorithm requires large number of computations during learning.[9,10,11] This is the main problem in applying this technique to large-scale domains. The following paragraphs provide a computationally cheaper method.

The second algorithm was proposed by Sebastiani and Marino. They were able to show that BC algorithm could also learn the structure of the network with small changes in the algorithm. [12,13]The algorithm has the following form:

Pick a random network structure B as starting point

Pick parameters for the network structure B

Compute score for B Repeat

Add an edge to the network, the network B' is created

Estimate the posterior expectations of parameters of B' using BC method

Estimate the posterior values of the network parameters

Compute score for the network with B'

If $score(B) > score(B)$

Then let $B := B'$ Else return(B)

This method is very similar to the search method described where we had fully observed data. The only difference is that, in this case, we have partially observed data or incomplete data. Therefore, the estimation of the parameters of the network is also necessary. The BC method is employed to estimate the parameters of the network. The estimation process is performed in each step, i.e., after adding each edge to the network. Consequently, the method involves both parameter learning and structure learning. However, the main attention was given to the parameter estimation part since it is newly discovered method. The structure learning part can be modified as a greedy search algorithm. In that case, ”delete an edge” operator and ”reverse an edge” operator have to be incorporated to the algorithm.

There is a slight difference between SEM and BC methods and the problem of self-organizing agents in terms of required data structure. The SEM and BC algorithms require a certain minimum length database. Unfortunately, there will not be a prior database to work with at the beginning of the agents' exploration of the environment. Thus our learning method has to be online: estimation of the network structure and parameters will be performed simultaneously with the gathering of new entries in the database. So, our method has to learn the network while the agents are exploring the environment and organizing themselves to manage a common task. Using the current methods this problem cannot be solved because they do not contain an online learning algorithm. In the next chapter we propose a method that allows the agents learn the environment while they are exploring the environment and organizing a common task.

4 Conclusions

The process of learning Bayesian networks takes different forms in terms of whether the structure of the network is known and whether the variables are all observable. The structure of the network can be known or unknown, and the variables can be observable or hidden in all or some of the data points. Learning Bayesian networks can also be examined as the combination of parameter learning and structure learning. Parameter learning is estimation of the conditional probabilities (dependencies) in the network. Structural learning is the estimation of the topology (links) of the network. Known network structure and observable variables(complete data) is the easiest and the most studied case of learning Bayesian networksin the unknown network structure and observable variables the inducer is given the set of variables in the model, and

needs to select the arcs between them and estimate the parameters.

Acknowledgements

This work is financially supported by the National Natural Science Foundation of China (Project No. 90718038). Thanks for the help.

References

[1] Singh, Jagdip, "Performance Productivity and Quality of Frontline Employees in Service Organization," Journal of Marketing, 64 (4), 15-34 (2009).
[2] Jong, Ad de, Ko de Ruyter, and Jos Lemmink, "Antecedents and Consequences of the Service Climate in Boundary-Spanning Self-Managing Service Team" .Journal of Marketing, 68 (2), 18-35 (2010).
[3] Louis, Meryl R. 0and Robert I. Sutton, "Switching Cognitive Gears: From Habits of Mind to Active Thinking," Human Relations, **44** (1), 55-76(2010).
[4] Argyris, C. and D.A. Schon, Organizational Learning: (Vol. **2**) Theory, Method, and Practice. Reading, MA: Addison-Wesley(2011).
[5] Friedman, Victor J., "The Individual as Agent of Organizational Learning," in Handbook of Organizational Learning and Knowledge, Meinolf Dierkes and Ariane Berthoin Antal and John Child and Ikujiro Nonaka, Eds. Oxford: University Press(2001).
[6] Tyre, M. J. and W.J. Orlikowski, "Windows of Opportunity: Temporal Patterns of Technological Adaptation in Organizations," Organization Science, **5**, 98-118(1994).
[7] Edmondson, Amy, "Psychological Safety and Learning Behavior in Work Teams," Administrative Science Quarterly, **44** (2), 350-383(1999).
[8] Zeithaml, Valarie A. and Mary Jo Bitner, Services Marketing. New York: McGraw Hill(1996).
[9] M.Ramoni and P.Sebastiani,"Learning Bayesian networks from incomplete data, "Technical Report KMi-TR-43,Knowledge Median Institute,The Open University, February 1997.
[10] J. Pearl, "A probabilistic calculus of actions,"in Proceedings of the Tenth Conference on Uncertainty in AI(UAI-94), San Mateo, CA: Morgan Kaufmann,1994.
[11] G. Shafer and J. Pearl, Readings in Uncertain Reasoning. San Mateo, CA: Morgan Kaufmann, 1990.
[12] S. Russell and P. Norvig, Artificial Intelligence :A modern Approach, New Jersey: Prentice Hall,1995.
[13] T. Malsch and I. Schulz-Schafer, "Generalized media of interaction and interagent coordination, "in Socially Intelligent Agents-Papers from the 1997 AAAI Fall Symposium, Technical Report FS-97-02, AAAI, 1997.

Dr. Wei Xiong is professor of the school of management Zhejiang University, doctoral tutor. His research interests are in the areas of QFD, routing systems, and information systems.

Yonghui Cao received the MS degree in business management from Zhejiang University in 2006. He is currently a doctorate candidate in Zhejiang University. His research interest is in the areas of management information systems.

Hui Liu received the MS degree in computer science and technology from Henan Normal University in 2008. He is currently a senior lecturer in Henan Normal University. Her research interest is in the areas of machine learning and computer network.