Distributed Bayesian Network Structure Learning

Yongchan Na, Jihoon Yang
Sogang University
{ycna, yangjh}@sogang.ac.kr

Abstract- **We propose a new method for learning the structure of a Bayesian network from distributed data sources. Traditional Bayesian network learning takes place at the central site with all data. In many cases, data are distributed over different sites and gathering them at one place is not practical.**

Our algorithm starts with individual learning at each site with the local data. Then it transmits the learned Bayesian network to the central site. Last, the central site determines the final Bayesian network by looking for frequently occurring parts among the aggregated structures. Experimental results verify that our algorithm successfully finds the same structure that the centralized algorithm produces, with comparable classification accuracy and even higher learning speed.

I. Introduction

The Bayesian network (BN) [1] is one of the representative machine learning algorithms that are in wide use nowadays. Therefore, it has become of interest to develop efficient and accurate learning algorithms for BNs to be used in data mining. Learning in a BN includes both the determination of the network structure and the parameter estimation for the variables in the network. In this paper, we call this structure learning.

A variety of learning algorithms for the BN have been proposed in the literature [1, 2, 3, 4]. Generally, the learning algorithms are for centralized data. However, we frequently find distributed data that are constantly generated in many industrial domains (e.g. bank transactions, discount store sales, medical records). Mining such huge amounts of distributed data using the conventional centralized learning algorithm is not practical since shipping all the data to the central site causes exorbitant overhead, let alone the issues related to privacy and security. Instead, we can mine the distributed data at each site and only send the resulting model, a BN in this case, which would be even smaller than the data, to the central site for integration. The design of distributed BN learning algorithms is thus required.

In this paper, we propose a simple but efficient distributed BN structure learning algorithm. Our algorithm goes through the following procedure. First, a simple (or central) structure learning algorithm is applied to the local data to produce a BN at each site (The K2 algorithm [2] is currently used for this purpose, but any other simple structure learning algorithm can be adopted as well). Then these local BNs are transmitted to the central site where the final BN is generated by finding the frequently occurring parts among the local structures. The idea of making the final structure based on the common parts is legitimate since all the sites are presumed to have data of the same characteristics in the same domain.

The rest of the paper is organized as follows: Section Ⅱ introduces background knowledge and related work on the structure learning of BNs. Section Ⅲ describes our distributed BN structure learning algorithm. Section Ⅳ presents the experimental results of our algorithm. Section Ⅴ concludes with summary and discussion of some directions for future research.

II. Learning in Bayesian Networks

In this section, we provide a brief overview of the BN and the structure learning algorithms. We also introduce related work on distributed structure learning.

A. Bayesian Network

A BN [1, 3] is a probabilistic graphical model that encodes variables and their dependence relationships into nodes and arcs, respectively. It is formalism for representing and reasoning with such a model for problems involving uncertainty. For this reason, BNs have been used in many domains where chasing the causes and inferring the effects are necessary.

If there is an arc from node A to another node B in a BN, A is called a parent of B, and B is a child of A. The set of parent nodes of a node x_i is denoted by $Pa(x_i)$. A directed acyclic graph is a BN relative to a set of variables if the joint distribution of the node values can be written as the product of the distributions of each node and its parents:

$$P(x_1, x_2, \dots, x_n) = \prod_{i=1}^{n} P(x_i | Pa(x_i)) \qquad (1)$$

BNs can be acquired from data or domain experts' knowledge. For complex domains, a number of learning algorithms that generate BNs from the data and estimate the parameters have been proposed [1, 5]. Bayesian structure learning is the method which determines the structure of the network from data in addition to the accompanying parameter estimation. In recent years, learning such a graphical model has become a very popular research topic, and many algorithms have been developed [1, 2, 4, 5, 6, 7, 18, 19]. Since the number of possible structures is huge, structure learning involves high computational complexity, and thus often resorts to heuristic or approximate learning algorithms.

The structure learning algorithms of BNs are generally divided into two categories: *score-based* [2, 4, 6] and *constraint-based* [5, 7]. The score-based approach defines a *score* metric that measures how a structure reflects the data and finds a BN structure with the highest score. The constraint-based approach tries to find a structure that satisfies the dependence relationships in the data.

B. Score-based learning and K2

As aforementioned, one of the most popular methods of inducing BNs from data is *score-based* learning. Assuming that the data are generated from a BN and that all the variables are visible, a search can be attempted to find the structure with the highest score which is calculated by a well-defined scoring function. Some of the commonly used scoring functions include Bayesian scores (e.g. Bayesian Information Criterion) and MDL function (e.g. Huffman code) [1].

The score is related to the posterior probability of a structure G given the data set D. The score-based learning algorithms attempt to maximize this score. By Baye's law, the score can be represented as:

$$Score(G,D) = Pr(G|D) = Pr(D|G)Pr(G)/Pr(D) \qquad (2)$$

One of the popular score-based algorithms is the K2 algorithm developed by Cooper and Herskovits [2] which has polynomial time complexity in the number of variables given an ordering of the variables. K2 algorithm uses both Bayesian evaluation function and Bayesian Information Criterion (*BIC*) [1]. One of the desirable characteristics of K2 is the decomposability in case of no missing values in data. Decomposability means that deciding the parents of a particular node does not affect the decision of the parent nodes of other nodes at all. Due to the decomposability, K2 is able to perform a local search.

```
(1) Input: an ordered set of $m$ nodes, an upper bound $u$ on
    the number of parents for a node.

(2) Output: BN structure

(3) For $i$ = 1 to $m$
    {
      $\pi_i = \emptyset$;
      $P_{old} = f(i, \pi_i)$;
      $KeepAdding = true$;
      While KeepAdding and $|\pi_i| < u$
      {
        let $z$ be the node in $Pred(x_i) - \pi_i$ that
          maximizes $f(i, \pi_i \cup \{z\})$ ;
        $P_{new} = f(i, \pi_i \cup \{z\})$ ;
        If $P_{new} > P_{old}$
            $P_{old} = P_{new}$;
            $\pi_i = \pi_i \cup \{z\}$;
        Else $KeepAdding = false$;
      }
    }
```

Fig. 1. K2 algorithm

So this algorithm is a greedy heuristic approach to efficiently determine a good BN representation of probabilistic relationships between nodes. Initially each node has no parents. It then adds incrementally the parent whose addition most increases the score of the resulting structure. When the addition of no single parent can increase the score, it stops adding parents to the node. K2 does not need to check for cycles and can choose the parents for each node independently under the assumption that a fixed ordering of variable is given. The pseudo-code of K2 is shown in Fig. 1.

In Fig.1, $f(i, \pi_i)$ is the scoring function and it presented as:

$$f(i,\pi_i) = \prod_{j=1}^{q_i} \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)} \prod_{k=1}^{r_i} \alpha_{ijk}! \qquad (3)$$

Where π_i is set of parents of node x_i, q_i is $|\phi_i|$. The ϕ_i is list of all possible instantiations of the parents of x_i in dataset D. The r_i is $|V_i|$. The V_i is list of all possible values of the attribute x_i, α_{ijk} is number of cases in D in which the attribute x_i is instantiated with its k^{th} value and the parents of x_i in π_i are instantiated with the j^{th} instantiation in ϕ_i. The N_{ij} equals $\sum_{k=1}^{r_i} \alpha_{ijk}$. That is the number of instances in the dataset in which the parents of x_i in π_i are instantiated with the j^{th} instantiation in ϕ_i.

C. Related work on distributed structure learning

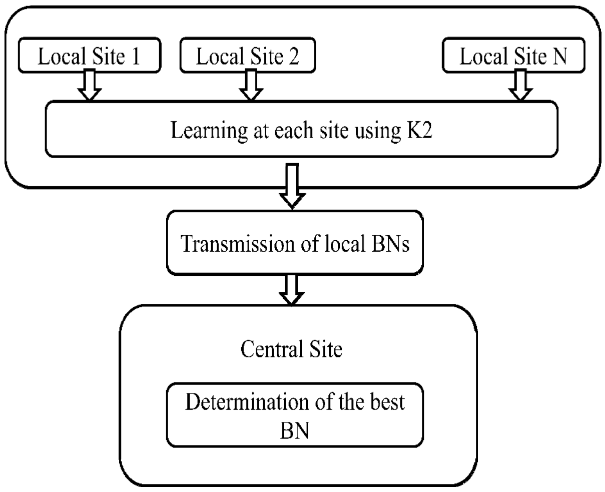
Distributed structure learning is a new, but active research area in BN structure learning. We introduce the following representative algorithms for distributed learning [9, 10, 11, 12, 20].

Chen *et al.* [9] proposed a collective method for learning the structure of a BN from distributed, heterogeneous data. This algorithm finds a cross link of cross variable after local site learning. The cross variable means that variable x_i and its parents are not in the same site. The cross learning is to identify the links whose vertices are in different sites. Combining the results, the algorithm puts together the local BNs and the BN learnt from cross learning and also remove any extra local links.

Gou *et al.* [10] introduced another distributed learning method. In this algorithm, each site has the same features but different observations. In other words, this algorithm is for distributed, homogeneous data. The algorithm goes through two steps of *local* and *global* learning. In local learning, the algorithm produces local BNs using *BN Power Constructor system* which consist of d-separation and conditional independency (CI) test. Then in global learning, local structures are combined into the final structure with CI tests.

III. Proposed algorithm

Fig. 2 depicts the flow of our algorithm. Our algorithm consists of three phases: local site learning, transmission of local BNs to the central site, and central learning.



Local Site 1
Local Site 2
Local Site N
Learning at each site using K2
Transmission of local BNs
Central Site
Determination of the best BN

Fig. 2. Our distributed BN structure learning algorithm

A. Local site learning

In this phase the K2 algorithm is applied to the local data to learn the local BN structure. Note that only the variables appearing in the data are included in the BN. The BNs generated at different sites might have different set of variables though the distributed data are from the same domain. Each local site produces a BN structure based on its data, as represented by an adjacency matrix R_i.

B. Transmission of local BNs to the central site

The BNs produced at each site are transmitted to the central site where they are used to determine the final, global BN that reflects all the data in the network. It is natural that the amount of transmitted information encoding the BN should be much less than the data itself. This in fact is the main motivation for our distributed BN structure learning algorithm.

C. Central site learning

The last step of our algorithm is to elicit the global BN that explains all the distributed data best. We apply the voting strategy to the local BNs represented as adjacency matrices for easy implementation. By counting the number of 1's in each matrix for a pair of nodes (i.e. the connectivity between two nodes in the graph which shows the parent-child relationship) and checking if majority has the connectivity, we can easily decide the existence of the edge in the global BN. Repeating this for all the variables given the ordering, we can finalize the global BN. In case particular nodes are missing in a local BN, all the edges connecting them are presumed to be non-existing.

Our algorithm is summarized in Fig. 3.

```
(1) Input: Distributed data and node ordering at $N$ sites

(2) Output: BN structure
            (Represented by the adjacency matrix $R$)

(3) Local sites: Compute local BNs
    For $i = 1$ to $N$ sites
    {
        Apply K2 to data and produce $R_i$;
        Transmit $R_i$ to the central site;
    }

    Central site: Determine global BN by majority voting
    {
        For $i=1$ to $N$
        {
            Receive $R_i$ from local site;
            $R[i, j]=R[i, j]+ R_i [i, j]$;
        }

        {
            For $i=1$ to $m$ nodes
            {
                For each node $j$ in $Pred(x_i)$
                {
                    If (the count for edge $(i, j)) / N > 0.5$
                        $R[i, j] = 1$;   // connect $i$ & $j$
                    else
                        $R[i, j] = 0$;
                }
            }
        }
    }
```

Fig. 3. Pseudo-code of our algorithm

IV. Experiments

A. Dataset

We tested our algorithm using some of the commonly used real world domains such as the ASIA [13], CAR [14], CANCER [15] and ALARM networks [16].

The ASIA is a small BN that calculates the probability of a patient having tuberculosis, lung cancer or bronchitis respectively based on different factors - for example whether or not the patient has been to ASIA recently. The CAR dataset represent a starting problem of automobile. The CANCER network is simple. The nodes represent genes and arcs represent causal pathways.

Especially the ALARM network is one of the benchmark data used for evaluating BN learning algorithms. The ALARM network is a medical diagnostic system for patient monitoring, which includes nodes for 8 diagnoses, 16 findings and 13 intermediate variables. Each variable has two to four possible values.

We generated six datasets (one with 10,000 samples and five with 2,000 samples randomly chosen from the 10,000 samples) for each domain. The sample was randomly generated using the structure and conditional probability distribution. We used Kevin Murphy's BN toolbox implemented in Matlab [17] for both the data generation and the experiments.

Table Ⅰ summarizes the original network of datasets used in our experiments.

TABLE Ⅰ
DATASETS

Name	Number of nodes	Number of arcs
CANCER	5	5
ASIA	8	8
CAR	12	9
ALARM	37	46

B. Experimental setup

The performance measures used in our experiments are evaluated based on the learning time and the accuracy. The former is the execution time for the algorithm given the samples, and the latter is the percentage of the similarity about the resulting network and the original network in each of the CANCER, ASIA, CAR and ALARM network.

To evaluate the performance of our proposed algorithm, we conducted experiment on three cases.

First, we generated a BN which used the proposed algorithm with 10,000 samples. Subsequently, we produced a BN with 2,000 samples. The centralized learning was used in both cases. We then compared the performance of the centralized learning with different sample sizes.

Second, we compared the BN produced with 10,000 samples with the original BN. Again, this approach is based on centralized learning.

Finally, we compared the BNs of proposed distributed algorithm with five 2,000 sample sets with the original BN.

The first and second settings are based on centralized learning, and the last setting is based on distributed learning.

C. Results

Table Ⅱ shows the result of centralized learning with 2,000 and 10,000 samples learned at one site. These results indicate that the large number of samples yield higher accuracy though it takes longer learning time.

Table Ⅲ compares the performance of centralized learning and distributed learning. The 10,000 training samples were used in the former, while 2,000 samples were assumed to be distributed at five sites in the latter. The learning time of distributed learning method is the average time of five sites. As we can see, the distributed learning is a lot faster (almost seven times) than the centralized learning. In terms of the accuracy, our proposed distributed algorithm is fairly comparable to the centralized algorithm. In particular, we know that result of the ALARM network presents better accuracy than the centralized case. During the learning process missing and extra arcs will occur frequently in distributed learning because the ALARM network is large and complex. Our proposed algorithm seems to make up for this problem by voting process, and to improve the accuracy.

TABLE Ⅱ
ACCURACY & TIME OF CENTRALIZED LEARNING

Dataset	Number of samples	Accuracy(%)	Learning time(sec)
CANCER	2000	96.0	38
	10000	100.0	233
ASIA	2000	96.9	115
	10000	98.4	790
CAR	2000	97.9	236
	10000	100.0	1848
ALARM	2000	92.9	3812
	10000	93.2	28980

TABLE Ⅲ
RESULT OF CENTRALIZED VS. DISTRIBUTED LEARNING

Dataset	Learning method	Accuracy(%)	Learning time(sec)
CANCER	Distributed	100.0	38
	Centralized	100.0	233
ASIA	Distributed	98.4	115
	Centralized	98.4	790
CAR	Distributed	99.3	242
	Centralized	100.0	1848
ALARM	Distributed	94.9	3847
	Centralized	93.2	28980

Our proposed algorithm is simpler than other distributed approaches introduced in section Ⅱ. When the results of learned network move from a local site to the central site, many nodes and arcs are moved. From the standpoint of load of movement, our algorithm moves only one BN to the central site. It means that if the number of distributed sites is large, our algorithm has even smaller movement of BNs. Since a direct comparison between our algorithm and other distributed algorithms is not feasible (without detailed descriptions on experimental setups) , we made a comparison among the algorithms only with the results given in [9, 10] and found that our algorithm produced a BN with the same accuracy.

In summary, our distributed BN structure learning. algorithm is shown to produce comparable or higher classification accuracy with even less computational overhead than the centralized algorithm as well as distributed algorithms.

V. CONCLUSION AND FUTURE WORK

A. Conclusion

We proposed a new method to address the problem of learning the structure of a BN from distributed data sources. The traditional BN structure learning has been developed by the method which is learned only at one place. It means that all data is located and learned at a single, central site.

Unfortunately, it does not mean that the real data is gathered at one place. The cost of data communication is a significant factor in a large number of distributed data sources. Some of the communication overhead includes heavy network traffic, security issues, etc.

We proposed a simple but efficient distributed BN structure learning algorithm. Our algorithm consists of three phases. First, local site learning is carried out by the K2 algorithm. After local site learning, each site transmits the resulting BN to the central site. Finally, the best global BN structure is elicited from the local BNs that are transferred from the local sites. This is done by the majority voting on the arcs in the local BNs.

We used four real-world datasets for the experiments. The datasets are ASIA, CAR, CANCER and ALARM. Our proposed algorithm produced comparable or improved classification accuracy than other approaches with significantly reduced computational overhead.

B. Future work

Currently, our dataset is homogeneous. However, real data sites may be heterogeneous. In other words, sites may contain tables with different schema. Extending the algorithm to handle heterogeneous data is thus of interest. In addition, instead of the K2 algorithm, we can also consider different score-metrics such as Fisher's z test, chi square [18] and mutual information [19].

VI. ACKNOWLEDGMENTS

This work was supported by the grants of Sogang university and Korea Science Foundation.

REFERENCES

[1] D. Heckerman, “A tutorial on learning with Bayesian networks”, *Technical Report MSR-TR-95-06*, Microsoft Research, Redmond, Washington, 1995.

[2] F. Gregory, Cooper and Edward Herskovits, “A Bayesian method for the induction of probabilistic networks from data”, *Machine Learning*, 1992, 9(4), pp. 309-347.

[3] J. Pearl, “Fusion, propagation and structuring in belief networks”, *Uncertainty in Artificial Intelligence*, 1986, pp. 357-370.

[4] D. Heckerman,D. Geiger and D. M. Chickering, “Learning Bayesian Networks: The Combination of Knowledge and Statistical Data”, *KDD Workshop*,1994, pp. 85-96.

[5] J. Cheng, R. Greiner, J. Kelly, D. A. Bell, and W. Liu, "Learning Bayesian networks from data : An information theory based approach.," *Artificial Intelligence*, 2002, vol. 137, no. 1-2, pp. 43–90.

[6] M. Singh, M. Valtorta, “Construction of Bayesian network structures from data : a brief survey and an efficient algorithm”, *International Journal of Approximate Reasoning*, 1995, pp. 111– 131.

[7] G. F. Cooper, “A Simple Constraint-Based Algorithm for Efficiently Mining Observational Databases for Causal Relationships”, *Data Mining and Knowledge Discovery*, 1997, vol. 1, pp. 2203–224.

[8] G. Schwarz, “Estimating dimension of a model”, *Annual Statistics*, 1978, vol. 6, pp.461–464.

[9] R. Chen, K. Sivakumar, and H. Kargupta, “Learning Bayesian network structure from distributed data”, in *Proceedings of SIAM International Conference on Data Mining*, 2003. pp.284-288.

[10] Kui Xiang Gou, Gong Xiu Jun, Zheng Zhao, "Learning Bayesian Network Structure from Distributed Homogeneous Data", in *Proceedings of the eighth ACIS International Conference*, 2007, vol. 3, pp. 250-254.

[11] Rebecca Wright, Zhiqiang Yang, “Privacy Preserving Bayesian Network Structure Computation on Distributed Heterogeneous Data”, *ACM KDD'04,* 2004, pp. 713-718.

[12] R. Tedesco, P. Dolog, W. Nejdl & H. Allert, “Distributed Bayesian Networks for User Modeling”, in *Proceedings of World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education,* 2006, pp. 292-299.

[13] Robert G. Cowell, A. Phillip Dawid, Steffen lauritzen, and David Piegelhalter, *"Probabilistic networks and expert systems"*, Springer Verlag 1999.

[14] D. Heckerman, K. Rommelse, "Troubleshooting under uncertainty", *Technical Report MSR-TR-94-07*, Microsoft Research, Redmond, Washington, 1994.

[15] Friedman, Nir and Murphy, Kevin and Russell, Stuart, "Learning the structure of dynamic probabilistic networks", in *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, 1998, pp. 139-147.

[16] I. Beinlich, H. Suermondt, R. Chavez, and G. Cooper, “The ALARM monitoring system : A case study with two probabilistic inference techniques for belief networks”, in *Proceedings of the Second European Conference on Artificial Intelligence in Medicine*, 1989, pp. 247–256.

[17] Kevin murphy.(2007, October 19) [online]. Available: http://www.cs.ubc.ca/~murphyk/Software/BNT/bnt.html

[18] Abramowitz, Milton & Stegun, Irene A., *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, Courier Dover Publications, 1965.

[19] Guiasu, Silviu, *“Information Theory with Applications”*, McGraw-Hill, New York, 1977.

[20] Shen, J. and Lesser, V., “Communication management using abstraction in distributed Bayesian networks”, In *Proceedings of the Fifth international Joint Conference on Autonomous Agents and Multiagent Systems* (Hakodate, Japan, May 08 - 12, 2006). AAMAS '06. ACM, New York, NY, pp.622-629.