Learning Bayesian Network Structure from Distributed Data

R. Chen* K. Sivakumar† H. Kargupta‡

Abstract

We propose a collective method to address the problem of learning the structure of a Bayesian network from a distributed heterogeneous data sources. In this case, the dataset is distributed among several sites, with different features at each site. The collective method has four steps: local learning, sample selection, cross learning, and combination of the results. The parents of local nodes can be correctly identified in local learning. The main task of cross learning is to identify the links whose vertices are in different sites (cross links). This is done by transmitting a small subset of samples from each local site to a central site. The combination step involves removing extra links from local Bayesian networks that may be introduced during local learning due to the well known hidden variable problem. The sample selection step selects samples, based on a likelihood criterion, that are possibly evidence of cross links. The overall procedure is called collective learning. Experimental results verify that, for sparsely connected networks, the collective learning method can learn the same structure as that obtained by a centralized learning method (which simply aggregates data from all local sites into a single site).

1 Introduction

In recent years, there has been a number of works on the Bayesian Network (BN) structure learning. In general, these structure learning algorithms are developed for the centralized case, where all the data is in a single site. However, there are many scientific and non-scientific applications in which the observed dataset is distributed among different sites. For example, the NASA Earth Observing System (EOS) generates more than 100 gigabytes of image data per hour, which are stored, managed, and distributed from eight Distributed Active Archive Centers (DAACs). Cost of data communication between the distributed databases is a significant factor in an increasingly mobile and connected world with a large number of distributed data sources. Other factors like limited bandwidth, privacy, and data security might also prevent the aggregation of data at a single site.

In this paper, we consider a distributed heterogenous dataset scenario, where each site has observations corresponding to a subset of the attributes. We assume there exists a "key" (like time index, user id, etc.) that is common to all sites and can be used to link an observation among different sites. A naive approach to learn a BN from a distributed heterogenous dataset is to aggregate all data to a central site and learn a BN from the merged data. We will refer to this as the centralized learning method. The BN learnt from this method will be denoted by B_{cntr}. However, in many situations, limited network bandwidth and/or data security might render this approach infeasible.

In our earlier work [CSK01b, CSK01a, CSK02], we proposed an approach to learn a BN from distributed heterogenous data. Both the distributed learning method proposed in this paper and that proposed in [CSK01b, CSK01a, CSK02] are within the framework of Collective Data Mining [KPHJ00]. In [CSK01b, CSK01a], the focus of our work was on distributed BN parameter learning although we gave some ideas about distributed structure learning. In this paper, we focus on the distributed BN structure learning. A collective method is proposed to solve the structure learning problem. To our knowledge, there is no significant work that addresses the problem of BN structure learning from distributed heterogenous data. The remainder of this paper is structured as follows. Section 2 provides an overview of BN, structure learning for BN, and discusses related literature. Section 3 discusses the proposed collective learning algorithm. Experimental results are presented in Section 4. We conclude with some discussions in Section 5.

2 Bayesian Network, Structure Learning, and Related Work

In this section, we provide a brief overview of BN and the structure learning algorithms. We then review some related literature.

2.1 Bayesian Network A Bayesian Network (BN) is a probabilistic graph model that can be defined as a

*School of EECS, Washington State University
†School of EECS, Washington State University
‡Department of CSEE, University of Maryland Baltimore County

pair B=($\mathcal{G}$, θ). Here, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a directed acyclic graph (DAG) that represents the structure of the BN. $\mathcal{V}$ is the node set and $\mathcal{E}$ is the edge set. For a node $X \in \mathcal{V}$, a parent of X is a node from which there exists a directed link to X. The set of parents of X is denoted by $pa(X)$.

In this paper, we consider a BN over N discrete variables X_i ($1 \leq i \leq N$). Conditional probability $\theta_{ijk} = P(X_i = k \mid pa(X_i) = \mathbf{j})$ is the probability of variable X_i in state k when $pa(X_i)$ is in state $\mathbf{j}$. If a variable X_i has no parents (root node), then θ_{ijk} corresponds to the marginal probability of node X_i. We will denote by θ the set of all parameters θ_{ijk}. A specific ordering of the variables in a BN is called a node ordering.

2.2 Structure Learning Structure learning is an active topic of research in the field of BN. Structure learning is a model selection problem in the sense that each structure corresponds to a model (for which parameters have to be estimated), and we need to select a model based on the data. In general, there are two approaches to learn the structure of a BN. The first approach is a dependence analysis method that poses learning as a constraint satisfaction problem. The algorithms in this approach try to discover the dependencies from the data, usually using a statistical hypothesis test. The algorithm proposed in [PV91] uses this approach. The second approach is a searching and scoring based method that poses the learning as an optimization problem. This kind of algorithm defines a "score" that describes the fitness of each possible structure to the observed data. Commonly used scores include Bayesian score [CH92, Hec98] and MDL score [Suz93]. Then the structure learning problem becomes an optimization problem: find the structure S_{opt} that maximizes (or minimizes depending on how the score is defined) the score. An important property of some score functions is decomposability. That is, the score function can be decomposed as follows:

(2.1)
$$Score(S, D) = \sum_i Score(X_i, pa(X_i), D(X_i, pa(X_i))).$$

Here S denotes the BN structure, D denotes the entire data, and $D(X_i, pa(X_i))$ denotes the data involving only X_i and $pa(X_i)$.

The most widely used structure learning algorithm is the K2 algorithm [CH92]. It belongs to the second approach. The structure learning problem can be stated as follows: Given the complete training dataset D (no missing value) and a node order, find a network structure S that best matches D. Suppose the prior of the parameters (when the structure is fixed) is Dirichlet: $p(\theta \mid S) \sim Di(\alpha_{ij1}, \alpha_{ij2}, \ldots, \alpha_{ijr_i})$. Let N_{ijk} be the number of samples in D for which $X_i = k$ and $pa(X_i) = \mathbf{j}$. Then the posterior distribution is also Dirichlet:

$$p(\theta \mid S, D) \sim Di(\alpha_{ij1}+N_{ij1}, \alpha_{ij2}+N_{ij2}, \ldots, \alpha_{ijr_i}+N_{ijr_i}).$$

We can then write

$$p(D \mid S) = K2(S, D) = \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{\Gamma(\alpha_{ij})}{\Gamma(\alpha_{ij} + N_{ij})} \times \Big(\sum_{k=1}^{r_i} \frac{\Gamma(\alpha_{ijk} + N_{ijk})}{\Gamma(\alpha_{ij})} \Big) \tag{2.2}$$

and

$$K2(X_i, pa(X_i)) = \prod_{j=1}^{q_i} \frac{\Gamma(\alpha_{ij})}{\Gamma(\alpha_{ij} + N_{ij})} \times \Big(\sum_{k=1}^{r_i} \frac{\Gamma(\alpha_{ijk} + N_{ijk})}{\Gamma(\alpha_{ij})} \Big), \tag{2.3}$$

where $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$ and $\alpha_{ij} = \sum_{k=1}^{r_i} \alpha_{ijk}$. $p(D \mid S)$ is called Cooper-Herskovits scoring function. In this paper, we refer to it as the K2 score since it is the score function of K2 algorithm. Note that the K2 score satisfies the decomposability property.

Having defined a score, the next step is to identify a network structure with the highest score. Generally, this search problem is NP-hard. So we need to use sub-optimal search methods. Most widely used search methods for BN structure learning use the decomposability property. These search methods make a series of arc changes (addition or deletion of one arc at a time). After each arc change, we must check whether the resulting graph S is a valid DAG. For each arc change, we have a score $Score_b$ for the DAG S_b before the change and $Score_a$ for DAG S_a after the change. Acceptance of the change depends on the difference between the two scores. If a score satisfies the decomposability property, we can do the search node by node. For each node, only $Score(X_i, pa(X_i)_a, D(X_i, pa(X_i)_a))$ needs to be evaluated and not the whole score. This can simplify the computation considerably.

2.3 Related Work Mining from heterogeneous data constitutes an important class of Distributed Data Mining (DDM) problems. Kargupta et. al. [KPHJ00] proposed the *Collective Data Mining* (CDM) framework for data mining from distributed heterogenous data. For learning BN from distributed datasets, Kenji [Ken97] introduced an algorithm that can handle the homogeneous distributed learning scenario (here, each site contains data for all the variables, but for a subset of the

observations). In [CSK01b, CSK01a, CS02], a collective method that deals with the heterogenous case is proposed.

3 Collective Structure Learning Algorithm from a Distributed Heterogeneous Database

We first provide an overview of the proposed algorithm. Details of some of the steps will be explained later.

The main steps in our collective method are as follows: (a) Local learning: Learn local BN (local model) involving the variables observed at each site based on local data set. (b) Sample selection: At each site, based on the local BN, identify the samples that are most likely to be evidence of coupling between local and non-local variables. The selection process is based on the likelihood of each sample under the local BN model. Transmit the index of low likelihood samples from each local site to the central site. At the central site, compute the intersection of these index sets and obtain samples corresponding to this intersection set from all the local sites. (c) Cross learning: At the central site, a limited number of observations of all the variables are now available. Using this dataset, learn a new BN (both structure and parameter). This BN would contain links involving variables across different sites. (d) Combination: Combine the local models with the cross links to obtain a collective BN B_{coll}. We shall now define some terminology.

There are two types of variables in a distributed BN. If a variable X_i and its parents are all in the same site, then X_i is called a *local variable*. Otherwise X_i is called a *cross variable*. If an edge $X \rightarrow Y$ whose parent variable X and child variable Y are in different sites, this edge is referred to as a *cross link*. Otherwise it is called a *local link*. From the description of the collective learning procedure, our algorithm includes four main steps: local learning, sample selection, cross learning, and combination. Local learning and cross learning steps are both similar to a centralized BN learning problem. In this paper, the structure learning algorithm which is used for local learning and cross learning step is called the base structure learning algorithm. We use the K2 algorithm as our base structure learning algorithm.

3.1 Local Learning

Local learning step consists of learning (structure and parameters of) the local BN (local model) involving the variables observed at each site based on local data set. For local learning, we can show the following proposition.

PROPOSITION 3.1. (LOCAL NODE) *For a local variable, a base structure learning algorithm with decomposability property can find the correct local structure for this variable.*

Proof: Let X be a local variable. For B_{cntr}, the search space $S(X)$ of local structure of X is the set of all subsets of $Pred(X)$ and the dataset is D. Let pa_{best} be the parent set that maximizes $score(X, pa(X), D)$. Suppose variable X is in site A. The search space $S_{local}(X)$ of local structure of X in site A is the set of all subsets of $\{Pred(X) \setminus \{Y : Y \notin NodeSet(A)\}\}$. The dataset is $D_{local}(A)$. Since the local structure learning algorithm has decomposability property, the score of any candidate parent set in $S_{local}(X)$ with respect to $D_{local}(A)$ is the same as that in $S(X)$ with respect to D. This is clear from equation (2.3) of K2 algorithm. For a local variable, all parents are in the same site, so pa_{best} is also in $S_{local}(X)$. Since pa_{best} can maximize $score(X, pa(X))$ and $score(X, pa(X), D) = score_{local}(X, pa(X), D_{local})$, the optimization result of local structure learning is also pa_{best}. Therefore, local learning can find the correct structure for X. This concludes the proof.

For cross variables Y, the situation is more complicated than that for local variables. Some parents of Y are not in the same site so the parent set can be split into two sets: $pa_{local}(Y)$ is the set of parents in the same site and $pa_{other}(Y)$ is the set of parents in other sites. Since we cannot observe any variables in $pa_{other}(Y)$, the edges from a variable in $pa_{other}(Y)$ to Y (cross links) will be missing during local learning. The following proposition shows that the edges from variables in $pa_{local}(Y)$ to Y (local links) can be correctly detected during local learning.

PROPOSITION 3.2. (CROSS NODE) *For a cross variable Y, a base structure learning algorithm with decomposability property can find all the edges from variables in $pa_{local}(Y)$ to Y.*

Proof: Structure learning is a global optimization problem. Fix a cross variable Y. For centralized learning, let $pa_{best}(Y)$ be the optimal solution that maximizes $score(Y, pa(Y), D)$. We then have

$$score(Y, pa(Y), D) > score(Y, ns(Y), D),$$

where $ns(Y) \in \{S(Y) \setminus pa_{best}(Y)\}$. This means that, in a step with fixed candidate parent set, when we add a variable in $pa_{best}(Y)$, the score will increase. This is again clear from the K2 algorithm. Since $score(Y, pa(Y), D) = score_{local}(Y, pa(Y), D_{local})$ and $pa_{local}(Y) \subseteq pa_{best}(Y)$, $score_{local}(Y, pa(Y), D_{local})$ will increase after we add a variable in $pa_{local}(Y)$. That is, the edges from variables in $pa_{local}(Y)$ to Y will be detected in local learning. This concludes the proof.

For a variable Z in $pa_{other}(Y)$, the edge from Z to Y will be missing, since we do not observe Z at the

local site. But if there exists a path $U \rightarrow Z \rightarrow Y$ from a variable U (in the same site as Y) to Z (in the other site) then to Y, we may get a wrong extra edge from U to Y in local structure learning, because variable Z is "hidden" (unobserved) at the local site.

3.2 Sample Selection and Cross Learning In this subsection, we discuss how to find the local structure of a cross node. We first discuss how to select a small subset of the samples to be transmitted to a central site.

Let $p_A(.)$ and $p_B(.)$ denote the estimated probability functions (or likelihoods) involving the local variables in site A and B, obtained from local learning. Since $p_A(x)$, $p_B(x)$ denote the probability or likelihood of obtaining observation x at sites A and B, we would call these probability functions the likelihood functions $l_A(x)$ and $l_B(x)$, for the local models obtained at sites A and B, respectively. The observations at each site are ranked based on how well they fit the local model, using the local likelihood functions. The observations at site A with large likelihood under $l_A(.)$ are evidence of "local relationships" between site A variables, whereas those with low likelihoods under $l_A(.)$ are possible evidence of "cross relationships" between variables across sites. Let $IndLow_A$ denote the set of keys associated with the observations with low likelihood under $l_A(.)$. In practice, this step can be implemented in different ways. For example, we can set a threshold ρ_A and if $l_A(x) \leq \rho_A$, then $Ind(x) \in IndLow_A$. The sites A and B transmit the set of keys $IndLow_A$, $IndLow_B$, respectively, to a central site, where the intersection $IndLow = Ind_A \cap Ind_B$ is computed. The observations corresponding to the set of keys in $IndLow$ are then obtained from each local site by the central site. We assume that there is a single "key" variable (e.g., time index, user id) at the local sites that can be used to merge the transmitted datasets. This dataset is called D_{coll}. The details of this selection strategy are in [CSK02]. Finally, a BN (B_{cross}) is learnt at the central site using the transmitted data.

In practice, the likelihood threshold ρ_A, ρ_B have to be chosen carefully. In our experiments, we first set a small threshold at each local site and obtain dataset D^1_{coll} at the central site. This is used in the cross learning step to get B^1_{cross}. The likelihood threshold is then increased to obtain a larger dataset D^2_{coll} (most of samples in D^2_{coll} and D^1_{coll} are the same, so we only need to transmit a small number of samples to the central site again). Then we get B^2_{cross} from the cross learning step. After comparing B^2_{cross} and B^1_{cross} we retain only the common cross links. This procedure can be considered as a noise removal process.

3.3 Combination The last step of our collective learning is to combine the local BNs and B_{cross} learnt from cross learning into B_{coll}. From the discussions in Sections 3.1 and 3.2, we know the following: (a) Local learning can find the local structure of local variables and the local links of cross variables. (b) Cross learning can find the cross links of cross variables. (c) If we only assemble all local BNs together and do not use any information from cross learning, there will be two types of errors: extra local link due to the hidden variable problem and missing cross links for cross variables. In the combining step, we focus on the last problem. That is, how to add the cross links and remove the extra local links.

First, we assemble all local BNs together and get a BN B^{all}_{local}. We then add all the cross links from B_{cross} to B^{all}_{local}. The third step is to remove the extra local links. Note that the child variable of an extra local link must be a cross variable. In general, we do not know which variable is a cross variable. But in cross learning, we can get the cross links and the child variables of these links are cross variables. In this way, we can identify all cross variables. Then for each cross variable Y, find whether there exists a path from a variable Z, which in the same site with Y, to a variable in another site, and then back to Y. If there does exist this kind of (hidden variable) structure, we check the cross learning result for this variable. An extra local link appearing due to the hidden variable problem will not be supported by the result of cross learning, which is then removed.

4 Experimental Results

We illustrate our method using a real world BN application called the ALARM network. Due to space constraint, we are unable to present results on other common BNs. ALARM network is widely used as a benchmark model to evaluate BN learning algorithms. ALARM network has been developed for on-line monitoring of patients in intensive care units and generously contributed to the community by Beinlich and his collaborators [BSCC89]. The structure of the ALARM network is shown in Figure 1. It has 37 nodes and 46 edges. These nodes are discrete valued, but not necessarily binary. In our distributed ALARM model, The 37 nodes were split into 3 sites as shown in Figure 1. There are five cross variables: 17, 20, 21, 23, 30 and six cross links: $2 \rightarrow 17, 19 \rightarrow 20, 10 \rightarrow 21, 11 \rightarrow 21, 22 \rightarrow 23, 28 \rightarrow 30$ in this distributed ALARM model. A dataset with 15000 samples was generated from this ALARM network model

In our experiment, local learning detected all local links for cross variables and the structure of local variables. But in site A, an extra local link $11 \rightarrow 23$

was detected because of the path $11 \rightarrow 21 \rightarrow 22 \rightarrow 23$. In the cross learning step, we obtained all six cross links correctly and obtained no extra cross links after about 10% of all samples were transmitted to the central site. In the combination step, we checked the link $11 \rightarrow 23$ because node 23 is a cross variables and there exists a path $11 \rightarrow 21 \rightarrow 22 \rightarrow 23$. We found that the cross learning does not support this link so it was correctly removed. This experiment shows that our collective learning algorithm can learn the correct structure with a small amount of data transmission, for complex distributed BN model and large dataset.

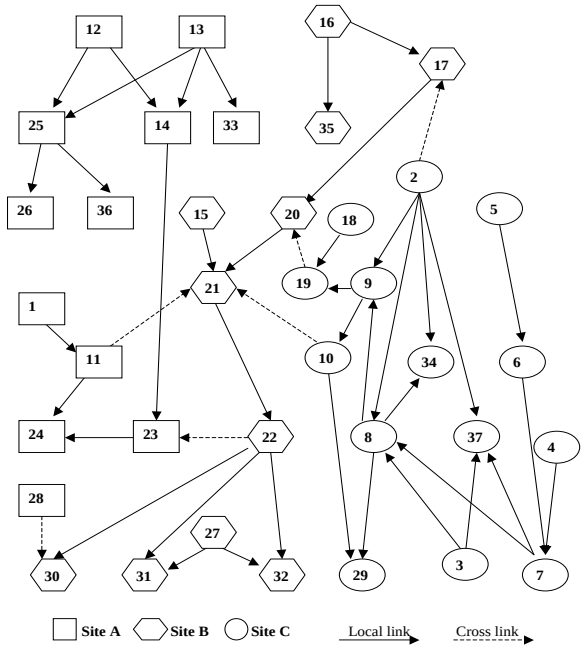

Figure 1: ALARM Network

5 Conclusion and Discussion

We have presented an approach to learn the structure of BN from distributed heterogenous data. This is based on a collective learning strategy, where a local model is obtained at each site and the global associations are determined by a selective transmission of data to a central site. In our collective method, local learning can identify the local structure of local variables and local links of cross variables. Cross learning can detect the cross links of cross variables. Combining the results, we can put together the local BNs and the BN learnt from cross learning and also remove any "extra" local links. In our experiments, the collective method learnt the same BN structure as that obtained by a centralized approach, even when only a small fraction of the data was transmitted. To our knowledge, this is the first approach to learn the structure of a BN from distributed heterogenous data.

Acknowledgements: The authors acknowledge supports from the United States National Science Foundation CAREER award IIS-0093353 and NASA (NRA) NAS2-37143.

References

[BSCC89] 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 Medical Care*, pages 247–256. Springer-Verlag, 1989.

[CH92] G. F. Cooper and E. Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9:309–347, 1992.

[CS02] R. Chen and K. Sivakumar. A new algorithm for learning parameters of a Bayesian network from distributed data. In *Proceedings of the 2002 IEEE International Conference on Data Mining*, pages 585–588, 2002.

[CSK01a] R. Chen, K. Sivakumar, and H. Kargupta. An approach to online Bayesian learning from multiple data streams. In *Proceedings of Workshop on Mobile and Distributed Data Mining, PKDD '01*, pages 31–45, 2001.

[CSK01b] R. Chen, K. Sivakumar, and H. Kargupta. Distributed web mining using Bayesian networks from multiple data streams. In *Proceedings of the IEEE Conference on Data Mining*, pages 75–82, 2001.

[CSK02] R. Chen, K. Sivakumar, and H. Kargupta. Collective mining of bayesian networks from distributed heterogeneous data. *Knowledge and Inofrmation Systems*, (Accepted) 2002.

[Hec98] D. Heckerman. A tutorial on learning with Bayesian networks. In M. I. Jordan, editor, *Proceedings of the NATO Advanced Study Institute on Learning in Graphical Models*, pages 301–354. Kluwer Academic Publishers, 1998.

[Ken97] Y. Kenji. Distributed cooperative Bayesian learning strategies. In *Proceedings of the Tenth Annual Conference on Computational Learning Theory*, pages 250–262, Nashville, Tennessee, 1997. ACM Press.

[KPHJ00] H. Kargupta, B. Park, D. Hershberger, and E. Johnson. Collective data mining: A new perspective toward distributed data mining. In H. Kargupta and P. Chan, editors, *Advances in Distributed and Parallel Knowledge Discovery*, pages 133–184. AAAI/ MIT Press, Menlo Park, California, USA, 2000.

[PV91] J. Pearl and T. Verma. A theory of inferred causation. In *KR'91*, pages 441–452, 1991.

[Suz93] J. Suzuki. A construction of Bayesian networks from databases based on an MDL scheme. In D. Heckerman and A. Mamdani, editors, *Proceedings of the Ninth Conference on Uncertainty in Artificial Intelligence*, pages 266–273. Morgam Kaufmann, 1993.