Behaviormetrika
Vol.35, No.2, 2008, 137–158

COLLABORATIVE FILTERING FOR MASSIVE DATASETS BASED ON BAYESIAN NETWORKS

Maomi Ueno* and Takahiro Yamazaki*

This paper proposes a collaborative filtering method for massive datasets that is based on Bayesian networks. We first compare the prediction accuracy of four scoring-based learning Bayesian networks algorithms (AIC, MDL, UPSM, and BDeu) and two conditional-independence-based (CI-based) learning Bayesian networks algorithms (MWST, and Polytree-MWST) using actual massive datasets. The results show that (1) for large networks, the scoring-based algorithms have lower prediction accuracy than the CI-based algorithms and (2) when the scoring-based algorithms use a greedy search to learn a large network, algorithms which make a lot of arcs tend to have less prediction accuracy than those that make fewer arcs. Next, we propose a learning algorithm based on MWST for collaborative filtering of massive datasets. The proposed algorithm employs a traditional data mining technique, the "a priori" algorithm, to quickly calculate the amount of mutual information, which is needed in MWST, from massive datasets. We compare the original MWST algorithm and the proposed algorithm on actual data, and the comparison shows the effectiveness of the proposed algorithm.

1. Introduction

Recommender systems automatically propose suitable items to individual users on the basis of various personal information. This has become an important research area since the appearance of the first papers on collaborative filtering in the mid-1990s (Hill et al., 1995; Resnick et al., 1994; Shardanand and Maes, 1995). Recommender systems are usually classified into the following categories according to how recommendations are made (Balabanovic and Shoham, 1997):

- Content-based recommendations: Items similar to those preferred by the user in the past are recommended.
- Collaborative recommendations: Items that people with similar tastes and preferences liked in the past are recommended.
- Hybrid approaches: These methods combine collaborative and content-based methods.

In content-based recommendations, an item profile is usually computed by extracting a set of features from the item, and the profile is used to determine the appropriateness of the item for recommendation purposes. The advantages of content-based recommendation are that we do not need peer user profiles and the latest items that have no peer user profiles are available for recommendation. The disadvantages are that only similar items tend to be recommended and it is sometimes difficult to extract features from an item.

Collaborative recommendations involve two general types of algorithm: memory-based

Key Words and Phrases: Bayesian networks, collaborative filtering, recommender system, preference, massive data.

* Graduate School of Information Systems, The University of Electro-Communications

Table 1: Comparing collaborative filtering methods with ranked scoring results for the MS Web dataset

Algorithm	Given2	Given5	Given10	AllBut1
BN	59.95	59.84	53.92	66.69
CR+	60.64	57.89	51.47	63.59
VSIM	59.22	56.13	49.33	61.70
BC	57.03	54.83	47.83	59.42
POP	49.14	46.91	41.14	49.77
RD	0.91	1.82	4.49	0.93

(or heuristic-based) and model-based. Memory-based algorithms (Breese et al., 1998; Delgado and Ishii, 1999; Nakamura and Abe, 1998; Resnick et al., 1994; Shardanand and Maes, 1995) essentially are heuristics that make rating predictions based on the entire collection of items previously rated by users. In contrast, model-based algorithms (Billsus and Pazzani, 1998; Breese et al., 1998; Getoor and Sahami, 1999; Goldberg et al., 2001; Hofmann, 2003; Marlin, 2003; Pavlov and Pennock, 2002; Ungar and Foster, 1998) use the collection of ratings to learn a model, which is then used to make rating predictions. The advantage of collaborative recommendations is that they do not need item profiles. This means collaborative recommendations do not inevitably recommend items that have similar profiles.

There are many kinds of collaborative recommendation algorithms, and the prediction accuracy of their recommendations depends on the collaborative recommendation algorithms. Breese et al. (1998) compare the prediction accuracies of various collaborative recommendation algorithms. Bayesian-network-based collaborative filtering has the best performance in Table 1, where "BN" indicates the Bayesian network model, "CR+" indicates the correlation method, "VSIM" indicates the vector similarity method, "BC" indicates the Bayesian clustering model, "POP" indicates the baseline performance, and "RD" indicates the required difference at the 90% confidence level for the experiment as a whole. "Given2", "Given5", and "Given10" in Table 1 respectively indicate the prediction accuracies given two, five, and ten observations. "Allbut1" in Table 1 indicates the prediction accuracy given the observations for all variables except the target variable. Although the results show that BN performs the best, Breese et al. (1998) use only the BDeu (likelihood-equivalence Bayesian Dirichlet with uniform prior) metric (Heckerman et al., 1995), which is one of various scoring metrics for learning Bayesian networks. It is known that the prediction accuracy of Bayesian network depends on the learning algorithm. For example, Yang and Chang (2002) compare the performances of various score metrics for learning Bayesian networks. The results of their simulation experiments show that the Dirichlet Prior Score Metric with a 10-th order hyper-parameter is the best and BDeu is the worst. We, therefore, expect that the other learning Bayesian networks algorithms perform better than BDeu for collaborative recommendation algorithms.

Learning Bayesian networks algorithms can be divided into two types (Cheng and Greiner, 1999): (1) scoring-based learning algorithms and (2) conditional-independence-based (CI-based) learning algorithms. The former uses a scoring metric such as AIC (Akaike, 1974), MDL (Rissanen, 1983), UPSM (Cooper and Herskovits, 1992), or BDeu

(Heckerman et al., 1995). In scoring-based learning, the search problem for the network structure is NP-complete. We therefore have to use heuristics to shrink the search space. The greedy search procedure using an ordering of the nodes is often used for that purpose. Even if we use this procedure, the computational cost for the search procedure increases exponentially with the number of nodes. The latter uses a conditional independence test (CI-test) such as the Chi-squared test or a mutual information test for finding conditionally independent nodes sets. The computational cost of the CI-based algorithms is $O(n^2)$ (n indicates the number of nodes). This indicates that the computational cost of these algorithms is much less than that of the scoring-based ones.

Heckerman et al. (1997) compare the two kinds of learning algorithm and show that the scoring-based algorithms often have certain advantages over the CI-based algorithms in terms of modeling a distribution. Their study does not use a large number of variables and evaluates only the accuracy of modeling a distribution. However, large numbers of variables are generally used in collaborative recommendations, and the accuracy of modeling a distribution does not reflect the prediction accuracy of collaborative recommendations.

The purpose of this paper is to find the best learning Bayesian networks algorithm for collaborative recommendation and modify the algorithm to decrease its computational cost on massive datasets.

We first compare the prediction accuracy of four scoring-based learning Bayesian networks algorithms (AIC, MDL, UPSM, and BDeu) and two conditional-independence-based (CI-based) learning Bayesian networks algorithms (MWST, and Polytree-MWST) using actual massive datasets. The results show that (1) for large networks, the scoring-based algorithms have less accuracy of prediction than the CI-based algorithms and (2) when the scoring-based algorithms use a greedy search to learn a large networks, the algorithms that make a lot of arcs tend to have less prediction accuracy than the those that make fewer arcs. Next, we propose a learning algorithm based on MWST for collaborative filtering of massive datasets. The proposed algorithm employs a traditional data mining technique, the "a priori algorithm", to quickly calculate the amount of mutual information, which is needed in MWST, from massive datasets. We compare the original MWST algorithm and the proposed algorithm on actual data, and the comparison show the effectiveness of the proposed algorithm.

2. Background

2.1 Bayesian networks

Consider a domain $\mathbf{U}$ of n discrete random variables $X_1, ..., X_n$, where each variable X_i takes a state $K \in \{0, \ldots, r_i - 1\}$.

A Bayesian network is an annotated acyclic directed graph that encodes a joint probability distribution over a set of random variables $\mathbf{U}$. Formally, a Bayesian network for $\mathbf{U}$ is a pair $B =< B_S, \Theta >$. The first component, B_S, is an acyclic directed graph whose vertices correspond to the random variables $X_1, ..., X_n$, and whose edges represent directed dependencies between the variables. The graph B_S encodes independence assumptions:

each variable X_i is independent of its nondescendants given its parents in B_S. The second component, Θ, represents the set of parameters that quantifies the network. It contains a parameter $\theta_{ijk} = p(X_i = k|\Pi_i = j)$ for each possible value k of X_i and j of Π_i, where Π_i denotes the set of parents of X_i in B_S. A Bayesian network B defines a unique joint probability distribution over $\mathbf{U}$ given by

$$p(X_1, \ldots, X_n|B) = \prod_{i=1}^{n} p(X_i|\Pi_i, B_S) \tag{2.1}$$

2.2 Learning algorithms for Bayesian networks

The problem of learning a Bayesian network can be informally stated as: Given a training data $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ of instances of $\mathbf{U}$, find a network B that best matches $\mathbf{X}$.

The common approach to this problem is to introduce a scoring metric that evaluates each network with respect to the training data and then to search for the best network according to this function.

There are two ways to view a Bayesian network, each suggesting an approach to learning. First, a Bayesian network is a structure that encodes the joint distribution of the attributes. This suggests that the best Bayesian network is the one that fits the data and leads to scoring-based learning algorithms which seek a structure that maximizes scoring metrics such as AIC, MDL, UPSM, and BDeu (Akaike, 1974; Cooper and Herskovits, 1992; Heckerman et al., 1995; Rissanen, 1983).

Second, the Bayesian network structure encodes a group of conditional independence relationships among the nodes, according to the concept of d-separation (Pearl, 1988). This suggests learning the Bayesian network structure by identifying the conditional independence relationships among the nodes. Using statistical tests such as the Chi-squared test and the mutual information test, we can find the conditional independence relationships among the attributes and use those relationships as constraints to construct a Bayesian network. These algorithms are referred as CI-based algorithms or constraint-based algorithms.

Heckerman et al. (1997) compare these two general approaches to learning and show that the scoring-based methods often have certain advantages over the CI-based methods in terms of modeling a distribution. However, Friedman et al. (1997) theoretically show that the general scoring-based methods may result in poor classifiers, because a good classifier maximizes a different function, namely classification accuracy. Greiner et al. (1997) reach the same conclusion, albeit via a different analysis. Cheng and Greiner (1999) also insists that CI-based methods are superior to scoring based algorithms for classifications. However, these studies focus on only Bayesian classifier models; thus we cannot expand these results to Bayesian network models.

In this paper, we compare the prediction accuracies of four scoring-based learning Bayesian networks algorithms (AIC, MDL, UPSM, and BDeu) and two CI-based learning Bayesian network algorithms (MWST, and Polytree-MWST) using actual massive datasets.

2.3 Scoring-based algorithms

2.3.1 UPSM – Uniform Prior Score Metric

This subsection introduces the most popular scoring metrics: the Dirichlet Prior Score Metric (DPSM) and Uniform Prior Score Metric (UPSM) (Cooper and Herskovits, 1992).

Let θ_{ijk} be a conditional probability parameter of $X_i = k$ when $\Pi_i = j$. The likelihood $L(\Theta \mid \mathbf{X}, B_S)$ is given by

$$L(\Theta \mid \mathbf{X}, B_S) \propto \prod_{i=1}^{n} \prod_{j=1}^{q_i} \prod_{k=0}^{r_i-1} \theta_{ijk}^{N_{ijk}}, \tag{2.2}$$

where $\Theta = \{\theta_{ijk}\}(i = 1, \cdots, n, j = 1, \cdots, q_i, k = 0, \cdots, r_i - 1)$, q_i is the number of instances of Π_i, r_i is the number of states of x_i, N_{ijk} is the number of samples of $X_i = k$ when $\Pi_i = j$, and $\mathbf{X}$ is a multinomial sample from Bayesian network $B =< B_S, \Theta >$.

We assume a Dirichlet distribution as a conjugate prior distribution, which is a class of likelihood functions if the resulting posterior distributions are in the same family, of the multinomial distribution,

$$p(\Theta \mid B_S) = \prod_{i=1}^{n} \prod_{j=1}^{q_i} \prod_{k=0}^{r_i-1} \frac{\Gamma(\sum_{k=0}^{r_i-1} N'_{ijk})}{\prod_{k=0}^{r_i-1} \Gamma(N'_{ijk})} \prod_{k=0}^{r_i-1} \theta_{ijk}^{N'_{ijk}-1}, \tag{2.3}$$
$$N'_{ijk} > 0(k = 0, \ldots, r_i - 1),$$

where N'_{ijk} is the hyper-parameter of the prior distribution corresponding to the multinomial sample N_{ijk}.

Consequently, we obtain the posterior as follows:

$$p(\Theta \mid \mathbf{X}, B_S) \propto \prod_{i=1}^{n} \prod_{j=1}^{q_i} \prod_{k=0}^{r_i-1} \theta_{ijk}^{N'_{ijk}+N_{ijk}-1}. \tag{2.4}$$

Thus, if the prior distribution for Θ has a Dirichlet distribution, so does the posterior distribution for Θ.

Given the Dirichlet distribution's properties, Cooper and Herskovits (1992) and Heckerman et al. (1995) employed an unbiased estimator, i.e., the expectation of the parameter estimator $\widehat{\theta_{ijk}}$:

$$\widehat{\theta_{ijk}} = \frac{N'_{ijk} + N_{ijk}}{N'_{ij} + N_{ij}}, (k = 0, \cdots, r_i - 2), \tag{2.5}$$

where $N'_{ij} = \sum_{k=1}^{r_i-1} N'_{ijk}, N_{ij} = \sum_{k=1}^{r_i-1} N_{ijk}$.

Furthermore, the predictive distribution can be obtained as follows:

$$\begin{aligned} p(\mathbf{X} \mid B_S) &= \int_{\Theta} p(\mathbf{X} \mid \Theta, B_S) p(\Theta \mid B_S) d\Theta \qquad (2.6) \\ &= \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{\Gamma(N'_{ij})}{\Gamma\left[\sum_{k=0}^{r_i-1} (N'_{ijk} + N_{ijk})\right]} \prod_{k=0}^{r_i-1} \frac{\Gamma(N'_{ijk} + N_{ijk})}{\Gamma(N'_{ijk})} \end{aligned}$$

$$= \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{\Gamma(N'_{ij})}{\Gamma(N'_{ij} + N_{ij})} \prod_{k=0}^{r_i - 1} \frac{\Gamma(N'_{ijk} + N_{ijk})}{\Gamma(N'_{ijk})}$$

The predictive distribution in (2.6) is called the "Dirichlet Prior Score Metric, (DPSM)". We maximize DPSM in order to select the true structure of the Bayesian network.

In particular, (Cooper and Herskovits, 1991, 1992) assumed that the prior distribution has a uniform $N'_{ijk} = 1, (i = 1, \cdots, N, j = 1, \cdots, q_i, k = 0, \cdots, r_i - 1)$ and derived the following criterion:

$$p(\mathbf{X} \mid B_S) = \int_{\Theta} p(\mathbf{X} \mid \Theta, B_S) p(\Theta \mid B_S) d\Theta \tag{2.7}$$

$$\propto \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \prod_{k=0}^{r_i - 1} N_{ijk}!.$$

This criterion led to their famous causal discovery program K2 (Cooper and Herskovits, 1992), which we call the "Uniform Prior Score Metric (UPSM)".

2.3.2 BDeu – Likelihood equivalence Bayesian Dirichlet with uniform prior

Heckerman et al. (1995) introduced a likelihood equivalence assumption: if two structures are equivalent, their parameter joint probability density functions are identical. Theoretically, the likelihood equivalence assumption is written as follows:

Given two network structures B_{S1} and B_{S2} such that $p(B_{S1} \mid \xi) > 0$ and $p(B_{S2} \mid \xi) > 0$, if B_{S1} and B_{S2} are equivalent, then $p(\Theta_{B_{S1}} \mid B_{S1}, \xi) = p(\Theta_{B_{S2}} \mid B_{S2}, \xi)$.

Heckerman et al. (1995) pointed out that formula (2.10) does not satisfy the likelihood equivalence assumption.

Furthermore, Heckerman et al. (1995) used the likelihood equivalence assumption, instead of the Dirichlet distribution assumption, and derived the same formula as (2.9). They also showed the following constraint about the hyper-parameters is a sufficient condition for satisfying the likelihood equivalence assumption:

$$N'_{ijk} = N' \cdot p(x_i = k, \Pi_i = j \mid B_S^h, \xi), \tag{2.8}$$

where N' is the equivalent sample size determined by users and B_S^h is the hypothetical Bayesian network structure which reflects the user's prior knowledge. They called this metric the "likelihood-equivalence Bayesian Dirichlet score metric" (BDe). Buntine (1991)'s uniform prior constraint $N'_{ijk} = N'/(r_i q_i)$ is considered to be a special case of the BDe metric, and Heckerman et al. (1995) call this special case "BDeu" ("u" for uniform prior). Buntine noted that this metric has the property of likelihood equivalence.

2.3.3 AIC – Akaike's Information Criterion

Akaike's Information Criterion (AIC) (Akaike, 1974) is a scoring-based method. The AIC scoring metric is based on expectation log likelihood with a penalty. Using maximal log likelihood $l_m(\Theta|D, B_S)$, the AIC scoring metric can be expressed as

$$AIC(B_S, D) = \log p(B_S) + l_m(\Theta|D, B_S) - \sum_{i=1}^{n} q_i(r_i - 1) \tag{2.9}$$

$$= \log p(B_S) + \sum_{i=1}^{n}\sum_{j=1}^{q_i}\sum_{k=1}^{r_i} N_{ijk} \log \frac{N_{ijk}}{N_{ij}} - \sum_{i=1}^{n} q_i(r_i - 1),$$

where N_{ijk} denotes the number of cases in the given training data D in which the variable X_i took its kth value $(k = 1, 2, ..., r_i)$ and its parent $Pa(X_i)$ was instantiated as its jth value $(j = 1, 2, ..., q_i)$, and $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$. In this paper, we assume that the prior distribution $p(B_S)$ is uniform because we have no prior information about the true network structure.

2.3.4 MDL – Minimum Description Length

The Minimum Description Length (MDL) (Rissanen, 1983) is a scoring-based method that can judge the quality of a network structure. The basic idea behind the MDL is to make a tradeoff between model simplicity and data fit by the minimizing the length of a joint description of the model and the data by assuming the model is correct. The MDL score metric is expressed as

$$\begin{aligned} MDL(B_S, D) &= \log p(B_S) + l_m(\Theta|D, B_S) - \frac{1}{2}\sum_{i=1}^{n} q_i(r_i - 1)\log N \qquad (2.10) \\ &= \log p(B_S) + \sum_{i=1}^{n}\sum_{j=1}^{q_i}\sum_{k=1}^{r_i} N_{ijk} \log \frac{N_{ijk}}{N_{ij}} - \frac{1}{2}\sum_{i=1}^{n} q_i(r_i - 1)\log N. \end{aligned}$$

2.4 Greedy search

It is known that the search problem for the network structure which maximize the scoring metrics is NP-complete (Chickering, 1995). Therefore, a heuristic search algorithm is required. In this paper, we employ the following algorithm for the search problems using the scoring metrics mentioned before.

Cooper and Herskovits (1992) developed a greedy search algorithm that searches for a network structure that approximately maximizes the scoring metrics. That is, the search space is the set of all network structures. For each node i, this search algorithm locally finds a value Π_i that maximizes the scoring metrics of the local network structures with X_i and Π_i. The single operation in this search algorithm is the addition of a parent to a node. The algorithm proceeds as follows: Assume an ordering of nodes such that if X_i precedes X_j in the order, an arc from X_j to X_i is not allowed. Let $Pred(X_i)$ be the set of nodes that precede X_i in the ordering. Initially, set the parents Π_i of X_i to empty and compute the scoring metric. Determine the node in $Pred(X_i)$ which most increases the scoring metric of the local network structure with X_i, Π_i, and the node in $Pred(X_i)$. Add the determined node to Π_i. This procedure is continued until the addition of a node does not increase the scoring metric.

2.5 CI-based algorithms

2.5.1 MWST

Maximum Weighted Spanning Tree (MWST) (Chow and Lin, 1968) is a CI-based method that makes a tree structure using the mutual information measure $I(X_i, X_j)$:

$$I(X_i, X_j) = \sum_{x_i, x_j} p(x_i, x_j) \log \frac{p(x_i, x_j)}{p(x_i)p(x_j)} \tag{2.11}$$

MWST makes a tree structure in which each node can only have one parent. The MWST procedure is:

1. Compute the mutual information measure $I(X_i, X_j)$ for all pairs of nodes.
2. Add to the list all pairs of nodes whose mutual information measure is greater than a threshold ε.
3. Order the pairs of nodes in the list by the mutual information measure.
4. Examine the first pair of nodes in the list and connect them with a non-directed arc.
5. Examine the next pair of nodes in the list, and connect them with a non-directed arc unless it forms a loop, in which case discard it and examine the next pair.
6. Repeat step 5 until $n - 1$ arcs have been connected or all pairs of nodes in the list have been examined.
7. Select a root node arbitrarily, and set the directions of all arcs to go from the root node to leaf nodes.

Here, we select a root node according to the ordering of the nodes that we determined for the greedy search algorithm (the first node is a root node). MWST requires $O(n^2)$ mutual information tests.

2.5.2 Polytree-MWST

Polytree-MWST (Pearl, 1988), which is an extended MWST, makes trees which allow multiple parents. In this study, we decide beforehand the directions of arcs according to the ordering of the nodes for the greedy search algorithm to use Polytree-MWST.

The Polytree-MWST procedure is:

1. Compute the mutual information measure $I(X_i, X_j)$ for all pairs of nodes.
2. Add to the list all pairs of nodes whose mutual information measure is greater than a threshold ε.
3. Order the pairs of nodes in the list by the mutual information measure.
4. Examine the first pair of nodes in the list, and connect them with a directed arc.
5. Examine the next pair of nodes in the list, and connect them with a directed arc unless it forms a loop, in which case discard it and examine the next pair.
6. Repeat step 5 until $n - 1$ arcs have been connected or all pairs of nodes in the list have been examined.

Polytree-MWST also requires $O(n^2)$ mutual information tests.

3. Experiments

This section compares the prediction accuracies of UPSM, BDeu, AIC, MDL, MWST, and Polytree-MWST operating on actual data sets for collaborative recommendations.

3.1 Experiment method

The prediction accuracy comparison used user browsing history data stored in a commercial web server. The web site provides information on recruiting, travel, housing, cars, qualifications, marriages, restaurants, education, healthcare, medical care, child-rearing, and so on. The site has over 250,000 pages and over four million users. An unique feature of the data is that the numbers of visits to most pages are very small; for almost all pages, the probability of it being visited is much smaller than the probability of it not being visited. Therefore, we first extracted the 100 pages with the largest numbers of visits from the 6,042 pages that belong to a certain category on the Web site. We regard whether or not the page was visited as a random variable for the corresponding node in a Bayesian network. The variable for each node has one of two values: 0 if the number of visits is zero and 1 if the number of visits is one or more. Here, the t-th user's data $\mathbf{x}_t$ in the dataset $\mathbf{X} = \{\mathbf{x}_1, ..., \mathbf{x}_N\}$ is represented as follows:

$$\mathbf{x}_t = \{x_1^t, ..., x_n^t\}, \text{ where } x_i^t = \begin{cases} 1 & : \ t\text{-th user visited page } i \\ 0 & : \ \text{otherwise} \end{cases}$$

The dataset $\mathbf{X}$ was divided into training data for learning Bayesian networks and test data for evaluating the prediction accuracy of the learning algorithms. The training data was a random sampling of 5,000 users who had visited any of the extracted 100 pages. The test data was the remaining data of the $\mathbf{X}$ that excluded the training data, and its size was around 15,000.

The experiments compared the six learning algorithms mentioned in section 2. We used the greedy search procedure using the node order for the scoring-based learning algorithms. Here, the nodes were placed in descending order according to the number of visiting users for the pages that correspond to the nodes. For MWST and Polytree-MWST, the thresholds for the amount of mutual information were set to $\varepsilon = 0.001$. For BDeu, the prior distribution hyper-parameter was determined to be $N' = 10.0$ (Yang and Chang, 2002). Figures 1–6 show the structures of the Bayesian networks constructed using UPSM, BDeu, AIC, MDL, tMWST, and Polytree-MWST, respectively. Table 2 shows the numbers of arcs in the networks constructed by the learning algorithms. The results in Table 2 shows that the complexity of the constructed network structure depends on the learning algorithm.

To compare the prediction accuracies of the learning algorithms, first we randomly sampled some visiting data from t-th user data $\mathbf{x}_t$. This sampled data is used to infer the visiting probabilities of t-th user data excepting the sampled data by Bayesian network constructed from the training data and is called as "observations". Next we evaluated the mean squared errors between the visiting probability, which is infered using the observa-

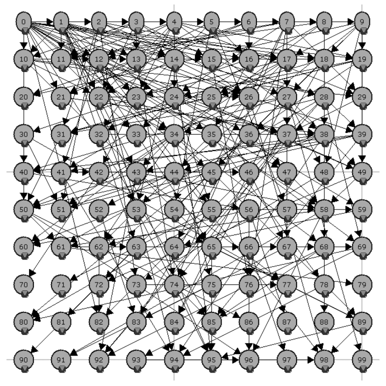


Figure 1: The network constructed by UPSM

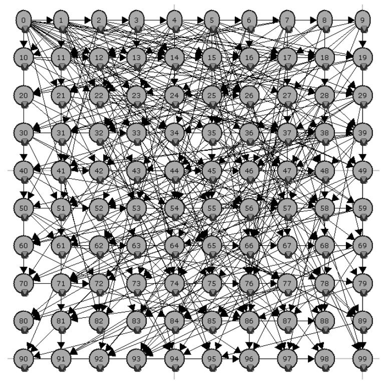


Figure 2: The network constructed by BDeu

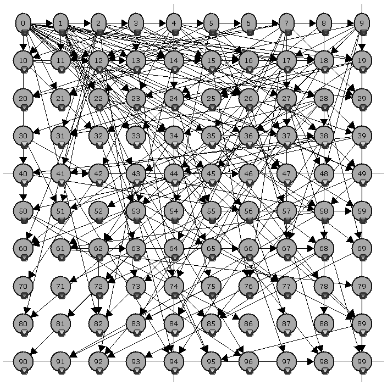


Figure 3: The network constructed by AIC

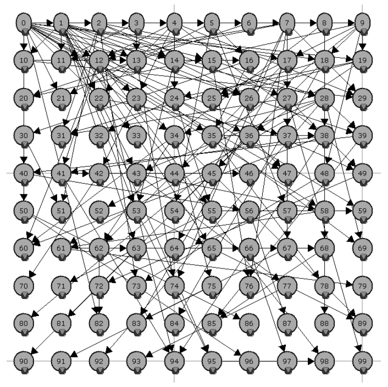


Figure 4: The network constructed by MDL

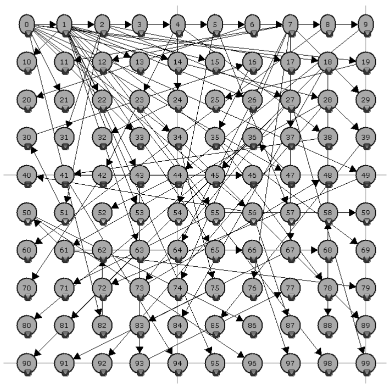


Figure 5: The network constructed by MWST

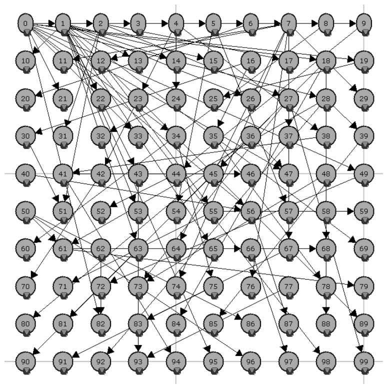


Figure 6: The network constructed by Polytree-MWST

Table 2: Numbers of arcs in networks constructed by the learning algorithms.

Algorithm	Number of Arcs
UPSM	261
BDeu	319
AIC	218
MDL	174
MWST	99
Polytree-MWST	99

tions and a Bayesian network constructed by the training data, and the actual data and repeated this procedure for all the sampled users.

The more details of the experiment procedure are as follows: We randomly sampled M users from the test data who had visited at least 15 of the extracted $100(=n)$ pages. Here, $M = 50$. For the sampled user t, furthermore, we randomly sampled k ($k = 1, \ldots, 14$) pages from the pages which user t had visited. The observations $\mathbf{e}_k^t$ corresponding to the sampled k pages are used for Bayesian inference. That is, the visiting probabilities for all the pages which are not used as the observations $\mathbf{e}_k^t$ are propagated by the observations in the constructed Bayesian network.

We obtained the following mean squared error (MSE) between the inferred visiting probability of page i given observations $\mathbf{e}_k^t$ and actual data x_i^t according to Breese et al. (1998);

$$MSE(k) \equiv \frac{1}{M} \sum_{t=1}^{M} \sum_{x_i^t \in \mathbf{x}_t \backslash \mathbf{e}_k^t} \frac{(x_i^t - p(X_i = 1|\mathbf{e}_k^t))^2}{n-k}. \tag{3.1}$$

The experiments were performed on a computer equipped with a Pentium D 3.20-GHz CPU and 3.50 GB of RAM running Windows XP Professional Service Pack 2.

3.2 Results

The results of the experiments are shown in Table 3 and Fig. 7. In Fig. 7, the horizontal axis indicates the number of observations and the vertical axis indicates the MSE values between the inferred visiting probability of page i given observations $\mathbf{e}_k^t$ and actual data x_i^t.

The MSE values for the learning algorithms tend to monotonically decrease as the number of observations increases. This means the prediction accuracy of the Bayesian network certainly improved as a result of adding observations for Bayesian inference. More specifically, the MSE values for BDeu decrease as the number of observations increases up to five, but they show no large change for five or more observations.

When the number of observations was one, there was no large difference between the MSEs of the learning algorithms. As the number of observations increases, the MSEs gradually become classified into the three groups.

The first group consists of MWST and Polytree-MWST, and it has the best prediction accuracy over the number of observations. The second group consists of UPSM, AIC

Table 3: Mean squared errors. Lower scores indicate better performance.

Number of observations	UPSM	BDeu	AIC	MDL	MWST	PolytreeMWST
1	0.1360	0.1423	0.1356	0.1408	0.1256	0.1304
2	0.1272	0.1368	0.1277	0.1297	0.1122	0.1133
3	0.1142	0.1308	0.1172	0.1216	0.0975	0.0995
4	0.1096	0.1236	0.1147	0.1103	0.0884	0.0929
5	0.1028	0.1196	0.1069	0.1034	0.0803	0.0834
6	0.1020	0.1192	0.1011	0.1008	0.0750	0.0775
7	0.0968	0.1210	0.0990	0.0941	0.0709	0.0693
8	0.0960	0.1191	0.0957	0.0882	0.0621	0.0668
9	0.0907	0.1193	0.0944	0.0850	0.0595	0.0618
10	0.0910	0.1180	0.0935	0.0815	0.0534	0.0581
11	0.0891	0.1211	0.0898	0.0776	0.0519	0.0540
12	0.0875	0.1224	0.0871	0.0742	0.0454	0.0532
13	0.0858	0.1166	0.0852	0.0688	0.0421	0.0454
14	0.0844	0.1192	0.0822	0.0664	0.0393	0.0440

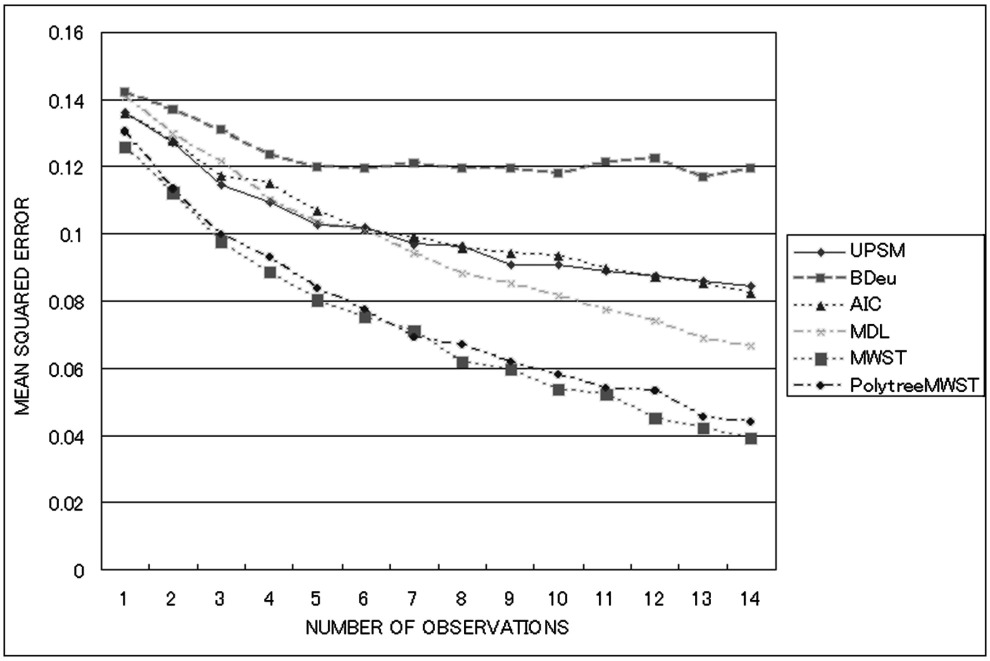


Figure 7: Mean squared errors. Lower scores indicate better performance.

and MDL, and it has the second-best prediction accuracy. The third group consists of BDeu and it has the worst prediction accuracy. Thus, the results show the CI-based algorithms (MWST and Polytree-MWST) are superior to the scoring-based algorithms (UPSM, BDeu, AIC, and MDL). We will discuss why the scoring-based algorithms are less effective in the next section.

Table 4: Running time (seconds) of the learning algorithm and the probabilities propagation algorithm.

Algorithm	Learning Time (second)	Probabilities Propagation Time (second)
UPSM	1522	6.5
BDeu	5697	25
AIC	659	2.4
MDL	385	0.89
MWST	42	0.034
Polytree-MWST	42	0.033

Table 4 lists the mean computing time required by each algorithm for learning the Bayesian networks and for the visiting probabilities propagation. For the probabilities propagation, we used the Message Passing algorithm (Pearl, 1988) for the networks constructed by MWST and Polytree-MWST since the networks are singly connected, and used theLoopy Belief Propagation algorithm (Weiss, 1997) for the networks constructed by the other algorithms since these networks are multiply connected. We know that compared with the other algorithms, MWST and Polytree-MWST have much shorter computation times for learning Bayesian networks and for probabilities propagation.

4. Why are the scoring-based algorithms less effective?

The results of the previous section indicate that the CI-based algorithms (MWST, Polytree-MWST) have better prediction accuracy than the scoring-based algorithms (UPSM, BDeu, AIC, and MDL). Thus, we believe that use of MWST or Polytree-MWST is optimum for collaborative filtering using Bayesian networks.

Let us consider why the prediction accuracies of a Bayesian network learned by the scoring-based algorithms were lower.

DPSM including UPSM and BDeu is known to converge to MDL (2.13) (Bouckaert, 1994; Suzuki, 1993, 1998). Hence, we can treat DPSM as the same as the MDL criteria for the purpose of this discussion. MDL and AIC have a log-likelihood term, but this term is considered to be less effective for large networks.

In a classification problem for a class variable C with attributes $A_1, \ldots, A_n$, Friedman et al. (1997) showed that the prediction accuracy of a Bayesian network classifier trained by a scoring metric using the following log likelihood is not good:

$$l(\Theta|\mathbf{X}, B_S) = \sum_{t=1}^{N} \log p(c^t|a_1^t, \ldots, a_n^t) + \sum_{t=1}^{N} \log p(a_1^t, \ldots, a_n^t), \tag{4.1}$$

where Θ is a set of conditional probabilities parameters, $\mathbf{X}$ is the dataset used for learning and $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, c^t is the t-th learning dataset value of C, which is the class variable, and $a_1^t, ..., a_n^t$ represent the l-th learning dataset values of the attributes $A_1, \ldots, A_n$. In Eq. (4.1), if the number of attributes increases, the probability of the second term on the right side, $p(a_1^t, \ldots, a_n^t)$, becomes smaller, and $\log p(a_1^t, \ldots, a_n^t)$ becomes larger in the negative direction. In contrast, the probability of the first term on the right side, $p(c^t|a_1^t, \ldots, a_n^t)$, does not decrease as the number of attribute increases. Accordingly, the value of $\log p(c^t|a_1^t,\dots,a_n^t)$ becomes smaller than $\log p(a_1^t,\dots,a_n^t)$ in the negative direction. If the number of attributes is large, this difference becomes very large. Thus, the serious error of the first term on the right side is hidden by the value of the second term on the right side and is hardly reflected in the log likelihood. Accordingly, it is difficult to select the optimum structure for representing the relations of the class variable and the attributes, and the prediction accuracy often suffers. If we try to rewrite the log likelihood equation for the general Bayesian network by writing x_i^t as the value of node X_i of the t-th learning data and writing Π_i^t as the value of the parent node set of node X_i, we get

$$l(\Theta|\mathbf{X},B_S) = \sum_{t=1}^{N}\sum_{i=1}^{n}\log p(x_i^t|\Pi_i^t). \tag{4.2}$$

Moreover, the likelihood for the local structure B'_S given a fixed node i, which is used in the greedy search algorithm, is

$$l(\Theta|\mathbf{X},B'_S) = \sum_{t=1}^{N}\log p(x_i^t|\Pi_i^t) + \sum_{t=1}^{N}\log p(\Pi_i^t). \tag{4.3}$$

Regarding $l(\Theta|\mathbf{X},B'_S)$, equation (4.3) means that when a given node has many parent nodes, the second term on the right side becomes larger in the negative direction the first term than the first term. Accordingly, the value of the first term is obscured by the second term and so is not well reflected in the log likelihood $l(\Theta|\mathbf{X},B'_S)$. Therefore, when a given node has a large number of parent nodes, the prediction accuracy of that node is often not good. In addition, the greedy search, which uses an ordering of the nodes and is generally used in the structure search, searches the parents node set of a node by computing the log likelihood $l(\Theta|\mathbf{X},B'_S)$. In that case, it is possible for a node to have a large number of parents when learning a Bayesian network that has a large number of nodes, so it might not be possible to correctly estimate a structure that represents the relations of nodes to their parent nodes through the entire network. For that reason, when the greedy search is used to learn a Bayesian network with a large number of nodes, the prediction accuracy of all of the network nodes might not be good. In other words, when the greedy search is employed, learning algorithms that make a lot of arcs in the network tend to have lower prediction accuracy than those that make fewer arcs. The results in Table 2 and in Fig. 7 confirm this conclusion. That is, the order of the number of arcs made by the learning algorithms in Table 2 is the same as the order of the prediction accuracies of the learning algorithms in Fig. 7. These analyses can be summarized as follows:

1. For large networks, the scoring-based algorithms have lower accuracy of prediction than the CI-based algorithms.
2. When the scoring-based algorithms use a greedy search to learn a large network, the algorithms that make a lot of arcs tend to have lower prediction accuracy than ones that make fewer arcs.

5. Modified learning algorithm for massive datasets

The prediction accuracy comparison experiments revealed that MWST and Polytree-MWST are superior to the scoring-based algorithms in both prediction accuracy and computation time. Nevertheless, the amount of computation still requires $O(n^2)$ for these algorithms, so computation time also becomes very large if we try to learn a Bayesian network from a massive dataset. To cope with this problem, this section proposes a modified MWST algorithm which employs a traditional data mining technique, the "a priori" algorithm (Agrawal et al., 1993; Agrawal and Srikant, 1994), to quickly calculate the amount of mutual information from massive datasets.

5.1 Hybrid algorithm combined MWST and a priori algorithm

Most of the data subject to collaborative filtering is purchasing history data taken from a massive dataset. For this kind of data, if $X_i = 1$ means the selection of item i and $X_i = 0$ means non-selection of item i, then $p(X_i = 1)$ tends to be very small and $p(X_i = 0)$ tends to be very large. Note that $X_i = 0$ is missing data rather than observed data. Therefore, the data for which $X_i = 1$ should be used with priority, with the result that item i for which $p(X_i = 1)$ is large is highly reliable. The MWST requires a calculation of the amount of mutual information $I(X_i, X_j)(i = 1, ..., n; j = 1, ..., n; i \neq j)$ between any two variables pairs $(X_i, X_j)(i = 1, ..., n; j = 1, ..., n; i \neq j)$ for all variables. Furthermore, $p(X_i = 1, X_j = 1)$, $p(X_i = 1, X_j = 0)$, $p(X_i = 0, X_j = 1)$, and $p(X_i, = 0, X_j = 0)$ are needed in order to obtain $I(X_i, X_j)(i = 1, ..., n, j = 1, ..., n, i \neq j)$. As mentioned before, $p(X_i = 1, X_j = 0)$, $p(X_i = 0, X_j = 1)$, and $p(X_i = 0, X_j = 0)$ are missing data because they include $X_i = 0$, or $X_j = 0$. Thus, $p(X_i = 1, X_j = 1)$ should be used with priority. In other words, if we observe data such as $p(X_i = 1, X_j = 1) = 0$, it means that we have no information about the variable pair (X_i, X_j). That is, it means nothing to calculate the amount of mutual information about data such as $p(X_i = 1, X_j = 1) = 0$. Hence, the proposed algorithm prunes variable pairs such as $p(X_i = 1, X_j = 1) = 0$ before calculating the amount of mutual information to reduce the computational cost for learning Bayesian networks. The remaining problem is how to quickly prune the variables pairs for massive datasets.

The "a priori" algorithm (Agrawal et al., 1993; Agrawal and Srikant, 1994) is used in data mining to prune infrequent variables. The algorithm computes frequent itemsets from a transactions database over multiple iterations. Each iteration involves (1) candidate generation and (2) candidate counting and selection. Utilizing knowledge about infrequent itemsets obtained from previous iterations, the algorithm prunes a priori those candidate itemsets that cannot become frequent. After discarding every candidate itemset that has an infrequent subset, the algorithm enters the candidate counting step.

The a priori algorithm is outlined as follows. Let F_k be the set of frequent itemsets of size k, let C_k be the set of candidate itemsets of size k, and let F_1 be the set of frequent items. We start from $k = 1$.

1. **for** all items in frequent itemset F_k **repeat** steps 2-4.
2. Generate new candidates C_{k+1} from F_k.
3. **for** each transaction T in the database, and increment the count of all candidates in C_{k+1} that are contained in T.
4. Generate frequent itemsets F_{k+1} of size k from candidates in C_{k+1} with a threshold support.

The final solution is $\bigcup_k F_k$. Here, "support" and "confidence" are technical terms of the association rule: "support" means the joint probability of all variables in the itemsets, and "confidence" means the conditional probability of the target variable given the other variables in the itemsets. A key observation is that every subset of a frequent itemset is also frequent. This implies that a candidate itemset in C_{k+1} can be pruned if even one of its subsets is not contained in F_k. Next, the a priori algorithm searches the itemsets of which the confidence is more than a threshold. Namely, it prunes the itemsets of which the support is less than a threshold and then searches the itemsets of which confidence is high in the remaining itemsets.

The proposed algorithm employs the a priori algorithm's pruning technique for MWST. That is, the main idea is to use the amount of mutual information instead of the confidence in the a priori algorithm. The proposed algorithm is shown in Fig. 8.

First, the algorithm quickly prunes variables pairs such as support $p(X_i = 1, X_j = 1)$ less than a threshold by using the a priori algorithm, and then it prunes variables pairs such as the amount of mutual information $I(X_i, X_j) < \varepsilon$, from the remaining variables pairs. The algorithm prunes variables such as support $P(X_i = 1) < \delta_1$ and then prunes the variables pairs such as support $p(X_i = 1, X_j = 1) < \delta_2$. Note that the amount of mutual information can be calculated using the previously calculated values of supports $p(X_i = 1)$ and $p(X_i = 1, X_j = 1)$.

5.2 Experiments and results

We performed experiments to evaluate the learning time and the prediction accuracy of the hybrid algorithm. For the training data, 10,000 users' data were randomly sampled from among the users who had visited any of the 6042 pages that had been selected from the Web site in section 4. The variable for each node had two values that simply indicated whether or not the page had been visited: 0 for zero visits or 1 for one or more visits. Using the training data, we constructed three Bayesian networks: one was trained by the ordinary MWST, one was trained by the hybrid algorithm with thresholds "$\delta_1 = 0.00017$"(= 1/6042, the probability of random sampling a node) and "$\delta_2 = 0.001$", and the other one was trained by the hybrid algorithm with t thresholds "$\delta_1 = 0.00017$" and "$\delta_2 = 0.0001$". The threshold for the amount of mutual information for MWST was $\varepsilon = 0.001$, as in section 4. The network constructed by the hybrid algorithm with "$\delta_1 = 0.00017$" and "$\delta_2 = 0.0001$" is shown in Fig. 9. The prediction accuracies were evaluated with test data for 50 users who had visited 15 or more of the 6042 pages. The evaluation method was the one described in section 4.1. For the network constructed by

```
Let $n$ be the number of items
Let $N$ be the number of users
Let $X_i$ be the binary (0 or 1) variable corresponding to $i$-th item viewed or not
Let $\mathbf{U} = \{X_1, ..., X_n\}$
Let $L \subset \mathbf{U}, V \subset \mathbf{U}$
Let $x_i^t$ be $X_i$'s value of $t$-th user's data
Let $\mathbf{x}_t = \{x_1^t, ...x_n^t\}$ be the $t$-th user's data
Let $Count[X_i]$ be the number of $X_i = 1$ for all user's data
Let $Count[X_i, X_j]$ be the number of $X_i = 1$ and $X_j = 1$ for all user's data
Let $MWST(I, \mathbf{X})$ be a function for MWST algorithm for variable set $I$ and dataset $\mathbf{X}$,
return a Bayesian network $B$
Input Dataset $\mathbf{X} = \{\mathbf{x}_1, ..., \mathbf{x}_N\}$
Output a Bayesian network $B$

1: for $(t = 1 : N)$ do
2:   for $(i = 1 : n)$ do
3:     if $x_i^t = 1$ then
4:       $Count[X_i] = Count[X_i] + 1$
5:     end if
6:   end for
7: end for
8: for $(i = 1 : n)$ do
9:   if $(Count[X_i]/N) \geq \delta_1$ then
10:    $V \leftarrow X_i$
11:  end if
12: end for
13: for $(t = 1 : N)$ do
14:   for (each $X_i \in V$) do
15:     for (each $X_j \in V \backslash X_i$) do
16:       if $j > i$ and $x_i^t = 1$ and $x_j^t = 1$ then
17:         $Count[X_i, X_j] = Count[X_i, X_j] + 1$
18:       end if
19:     end for
20:   end for
21: end for
22: for (each $Count[X_i, X_j]]$) do
23:   if $(Count[X_i, X_j]/N) \geq \delta_2$ then
24:     if $X_i \notin L$ then
25:       $L \leftarrow X_i$
26:     end if
27:     if $X_j \notin L$ then
28:       $L \leftarrow X_j$
29:     end if
30:   end if
31: end for
32: return $MWST(L, \mathbf{X})$
```

Figure 8: Hybrid algorithm combining MWST and a priori algorithm

the hybrid algorithm, if a randomly selected observation node was not connected to any other nodes, the observation was not used for probabilities propagation in the network. The computer used in the experiment was the same as described in section 4. The learning

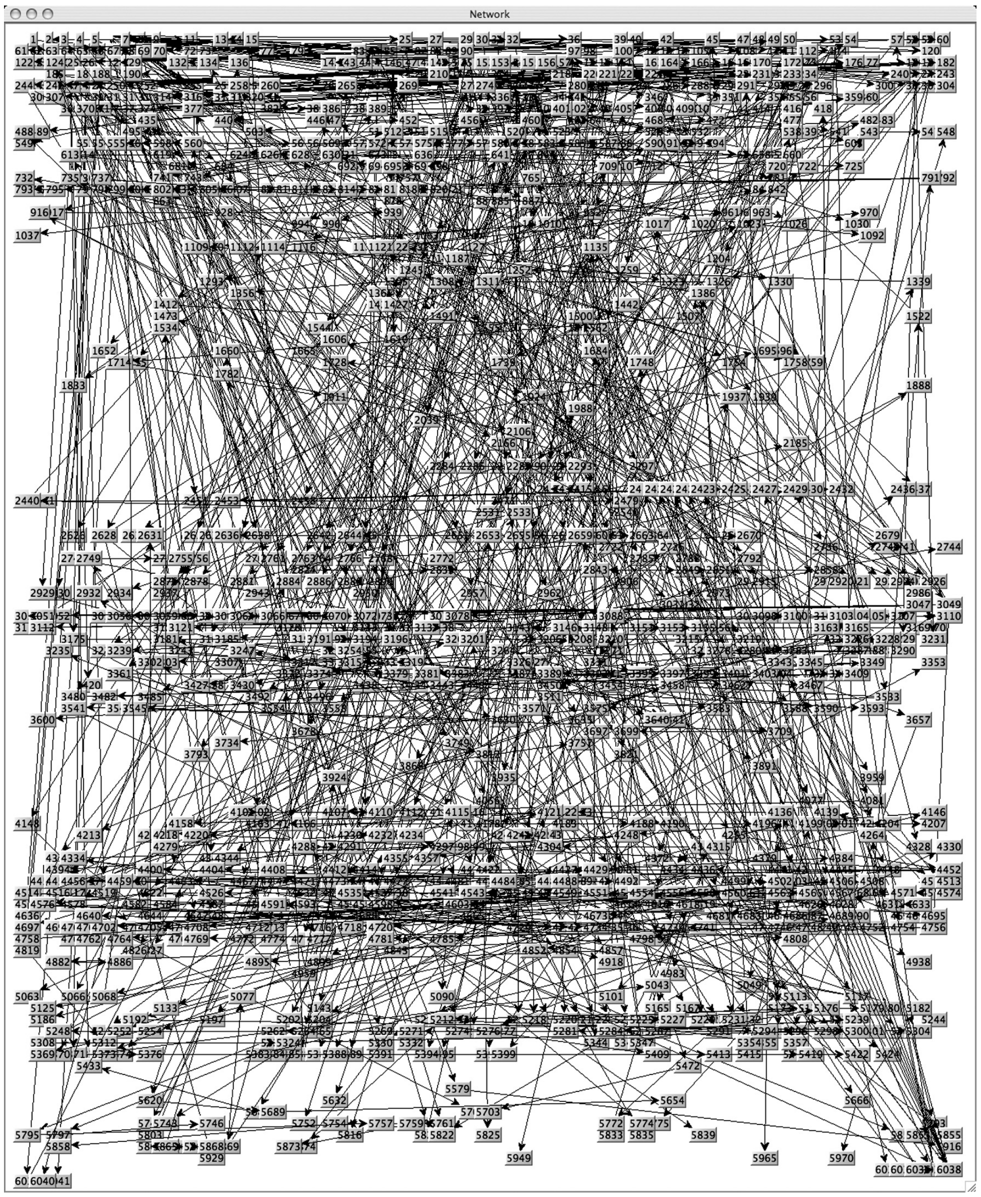


Figure 9: The network constructed by the hybrid algorithm.

time for the algorithms and the number of arcs in the obtained networks are shown in Table 5.

Table 5 shows that the hybrid algorithm is much faster than the ordinary MWST algorithm. The hybrid algorithm with "$\delta_2 = 0.001$" is about 25 times faster than the ordinary MWST algorithm. Moreover, the hybrid algorithm with "$\delta_2 = 0.0001$" is over ten times faster than the ordinary MWST algorithm.

The results of prediction performance are shown in Fig. 10. "MWST" in Fig. 10 indi-

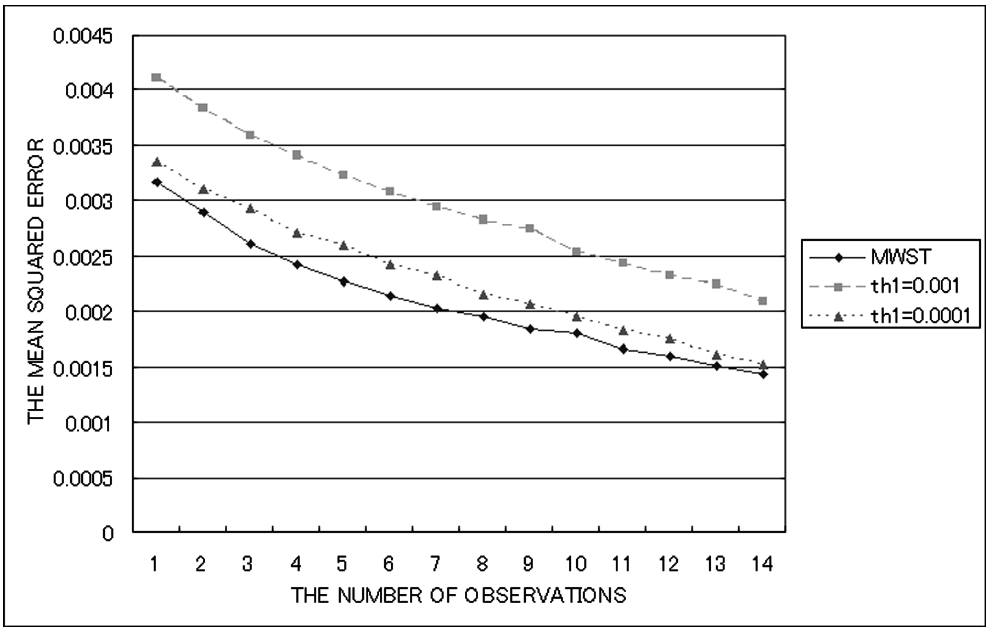


Figure 10: Mean squared error for MWST and hybrid algorithm. Lower scores indicate better performance.

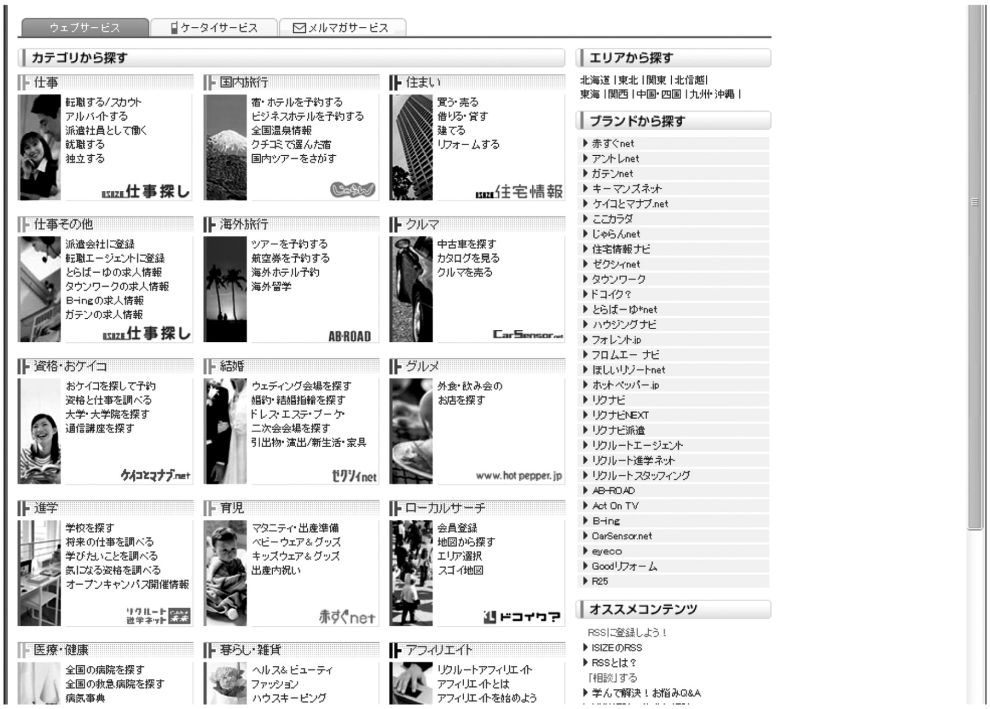
Table 5: Number of arcs in the obtained networks and running time of the learning algorithm.

Algorithm	Number of Arcs	Learning Time (second)
MWST	6041	39489
Hybrid (δ_2=0.001)	315	1522
Hybrid (δ_2=0.0001)	1475	3899

cates the ordinary MWST algorithm, "$\delta_2 = 0.001$" and "$\delta_2 = 0.0001$" indicate the hybrid algorithms with thresholds "$\delta_2 = 0.001$" and "$\delta_2 = 0.0001$", respectively. The horizontal axis is the number of observations, and the vertical axis is the MSE values between the inferred visiting probability of page i given the observations $\mathbf{e}_k^t$ and the actual data x_i^t for the 50 users. The results show that the MSEs of the hybrid algorithm with "$\delta_2 = 0.0001$" are almost the same as those of the ordinary MWST, and the learning time of the hybrid method is reduced by less than 1/10th that of the ordinary MWST (see Table 5). This result shows the validity of the hybrid algorithm for massive datasets. In addition, Fig. 10 and Table 5 also show the trade-off between prediction accuracy and computational cost (learning time). In actual situations, it is important to search for the optimum threshold values to adapt to the required conditions.

6. Web based Reccomender System

We installed the proposed hyblyd algorithm on a web based reccomender system shown in Fig. 11. This system has 60 window flames which are classified by categories. The cat-


ウェブサービス ケータイサービス メルマガサービス

カテゴリから探す

仕事
転職する/スカウト
アルバイトする
派遣社員として働く
就職する
独立する

国内旅行
宿・ホテルを予約する
ビジネスホテルを予約する
全国温泉情報
クチコミで選んだ宿
国内ツアーをさがす

住まい
買う・売る
借りる・貸す
建てる
リフォームする

仕事その他
派遣会社に登録
転職エージェントに登録
とらばーゆの求人情報
タウンワークの求人情報
B-ingの求人情報
ガテンの求人情報

海外旅行
ツアーを予約する
航空券を予約する
海外ホテル予約
海外留学

クルマ
中古車を探す
カタログを見る
クルマを売る

資格・おケイコ
おケイコを探して予約
資格と仕事を調べる
大学・大学院を探す
通信講座を探す

結婚
ウェディング会場を探す
婚約・結婚指輪を探す
ドレス・エステ・ブーケ・
二次会会場を探す
引出物・演出/新生活・家具

グルメ
外食・飲み会の
お店を探す

進学
学校を探す
将来の仕事を調べる
学びたいことを調べる
気になる資格を調べる
オープンキャンパス開催情報

育児
マタニティ・出産準備
ベビーウェア＆グッズ
キッズウェア＆グッズ
出産内祝い

ローカルサーチ
会員登録
地図から探す
エリア選択
スゴイ地図

医療・健康
全国の病院を探す
全国の救急病院を探す
病気事典

暮らし・雑貨
ヘルス＆ビューティ
ファッション
ハウスキーピング

アフィリエイト
リクルートアフィリエイト
アフィリエイトとは
アフィリエイトを始めよう

エリアから探す

北海道 | 東北 | 関東 | 北信越 |
東海 | 関西 | 中国・四国 | 九州・沖縄 |

ブランドから探す

- 赤すぐnet
- アントレnet
- ガテンnet
- キーマンズネット
- ケイコとマナブ.net
- ここカラダ
- じゃらんnet
- 住宅情報ナビ
- ゼクシィnet
- タウンワーク
- ドコイク？
- とらばーゆ*net
- ハウジングナビ
- フォレント.jp
- フロムエー ナビ
- ほしいリゾートnet
- ホットペッパー.jp
- リクナビ
- リクナビNEXT
- リクナビ派遣
- リクルートエージェント
- リクルート進学ネット
- リクルートスタッフィング
- AB-ROAD
- Act On TV
- B-ing
- CarSensor.net
- eyeco
- Goodリフォーム
- R25

オススメコンテンツ

RSSに登録しよう！
- ISIZEのRSS
- RSSとは？

「相談」する
- 学んで解決！お悩みQ&A

Figure 11: Web based reccomender system

egories are 1. Recruiting, 2. Travel, 3. Housing, 4.Cars, 5. Qualifications, 6. Marriages, 7. Restaurants, 8. Education, 9. Healthcare, 10. Medical care, 11. Child-reaing, and so on. There are about 5,000 pages which provide the corresponding commercial information in each category. However, each category window can present only seven web page links. Therefore, the system searchs and presents the page links of which visiting probabilities are in the top seven for each category using a user's browsing history data as observations of Bayesian network. Thus, the system can addaptively construct commercial web pages according to the user's past history data.

7. Conclusion

This paper proposed a collaborative filtering method for massive datasets based on Bayesian networks. We first compared the prediction accuracy of four scoring-based learning Bayesian networks algorithms (AIC, MDL, UPSM, and BDeu) and two CI-based learning Bayesian networks algorithms (MWST and Polytree-MWST) using actual massive datasets. The results showed (1) for large networks, the scoring-based algorithms have lower prediction accuracy than the CI-based algorithms, and (2) when the scoring-based algorithms uses a greedy search to learn a large network, the algorithms that make a lot of arcs tend to have lower prediction accuracy than the algorithms that make fewer arcs. Next, we proposed a learning algorithm based on MWST for collaborative filtering of massive datasets. The proposed algorithm employs a traditional data mining technique, "a priori algorithm", to quickly calculate the amount of mutual information from massive datasets. We compared the original MWST algorithm and the proposed algorithm using actual data, and our comparison showed the effectiveness of the proposed algorithm.

The remaining problems are as follows:

1. There are still doubts as to how the obtained results are affected by the kind of learning algorithms or by the kind of probabilities propagation algorithm (Murphy et al., 1999). If we use the junction tree algorithm (Jensen, 1996) instead of the loopy belief propagation algorithm, (Weiss, 1997) the results might change.
2. It is not clear how the method of determining the node ordering affects the prediction accuracy of the constructed networks.

The future works are to solve the above problems and to conduct experiments with other datasets to obtain results that are more general. We will also try to find a way to determine appropriate threshold values of the proposed algorithm.

References

Agrawal, R., Imielinski, T., and Swami, A. (1993). Mining association rules between sets of items in large databases. In *Proceedings of 1993 ACM SIGMOD International Conference on Management of Data* (pp.207–216).

Agrawal, R. and Srikant, R. (1994). Fast algorithms for mining association rules in large databases. In *Proceedings of 20th International Conference on Very Large Databases* (pp.478–499).

Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19:716–723.

Balabanovic, M. and Shoham, Y. (1997). Fab: Content-based, collaborative recommendation. *Communications of the ACM*, **40**(3), 66–72.

Billsus, D. and Pazzani, M. (1998). Learning collaborative information filters. In *Proc. Int'l Conf. Machine Learning.*

Bouckaert, R. (1994). Probablistic network construction using the minimum description length principle. Technical report ruu-cs-94-27, Utercht University.

Breese, J., Heckerman, D., and Kadie, C. (1998). Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of 14th Conference on Uncertainty in Artificial Intelligence.*

Buntine, W.L. (1991). Theory refinement on Bayesian networks. In *Proceedings of UAI-91.*

Cheng, J. and Greiner, R. (1999). Comparing Bayesian network classifiers. In *Uncertainty in Artificial Inteligence* (pp.101–108).

Chickering, D. (1995). Learning Bayesian networks is np-complete. In *Learning from Data: Artificial Intelligence and Statistics*, volume V (pp.121–130).

Chow, C.K. and Lin, C.N. (1968). Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, IT-14:462–467.

Cooper, G.F. and Herskovits, E. (1991). A Bayesian method for constructing Bayesian belief networks from databases. In *Proceedings of UAI-91.*

Cooper, G.F. and Herskovits, E. (1992). A Bayesian methods for the inducation of probabilistic networks from data. *Machine Learning*, **9**, 309–347.

Delgado, J. and Ishii, N. (1999). Memory-based weighted-majority prediction for recommendar systems. In *Proc. ACM SIGIR '99 Workshop Recommender Systems: Algorithms and*

Evaluation.

Friedman, N., Geiiger, D., and Goldszmidt, M. (1997). Bayesian network classifiers. *Machine Learning*, **29**, 131–161.

Getoor, L. and Sahami, M. (1999). Using probabilistic relational models for collaborative filtering. In *Proc. Workshop Web Usage Analysis and User Profiling (WEBKDD '99).*

Goldberg, K., Roeder, T., Gupta, D., and Perkins, C. (2001). Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval J.*, **4**(2), 133–151.

Greiner, R., Grove, A., and Schuurmans, D. (1997). Learning Bayesian nets that perform well. In *Proceedings of UAI-97.*

Heckerman, D., Geiger, D., and Chickering, D. (1995). Learning Bayesian networks: the combination of knowledge and statistical data. *Machine Learning*, **20**, 197–243.

Heckerman, D., Meek, C., and Cooper, G. (1997). A Bayesian approach to causal discovery. Technical report msr-tr-97-05, Microsoft Research.

Hill, W., Stead, L., Rosenstein, M., and Furnas, G. (1995). Recommending and evaluating choices in a virtual community of use. In *Proceedings of the Conference on Human Factors in Computing Systems.*

Hofmann, T. (2003). Collaborative filtering via Gaussian probabilistic latent semantic analysis. In *Proc. 26th Ann. Int'l ACM SIGIR Conf.*

Jensen, F. (1996). *An Introduction to Bayesian networks.* University College London Press.

Marlin, B. (2003). Modeling user rating profiles for collaborative filtering. In *Proc. 17th Ann. Conf. Neural Information Processing Systems (NIPS '03).*

Murphy, K.P., Weiss, Y., and Jordan, M.I. (1999). Loopy belief propagation for approximate inference: An empirical study. In *Proceedings of Uncertainty in Artificial Intelligence* (pp.467–475).

Nakamura, A. and Abe, N. (1998). Collaborative filtering using weighted majority prediction algorithms. In *Proc. 15th Int'l Conf. Machine Learning.*

Pavlov, D. and Pennock, D. (2002). A maximum entropy approach to collaborative filtering in dynamic, sparse, high-dimensional domains. In *Proc. 16th Ann. Conf. Neural Information Processing Systems (NIP '02).*

Pearl, J. (1988). *Probabilistic Reasoning Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann.

Resnick, P., Iakovou, N., Sushak, M., Bergstrom, P., and Riedl, J. (1994). Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of 1994 Computer Supported Cooperative Work Conference.*

Rissanen, J. (1983). A universal prior for integers and estimation by minimum description length. *Annals of Statistics*, **11**, 416–431.

Shardanand, U. and Maes, P. (1995). Social information filtering: Algorithms for automating 'word of mouth'. In *Proceedings of Conference on Human Factors in Computing Systems.*

Suzuki, J. (1993). A construction of Bayesian networks from databases on mdl principle. In *Proceedings of UAI-93* (pp.266–273).

Ungar, L. and Foster, D. (1998). Clustering methods for collaborative filtering. In *Proc. Recommender Systems, Papers from 1998 Workshop.* Technical Report WS-98-08.

Weiss, Y. (1997). Belief propagation and revision in networks with loops. Technical report: Aim-1616, Massachusetts Institute of Technology.

Yang, S. and Chang, K.-C. (2002). Comparison of score metrics for Bayesian network lerarning. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, **32**(3).

(Received November 20 2007, Revised July 22 2008)