An approach to online Bayesian learning from multiple data streams

R. Chen[1], K. Sivakumar[1], and H. Kargupta[2]

[1] School of EECS, Washington State University
Pullman, WA 99164-2752, USA
rchen@eecs.wsu.edu, siva@wsu.edu
[2] Department of CSEE, University of Maryland Baltimore County,
Baltimore, MD 21250, USA
hillol@csee.umbc.edu

Abstract. We present a collective approach to mine Bayesian networks from distributed heterogenous web-log data streams. In this approach we first learn a local Bayesian network at each site using the local data. Then each site identifies the observations that are most likely to be evidence of coupling between local and non-local variables and transmits a subset of these observations to a central site. Another Bayesian network is learnt at the central site using the data transmitted from the local site. The local and central Bayesian networks are combined to obtain a collective Bayesian network, that models the entire data. This technique is then suitably adapted to an online Bayesian learning technique, where the network parameters are updated sequentially based on new data from multiple streams. We applied this technique to mine multiple data streams where data centralization is difficult because of large response time and scalability issues. This approach is particularly suitable for mining applications with distributed sources of data streams in an environment with non-zero communication cost (e.g. wireless networks). Experimental results and theoretical justification that demonstrate the feasibility of our approach are presented.

1 Introduction

The World Wide Web (WWW) is growing at an astounding rate. In order to optimize tasks such as Web site design, Web server design, and to simplify navigation through a Web site, the analysis of how the Web is used is very important. Usage information can be used to optimize web site design. For example, if we find that 80% of users who buy a computer model A in a web shop also visit links to a specific peripheral device B, or software package C, we can set up appropriate dynamic links for such users. Another example is for web server design. The different resources like html, jpeg, midi etc. are typically distributed among a set of servers. If we find a significant fraction of users requesting resources from server A also request some resource from server B, we can either keep a copy of those resources in server A or re-distribute the resources among the servers in a different fashion to minimize the communication between servers. Web server log contains records of user interactions when request for the resources in the servers is received. This contains a wealth of data for the analysis of web usage and identifying different patterns.

In an increasingly mobile world, the log data for an individual user would be distributed among many servers. For example, consider a subscriber of a wireless network who travels frequently and uses her PDA and cell phone to do business and personal transactions; her transactions go through different servers depending upon her location during the transaction. The PDA has very limited memory and communication ability, and her wireless service provider could offer more personalized service by paying careful attention to her needs and tastes. This may be useful for choosing the instant messages appropriate for her interests and physical location. For example, if she is visiting the Baltimore area the company may choose to send her instant messages regarding the area Sushi and Italian restaurants that she usually prefers, or local concert information. Since too many of such instant messages are likely to be considered a nuisance, accurate personalization is very important.

In this scenario, the web log files of the user are distributed in different sites of the service provider. Since these log files are very large, it's not feasible to transmit them to a central site for analysis. Moreover, these transaction data are heterogeneous. There is no guarantee that the user will perform the same type of transactions at every location. The user may choose to perform a wide variety of transactions at different sites (e.g. ordering pizza, purchasing gifts, bank transactions, monitoring personal financial portfolio, checking local weather). Therefore the features defining the transactions observed at different sites are likely to be different in general although we may have some overlap (e.g. monitoring personal financial portfolio, weather). Traditional data mining approach to this problem is aggregating all the log files to a central site before analysis. This would involve substantial data communication, large response time, and this approach does not scale well. A collective learning approach, that builds an overall model for the data based on local models, is a more logical approach.

A further challenge to collective learning is posed when the dataset is an online stream of observations. In this case, at each time point, we want to learn a model based on the observations upto that time point. In order to reduce computational complexity, this has to be done in an incremental fashion. In other words, at time $k+1$, we need to update the model obtained at time k, using the new set of observations available at time $k+1$. We refer to this incremental updating process as online learning, as opposed to a batch mode learning approach, where we store all the past data and at each time k, we learn a model afresh using all the data obtained upto that time point.

In this paper, we consider a Bayesian network (BN) to model the user log data, which is distributed over different sites. Specifically, we address the problem of learning a BN from heterogenous distributed data. A collective data mining (CDM) approach introduced earlier by Kargupta et. al. [11, 13] is used to learn a BN from distributed data. Section 2 provides some background and reviews

Table 1. Homogeneous case: Site A with a table for credit card transaction records.

Account No.	Amount	Location	Previous	Unusual transaction
11992346	-42.84	Seattle	Poor	Yes
12993339	2613.33	Seattle	Good	No
45633341	432.42	Portland	Okay	No
55564999	128.32	Spokane	Okay	Yes

Table 2. Homogeneous case: Site B with a table for credit card transaction records.

Account No.	Amount	Location	Previous	Unusual transaction
87992364	446.32	Berkeley	Good	No
67845921	978.24	Orinda	Good	Yes
85621341	719.42	Walnut	Okay	No
95345998	-256.40	Francisco	Bad	Yes

existing literature in related area. Section 3 presents our approach to distributed web log mining using Bayesian networks. An approach to learn a global Bayesian network from distributed data, with selective data transmission is presented. We then extend this approach to online Bayesian learning for multiple data streams. This is particularly relevant to real-time and time-sensitive applications like stock-market data or web-log data. Experimental results are presented in Section 4. Finally, we provide some discussions and concluding remarks in Section 5.

2 Background and related work

In this section, we first illustrate the difference between homogenous and heterogenous databases. We then review important literature related to Bayesian networks (BN) and web mining and provide a brief review of BNs.

Distributed data mining (DDM) must deal with different possibilities of data distribution. Different sites may contain data for a common set of features of the problem domain. In case of relational data this would mean a consistent database schema across all the sites. This is the homogeneous case. Tables 1 and 2 illustrate this case using an example from a hypothetical credit card transaction domain.[1] There are two data sites A and B, connected by a network. The DDM-objective in such a domain may be to find patterns of fraudulent transactions. Note that both the tables have the same schema.

In the general case the data sites may be *heterogeneous.* In other words, sites may contain tables with different schemata. Different features are observed at different sites. Let us illustrate this case with relational data. Table 3 shows two data-tables at site X. The upper table contains weather-related data and the lower one contains demographic data. Table 4 shows the content of site Y, which contains holiday toy sales data. The objective of the DDM process may be

[1] Please note that the credit card domain may not always have consistent schema. The domain is used just for illustration.

Table 3. Heterogeneous case: Site X with two tables, one for weather and the other for demography.

City	Temp.	Humidity	Wind Chill
Boise	20	24%	10
Spokane	32	48%	12
Seattle	63	88%	4
Portland	51	86%	4
Vancouver	47	52%	6

City	State	Size	Average earning	Proportion of small businesses
Boise	ID	Small	Low	0.041
Spokane	WA	Medium	Medium	0.022
Seattle	WA	Large	High	0.014
Portland	OR	Large	High	0.017
Vancouver	BC	Medium	Medium	0.031

Table 4. Heterogeneous case: Site Y with one table holiday toy sales.

State	Best Selling Item	Price ($)	Number Items Sold
WA	Snarc Action Figure	47.99	23K
ID	Power Toads	23.50	2K
BC	Light Saber	19.99	5K
OR	Super Squirter	24.99	142K
CA	Super Fun Ball	9.99	24K

detecting relations between the toy sales, the demographic and weather related features. In the general heterogeneous case the tables may be related through different sets of key indices. For example, Tables 3(upper) and (lower) are related through the key feature *City*; on the other hand Table 3 (lower) and Table 4 are related through key feature *State*. Note that the key is just an index and is usually not a variable of interest in the knowledge discovery process. The role of the key index is only to link observations across different sites. For example, in a time series data, the time index can be used to link observations of variables made at different sites. For a web log mining application, this key could be produced using either a “cookie” or the user IP address (in combination with other log data like time of access). In this paper, we consider the heterogenous data scenario described above.

2.1 Related Work

A BN is a probabilistic graphical model that represents uncertain knowledge [19, 12, 3].

The problem of online Bayesian learning for centralized data has been addressed in the literature. In [21, 2, 7], the authors give methods to update the parameters of a Bayesian network when the network structure is known. In general, this updating process is based on the maximum a posteriori (MAP) framework. In [15], a MAP method for updating the Bayesian network structure is proposed. Friedman [8] provides an incremental network structure updating method for online Bayesian learning.

An important problem is how to learn the Bayesian network from data in distributed sites. The centralized solution to this problem is to download all datasets from distributed sites. Kenji [14] worked on the homogeneous distributed learning scenario. In this case, every distributed site has the same feature but different observations. In this paper, we address the heterogenous case, where each site has data about only a subset of the features. To our knowledge, there is no significant work that addresses the heterogenous case.

We now review some important work reported on mining useful pattern from web logs. The concept of applying data mining algorithm to web log was proposed in [4, 17]. Chen et. al. [4] introduce the concept of maximal forward reference, whereas Mannila et. al. [17] propose discovering frequent episodes from web log. In [1] the sequential pattern mining technique is used to discover user patterns. The application of web log mining in improving web sites design, system performance analysis, and building dynamic links is reported in [6, 20]. Cooley et. al. [5] address the preprocessing technique for web log mining. Some work on using probability model in web mining has also been reported. In [9] the authors use probabilistic relational models to optimize web site design, whereas Pazzani [18] uses a naive Bayesian classifier to learn user preference.

2.2 A brief review of Bayesian networks

A Bayesian network (BN) is a probabilistic graph model. It can be defined as a pair $(\mathcal{G}, p)$, where $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a directed acyclic graph (DAG). Here, $\mathcal{V}$ is the node set which represents variables in the problem domain and $\mathcal{E}$ is the edge set which denotes probabilistic relationships among the variables. For a variable $X \in \mathcal{V}$, a parent of X is a node from which there exists a directed link to X. Figure 1 is a BN called the ASIA model (adapted from [16]). The variables are Dyspnoea, Tuberculosis, Lung cancer, Bronchitis, Asia, X-ray, Either, and Smoking. They are all binary variables.

Let $pa(X)$ denote the set of parents of X, then the conditional independence property can be represented as follows:

$$P(X \mid \mathcal{V} \setminus X) = P(X \mid pa(X)). \tag{1}$$

This property can simplify the computations in a Bayesian network model. For example, the joint distribution of the set of all variables in $\mathcal{V}$ can be written as a product of conditional probabilities as follows:

$$P(\mathcal{V}) = \prod_{X \in \mathcal{V}} P(X \mid pa(X)). \tag{2}$$

The set of conditional distributions $\{P(X \mid pa(X)), X \in \mathcal{V}\}$ are called the parameters of a Bayesian network. If variable X has no parents, then $P(X \mid pa(X)) = P(X)$ is the marginal distribution of X. The ordering of variables constitutes a constraint on the structure of a Bayesian network. If variable X appears before variable Y, then Y can not be a parent of X.

Two important issues in using a Bayesian network are : (a) learning a Bayesian network and (b) probabilistic inference. Learning a BN involves learning the structure of the network (the directed graph), and obtaining the conditional

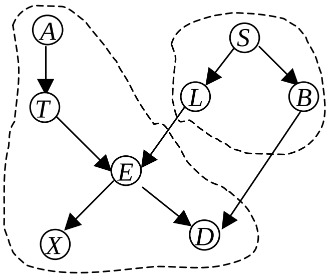

Fig. 1. ASIA Model

probabilities (parameters) associated with the network. Once a Bayesian network is constructed, we usually need to determine various probabilities of interest from the model. This process is referred to as probabilistic inference.

Bayesian network is an important tool to model probabilistic or imperfect relationship among problem variables. It gives useful information about the mutual dependencies among the features in the application domain. Such information can be used for gaining better understanding about the dynamics of the process under observation. It is thus a promising tool to model customer usage patterns in web data mining applications, where specific user preferences can be modeled as in terms of conditional probabilities associated with the different features.

3 Collective Bayesian learning

In the following, we discuss our collective approach to learning a Bayesian network that is specifically designed for a distributed data scenario.

The primary steps in our approach are:

(a) Learn local BNs (local model) involving the variables observed at each site based on local data set.

(b) At each site, based on the local BN, identify the observations that are most likely to be evidence of coupling between local and non-local variables. Transmit a subset of these observations to a central site.

(c) At the central site, a limited number of observations of all the variables are now available. Using this to learn a non-local BN consisting of links between variables across two or more sites.

(d)Combine the local models with the links discovered at the central site to obtain a collective BN.

The non-local BN thus constructed would be effective in identifying associations between variables across sites, whereas the local BNs would detect associations among local variables at each site. The conditional probabilities can also be estimated in a similar manner. Those probabilities that involve only variables from a single site can be estimated locally, whereas the ones that involve variables from different sites can be estimated at the central site. Same methodology could be used to update the network based on new data. First, the new data is tested for how well it fits with the local model. If there is an acceptable statistical fit, the observation is used to update the local conditional probability estimates. Otherwise, it is also transmitted to the central site to update the appropriate conditional probabilities (of cross terms). Finally, a collective BN can be obtained by taking the union of nodes and edges of the local BNs and the nonlocal BN and using the conditional probabilities from the appropriate BNs. Probabilistic inference can now be performed based on this collective BN. Note that transmitting the local BNs to the central site would involve a significantly lower communication as compared to transmitting the local data.

It is quite evident that learning probabilistic relationships between variables that belong to a single local site is straightforward and does not pose any additional difficulty as compared to a centralized approach.[2] The important objective is to correctly identify the coupling between variables that belong to two (or more) sites. These correspond to the edges in the graph that connect variables between two sites and the conditional probability(ies) at the associated node(s). In the following, we describe our approach to selecting observations at the local sites that are most likely to be evidence of strong coupling between variables at two different sites. The key idea of our approach is that the samples that do not fit well with the local models are likely to be evidence of coupling between local and non-local variables. We transmit these samples to a central site and use them to learn a collective Bayesian network.

3.1 Selection of samples for transmission to global site

For simplicity, we will assume that the data is distributed between two sites and will illustrate the approach using the BN in Figure 1. The extension of this approach to more than two sites is straightforward. Let us denote by $\mathcal{A}$ and $\mathcal{B}$, the variables in the left and right groups, respectively, in Figure 1. We assume that the observations for $\mathcal{A}$ are available at site A, whereas the observations for $\mathcal{B}$ are available at a different site B. Furthermore, we assume that there is a common feature ("key" or index) that can be used to associate a given observation in site A to a corresponding observation in site B. Naturally, $\mathcal{V} = \mathcal{A} \cup \mathcal{B}$.

At each local site, a local Bayesian network can be learned using only samples in this site. This would give a BN structure involving only the local variables at each site and the associated conditional probabilities. Let $p_A(.)$ and $p_B(.)$ denote the estimated probability function involving the local variables. This is the product of the conditional probabilities as indicated by (2). 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 $l_A(.)$ and $l_B(.)$, for the local model obtained at sites A and B, respectively. The observations at each site are ranked based on how well it fits 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

[2] This may not be true for arbitrary Bayesian network structure. We will discuss this issue further in the last section.

low likelihoods under $l_A(.)$ are possible evidence of "cross relationships" between variables across sites. Let $S(A)$ denote the set of keys associated with the latter observations (those 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 $x \in S_A$. The sites A and B transmit the set of keys S_A, S_B, respectively, to a central site, where the intersection $S = S_A \cap S_B$ is computed. The observations corresponding to the set of keys in S are then obtained from each of the local sites by the central site.

The following argument justifies our selection strategy. Using the rules of probability, and the assumed conditional independence in the BN of Figure 1, it is easy to show that:

$$P(\mathcal{V}) = P(\mathcal{A}, \mathcal{B}) = P(\mathcal{A} \mid \mathcal{B})P(\mathcal{B}) = P(\mathcal{A} \mid nb(\mathcal{A}))P(\mathcal{B}), \tag{3}$$

where $nb(\mathcal{A}) = \{B, L\}$ is the set of variables in $\mathcal{B}$, which have a link connecting it to a variable in $\mathcal{A}$. In particular,

$$P(\mathcal{A} \mid nb(\mathcal{A})) = P(A)P(T \mid A)P(X \mid E)P(E \mid T, L)P(D \mid E, B). \tag{4}$$

Note that, the first three terms in the right-hand side of (4) involve variables local to site A, whereas the last two terms are the so-called *cross terms*, involving variables from sites A and B. Similarly, it can be shown that

$$P(\mathcal{V}) = P(\mathcal{A}, \mathcal{B}) = P(\mathcal{B} \mid \mathcal{A})P(\mathcal{A}) = P(\mathcal{B} \mid nb(\mathcal{B}))P(\mathcal{A}), \tag{5}$$

where $nb(\mathcal{B}) = \{E, D\}$ and

$$P(\mathcal{B} \mid nb(\mathcal{B})) = P(S)P(B \mid S)P(L \mid S)P(E \mid T, L)P(D \mid E, B). \tag{6}$$

Therefore, an observation $\{A = a, T = t, E = e, X = x, D = d, S = s, L = l, B = b\}$ with low likelihood at both sites A and B; i.e. for which both $P(\mathcal{A}) = P(A = a, T = t, E = e, X = x, D = d)$ and $P(\mathcal{B}) = P(S = s, L = l, B = b)$ are small, is an indication that both $P(\mathcal{A} \mid nb(\mathcal{A}))$ and $P(\mathcal{B} \mid nb(\mathcal{B}))$ are large for that observation (since observations with small $P(\mathcal{V})$ are less likely to occur). Notice from (4) and (6) that the terms common to both $P(\mathcal{A} \mid nb(\mathcal{A}))$ and $P(\mathcal{B} \mid nb(\mathcal{B}))$ are precisely the conditional probabilities that involve variables from both sites A and B. In other words, this is an observation that indicates a coupling of variables between sites A and B and should hence be transmitted to a central site to identify the specific coupling links and the associated conditional probabilities.

In a sense, our approach to learning the cross terms in the BN involves a selective sampling of the given dataset that is most relevant to the identification of coupling between the sites. This is a type of *importance sampling*, where we select the observations that have high conditional probabilities corresponding to the terms involving variables from both sites. Naturally, when the values of the different variables (features) from the different sites, corresponding to these selected observations are pooled together at the central site, we can learn the coupling links as well as estimate the associated conditional distributions. These selected observations will, by design, not be useful to identify the links in the BN that are local to the individual sites. This has been verified in our experiments (see Section 4).

3.2 Online Distributed Bayesian Network Learning

The proposed collective approach to learning a BN is well suited for a scenario with multiple data streams. Suppose we have an existing BN model, which has to be constantly updated based on new data from multiple streams. For simplicity, we will consider only the problem of updating the BN parameters, assuming that the network structure is known. As in the case of batch mode learning, we shall use techniques for online updating of BN parameters for centralized data. In the centralized case, there exists simple techniques for parameter updating for commonly used models like the unrestricted multinomial model. For example, let us denote by $p_{ijl} = \Pr(x_i = l \mid pa_{x_i} = j)$, the conditional probability at node i, given the parents of node i. We can then obtain the estimate $p_{ijl}(k+1)$ of p_{ijl} at step $k+1$ as follows (see [10, Section 5]):

$$p_{ijl}(k+1) = \frac{\alpha_{ijl}(k) + N_{ijl}(k+1)}{\alpha_{ij}(k) + N_{ij}(k+1)}, \tag{7}$$

where $\alpha_{ij}(k) = \sum_l \alpha_{ijl}(k)$ and $N_{ij}(k+1) = \sum_l N_{ijl}(k+1)$. In eq. (7), $N_{ijl}(k+1)$ denotes the number of observations in the dataset obtained at time $k+1$ for which, $x_i = l$ and $pa_{x_i} = j$, and we can set $\alpha_{ijl}(k+1) = \alpha_{ijl}(k) + N_{ijl}(k+1)$. Note that $N_{ijl}(k)$ are a set of sufficient statistics for the data observed at time k.

For online distributed case, parameters for local terms can be updated using the same technique as in a centralized case. Next, we need to update the parameters for the cross-links, without transmitting all the data to a central site. Again we choose the samples with low likelihood in local sites and transmit them to a central site. This is then used to update the cross-terms at the central site. We can summarize our approach by the following steps:

1. Learn an initial collective Bayesian network from the first dataset observed (unless a prior model is already given). Thus we have a local BN at each site and a set of cross-terms at the central site.
2. At each step k:
   - Update the local BN parameters at each site using eq. (7).
   - Update the likelihood threshold at each local site, based on the sample mean value of the observed likelihoods. This is the threshold used to determine if a sample is to be transmitted to a central site (see Section 3.1).
   - Transmit the low likelihood samples to a central site.
   - Update the parameters of the cross-terms at the central site.
   - Combine the updated local terms and cross terms to get an updated collective Bayesian network.
3. Increment k and repeat step (2) for the next set of data.

4 Experimental Results

We tested our approach on two different datasets. A small real web log dataset was used for batch mode distributed Bayesian learning. This was used to test both structure and parameter learning. We also tested our online distributed learning approach on a simulated web log dataset. More extensive examples for batch mode learning, demonstrating scalability with respect to number of distributed sites will be presented elsewhere. We present our results for the three cases in the following two subsections.

4.1 Webserver Log Data

In this experiment, we used data from real world domain — a web server log data. This experiment illustrates the ability of the proposed collective learning approach to learn the parameters of a BN from real world web log data. Web server log contains records of user interactions when request for the resources in the servers is received. Web log mining can provide useful information about different user profiles. This in turn can be used to offer personalized services as well as to better design and organize the web resources based on usage history.

In our application, the raw web log file was obtained from the web server of the School of EECS at Washington State University — `http://www.eecs.wsu.edu`. There are three steps in our processing. First we preprocess the raw web log file to transform it to a session form which is useful to our application. This involves identifying a sequence of logs as a single session, based on the IP address (or cookies if available) and time of access. Each session corresponds to the logs from a single user in a single web session. We consider each session as a data sample. Then we categorize the resource (html, video, audio etc.) requested from the server into different categories. For our example, based on the different resources on the EECS web server, we considered eight categories: E-EE Faculty, C-CS Faculty, L-Lab and facilities, T-Contact Information, A-Admission Information, U-Course Information, H-EECS Home, and R-Research. These categories are our features. In general, we would have several tens (or perhaps a couple of hundred) of categories, depending on the webserver. This categorization has to be done carefully, and would have to be automated for a large web server. Finally, each feature value in a session is set to one or zero, depending on whether the user requested resources corresponding to that category. An 8-feature, binary dataset was thus obtained, which was used to learn a BN.

A central BN was first obtained using the whole dataset. Figure 2 depicts the structure of this centralized BN. We then split the features into two sets, corresponding to a scenario where the resources are split into two different web servers. Site A has features E, C, T, and U and site B has features L, A, H, and R. We assumed that the BN structure was known, and estimated the parameters (probability distribution) of the BN using our collective BN learning approach. Figure 3 shows the KL distance between the central BN and the collective BN as a function of the fraction of observations communicated. Clearly the parameters

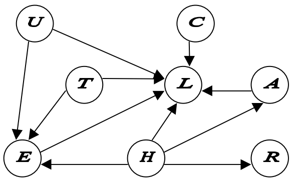

Fig. 2. Bayesian Network Structure learnt from Web Log Data

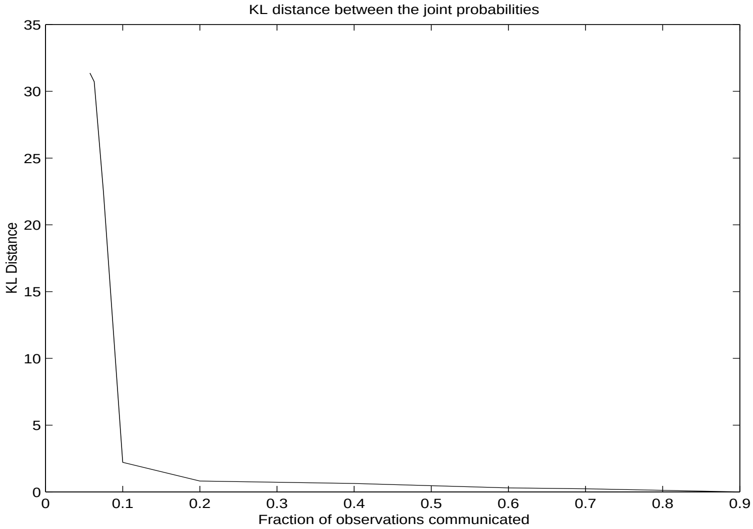

Fig. 3. KL distance between joint probabilities

of collective BN is close to that of central BN even with a small fraction of data communication.

4.2 Online distributed learning

We now illustrate the results of online BN parameter learning, assuming the network structure is known. We use the model shown in Figure 4. The 32 nodes in the network are distributed among four different sites. Nodes 1, 5, 10, 15, 16, 22, 23, 24, 30, and 31 are in site A. Nodes 2, 6, 7, 11, 17, 18, 25, 26, and 32 are in site B. Nodes 3, 8, 12, 19, 20, and 27 are in site C. Nodes 4, 9, 13, 14, 21, 28, and 29 are in site D. A dataset with 80000 observations was generated. We assumed that at each step k, 5000 observations of the data are available (for a total of 16 steps).

We denote by B_{be}, the Bayesian network obtained by using all the 80,000 samples in batch mode (the data is still distributed into four sites). We denote by $B_{ol}(k)$, the Bayesian network obtained at step k using our online learning approach and by $B_{ba}(k)$, the Bayesian network obtained using a reular batch

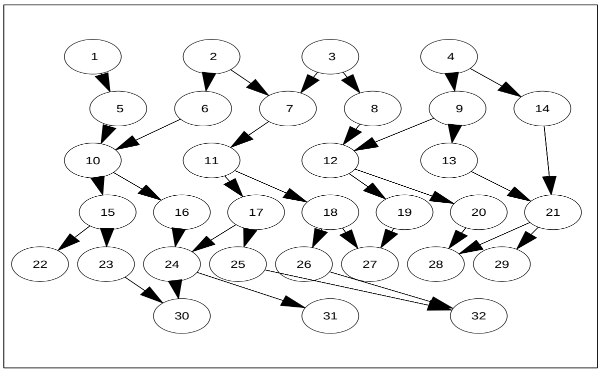

Fig. 4. Bayesian network for online distributed parameter learning

mode learning, but using only data observed upto time k. We choose three typical cross terms (nodes 12, 27, and 28) and compute the KL distance between the conditional probabilities to evaluate the preformance of online distributed method. The results are depicted in Figure 5.

Figure 5 (left) shows the KL distance between the conditional probabilities for the networks $B_{ol}(k)$ and B_{be} for the three nodes. We can see that the perfomance of online distributed method is good, with the error (in terms of KL distance) dropping rapidly. Figure 5 (right) shows the KL distance between the conditional probabilities for the networks $B_{ba}(k)$ and B_{ol} for the three nodes. We can see that the perfomance of a network learned using our online distributed method is comparable to that learned using a batch mode method, with the same data.

5 Discussions and Conclusions

We have presented an approach to learning Bayesian networks 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 experiments, the performance of the collective Bayesian network was quite comparable to that obtained from a centralized approach, even for a small data communication. To our knowledge, this is the first approach to learning Bayesian networks from distributed heterogenous data.

Bayesian networks are used to model probabilistic relationships among features and are well suited for modeling user patterns and preferences in web mining applications. Moreover, our collective learning approach lends itself to easy update of the model based on new observations, particularly when these observations are also distributed. This is ideally suited for a online learning applications with multiple data streams. Our experiments (some presented elsewhere) suggest that the collective learning scales well with respect to number of sites, samples, and features.

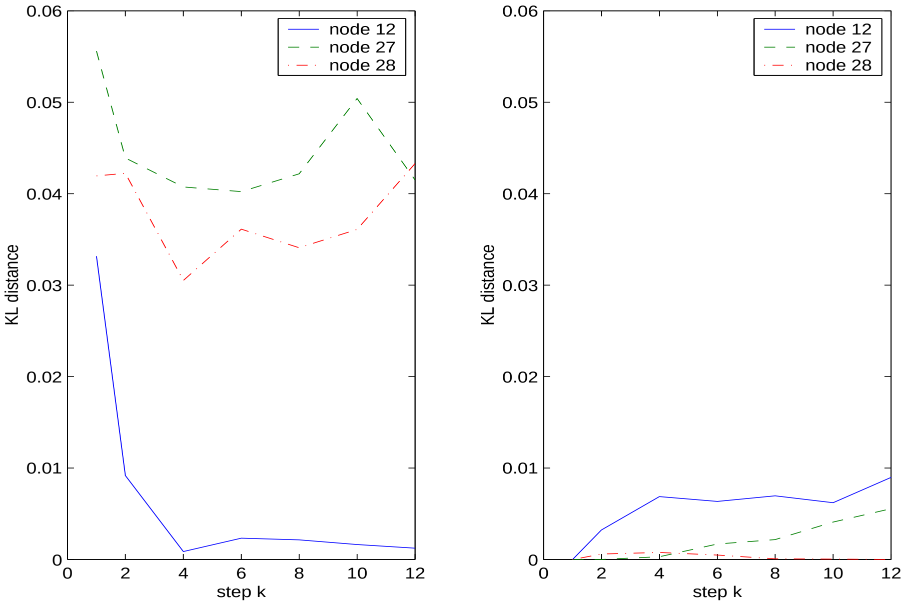

Fig. 5. Simulation results for online Bayesian learning: (left) KL distance between the conditional probabilities for the networks $B_{ol}(k)$ and B_{be} for three nodes (right) KL distance between the conditional probabilities for the networks $B_{ol}(k)$ and B_{ba} for three nodes

Many interesting applications are possible from a BN model of the web log data. For example, specific structures in the overall BN would indicate special user patterns. This could be used to identify new user patterns and accordingly personalize offers and services provided to such users. Another interesting application is to classify the users into different groups based on their usage patterns. This can be thought of decomposing the overall BN (obtained from the log data by collective learning) into a number of sub-BNs, each sub-BN representing a specific group having similar preferences. We are actively pursuing these ideas and would report results in a future publication.

We now discuss some limitations of our proposed approach, which suggest possible directions for future work.

– **Hidden node at local sites**: For certain network structures, it may not be possible to obtain the correct (local) links, based on local data at that site. For example, consider the ASIA model shown in Figure 1, where the observations corresponding to variables A, T, E, and X are available at site A and those corresponding to variables S, L, B, and D are available at site B. In this case, when we learn a local BN at site B, we would expect a (false) edge from node L to node D, because of the edges $L \rightarrow E$ and $E \rightarrow D$ in the overall BN and the fact that node E is "hidden" (unobserved) at site B. This was verified experimentally as well. However, the cross-links

$L \rightarrow E$ and $E \rightarrow D$ were still detected correctly at the central site, using our "selctively sampled" data. Therefore, it is necessary to re-examine the local links after discovering the cross-links. In other words, some post-processing of the resulting overall BN is required to eliminate such false local edges. This can be done by evaluating an appropriate score metric on BN configurations with and without such suspect local links. We are currently pursuing this issue. Note, however, that we do not encounter this problem in the examples presented in Section 4.
- **Assumptions about the data**: As mentioned earlier, we assume the existence of a key that links observations across sites. Moreover, we consider a simple heterogenous partition of data, where the variable set at different sites are non-overlapping. We also assume that our data is stationary (all data points come from the same distribution) and free of outliers. These are simplifying assumptions to derive a reasonable algorithm for distributed Bayesian learning. Suitable learning strategies that would allow us to relax of some of these assumptions would be an important area of research.
- **Structure Learning**: Even when the data is centralized, learning the structure of BN is considerably more involved than estimating the parameters or probabilities associated with the network. In a distributed data scenarion, the problem of obtaining the correct network structure is even more pronounced. The "hidden node" problem discussed earlier is one example of this. As in the centralized case, prior domain knowledge at each local site, in the form of probabilistic independence or direct causation, would be very helpful. Our experiments on the ASIA model demonstrate that the proposed collective BN learning approach to obtain the network structure is reasonable, at least for simple cases. However, this is just a beginning and deserves careful investigation.
- **Performance Bounds**: Our approach to "selective sampling" of data that maybe evidence of cross-terms is reasonable based on the discussion in Section 3 (see eq. (3)-(6)). This was verified experimentally for the three examples in Section 4. Currently, we are working towards obtaining bounds for the performance of our collective BN as compared to that obtained from a centralized approach, as a function of the data communication involved.

Acknowledgements: This work was partially supported by NASA under Cooperative agreement NCC 2-1252.

References

1. R. Agrawal and R. Srikant, "Mining sequential pattern," in *Proceedings of the International Conference on Data Engineering*, (Taipei), pp. 3–14, 1995.
2. W. Buntine, "Theory refinement on Bayesian networks," in *Proceedings of the Seventh Annual Conference on Uncertainty in Artificial Intelligence* (B. D. D'Ambrosio and P. S. amd P. P. Bonissone, eds.), pp. 52–60, Morgan Kaufmann, 1991.

3. E. Charniak, “Bayesian networks without tears,” *AI Magazine*, vol. 12, pp. 50–63, 1991.
4. M. Chen, J. Park, and P. Yu, “Data mining for path traversal patterns in a web enviroment,” in *Proceedings of 16th International Conference on Distributed Computing Systems*, pp. 385–392, 1996.
5. R. Cooley, B. Mobasher, and J. Srivastava, “Data preparation for mining world wide web browsing patterns,” *Knowledge and Information Systems*, 1999.
6. J. G. Cumming, “Hits and miss-es: A year watching the web,” in *Proceedings of the Sixth International World Wide Web Conference*, (Santa Clara), 1997.
7. F. J. Diez, “Parameter adjustment in Bayes networks: the generalized noisy OR-date,” in *Proceedings of the Ninth Conference on Uncertainty in Artificial Intelligence*, pp. 99–105, 1993.
8. N. Friedman and M. Goldszmidt, “Sequential update of Bayesian network structure,” in *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence* (D. Geiger and P. Shanoy, eds.), Morgan Kaufmann, 1997.
9. L. Getoor and M. Sahami, “Using probabilistic relational models for collaborative filtering,” in *Proceedings of WEBKDD*, 1999.
10. D. Heckerman, “A tutorial on learning with Bayesian networks,” Tech. Rep. MSR-TR-95-06, Microsoft Research, 1995.
11. D. Hershberger and H. Kargupta, “Distributed multivariate regression using wavelet-based collective data mining,” Tech. Rep. EECS-99-02, School of EECS, Washington State University, 1999. To be published in the Special Issue on Parallel and Distributed Data Mining of the Journal of Parallel Distributed Computing, Guest Eds: Vipin Kumar, Sanjay Ranka, and Vineet Singh.
12. F. Jensen, *An Introduction to Bayesian Networks*. Springer, 1996.
13. E. Johnson and H. Kargupta, “Collective, hierarchical clustering from distributed, heterogeneous data,” in *Lecture Notes in Computer Science*, vol. 1759, pp. 221–244, Springer-Verlag, 1999.
14. Y. Kenji, “Distributed cooperative Bayesian learning strategies,” in *Proceedings of the Tenth Annual Conference on Computational Learning Theory*, (Nashville, Tennessee), pp. 250–262, ACM Press, 1997.
15. W. Lam and F. Bacchus, “Using new data to refine a Bayesian network,” in *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence* (R. L. de Mantaras and D. Poole, eds.), pp. 383–390, Morgan Kaufmann, 1994.
16. S. L. Lauritzen and D. J. Spiegelhalter, “Local computations with probabilities on graphical structures and their application to expert systems (with discussion),” *Journal of the Royal Statistical Society, series B*, vol. 50, pp. 157–224, 1988.
17. H. Mannila, H. Toivonen, and A. Verkamo, “Discovering frequent episodes in sequences,” in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, (Portland), pp. 210–215, 1996.
18. M. Pazzani, J. Muramatsu, and D. Billsus, “Syskill & Webert: Identifying interesting web sites,” in *Proceedings of the National Conference on Artificial Intelligence*, 1996.
19. J. Pearl, *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, 1988.
20. M. Perkowitz and O. Etzioni, “Adaptive sites: Automatically learning from user access patterns,” in *Proceedings of the Sixth International World Wide Web Conference*, (Santa Clara), 1997.
21. D. J. Spiegelhalter and S. L. Lauritzen, “Sequential updating of conditional probabilities on directed graphical structures,” *Networks*, vol. 20, pp. 570–605, 1990.