Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

A parallel and constraint induced approach to modeling user preference from rating data

Kun Yue [a], Xinran Wu [a], Liang Duan [a,*], Shaojie Qiao [b], Hao Wu [a]

[a] *School of Information Science and Engineering, Yunnan University, Kunming, China*
[b] *School of Software Engineering, Chengdu University of Information Technology, Chengdu, China*

ARTICLE INFO

Article history:
Received 23 April 2020
Received in revised form 26 June 2020
Accepted 29 June 2020
Available online 2 July 2020

Keywords:
Rating data
User preference
Latent variable
Bayesian network
Expectation maximization
Spark

ABSTRACT

Observed rating data in Web2.0 applications concerns user attributes and rating scores, which explicitly reflects users' overall evaluation on events, products and various informative items. However, the unobservable user preference is critical for personalized services, precise marketing, accurate advertising, etc. In this paper, by adopting Bayesian network (BN) with a latent variable as the knowledge framework to describe user preference using the latent variable, we propose user preference Bayesian network (UPBN) to represent dependence relations among the latent and observed variables. By incorporating the classic expectation maximization (EM) algorithm and scoring & search idea for learning a BN, we focus on UPBN construction from rating data, i.e., the learning of probability parameters and graphical structure. To make UPBN fit the rating data, we first give the constraints of structure and parameters in terms of inherence dependencies among user preference, latent variable and characteristics of EM. Consequently, we present a parallel and constraint induced algorithm for UPBN construction based on EM, structural EM (SEM) and Bayesian information criterion. To deal with the large amount of iterations of probability computations and guarantee the efficiency of model construction, we implement our algorithms upon Spark for the massive intermediate results and large scale rating datasets. Experimental results show the expressiveness of UPBN for preference modeling and the efficiency of model construction, and also demonstrate that UPBN outperforms some state-of-the-art models for user preference estimation and rating prediction.

1. Introduction

More and more user behavioral data are collected, such as rating data in e-commence and social network applications [1]. To make full use of these data for decision support, much attention has been paid to personalized recommendation and accurate user targeting in both academic and industry paradigms [2–4]. Generated in Web2.0 applications, rating data include attributes of concerned items and users, as well as ratings (i.e., scores) that reflect user preferences or interest on items (e.g., events or products) [5,6]. As highly problem-rich areas, preference estimation, rating prediction, and query processing based on rating data analysis, are useful for personalized services [5,7]. For example, MovieLens dataset, provided by GroupLens [8], includes ratings on movies, metadata of movies and demographic properties of users. A user may give a positive rating if he/she prefers some attributes of an item, while the rating reflects this user's preference w.r.t. demographic properties like *age*, *gender* and *occupation*. It is worthwhile to establish a preference model from rating data to facilitate preference estimation, rating prediction or reasoning of inherent dependence relations among concerned attributes.

Unlike the observable attributes of items and users, user preference could not be observed directly. This makes it challenging to construct a model on incomplete data due to the presence of unobservable preference. To this end, we introduce a latent (a.k.a. hidden or unobservable) variable to represent user preference, which depicts the possible preferred item type in rating data. Although the unobservable preference could be trivially measured by the frequency of rating behavior quantitatively, nonlinear dependence relations exist among the latent variable and observed attributes of users/items in general situations. For example, a *male* and *young* user may 65% probably prefer an *action* movie, and a user who prefers a *science fiction* movie may 85% probably rate "*Star Wars*" with 5 stars. Therefore, in this paper, we focus on the establishment of a preference model to represent these dependence relations while guaranteeing the expressiveness of uncertainties and applicability of analysis-centric tasks on rating data.

However, the state-of-the-art methods for learning latent variable models (LVMs) [9] are carried out by learning or tuning

[*] Corresponding author.
E-mail addresses: kyue@ynu.edu.cn (K. Yue), xrwu@mail.ynu.edu.cn (X. Wu), duanl@ynu.edu.cn (L. Duan), sjqiao@cuit.edu.cn (S. Qiao), haowu@ynu.edu.cn (H. Wu).

https://doi.org/10.1016/j.knosys.2020.106206
0950-7051/© 2020 Elsevier B.V. All rights reserved.

parameters upon the given structure according to the domain-specific sense of concerned variables [10,11]. By these methods, any form of dependence relations could not be represented, since the model is established on the fixed structure. Some other methods, such as matrix factorization [4,5] and deep neural network [6,12], behave well in rating prediction, while they are poorly interpretable due to the non-intuitive description of user preference. Therefore, we focus on developing a interpretable and reasoning-facilitated preference model by learning both the structure and parameters from rating data.

As an important and popular probabilistic graphical model (PGM), Bayesian network (BN) is a well adopted framework for representing and reasoning uncertain knowledge [13,14]. A BN is a directed acyclic graph (DAG), where nodes represent random variables and edges describe dependence relations among these variables [15]. Each variable in a BN is associated with a conditional probability table (CPT) including a set of conditional probabilities given the states of its parent nodes. BN has been widely used in user preference modeling [14], medical decision support [13], learning of gene regulatory networks with time-delayed regulations [16], quality control [17], etc. Thus, in this paper, we adopt the BN with a latent variable as the framework of user preference model, abbreviated as UPBN, such that any form of dependence relations among the observed and latent variable could be represented qualitatively and quantitatively. By constructing an UPBN, user preference could be estimated according to the probabilistic reasoning w.r.t. the latent variable. For example, if a *29-year-old male reporter* often buys the "*science*" books like "*Foundation of Data Science*" and gives them high rating scores, his preference could be estimated according to $P(genre\_preference = \text{``}science\text{''} | sex = \text{``}M\text{''}, age = 29)$.

Straightforwardly, the classic expectation maximization (EM) algorithm [18,19] could be adopted to learn parameters in the presence of latent variable, but sensitive to the initial value, since different initializations may lead to the results with great difference [20]. The more the missing data or latent variables, the more significant the difference will exist [20]. In view of the characteristics of rating data and the introduced latent variable to represent user preference (i.e., preferred item type), we discuss the property to guarantee that an UPBN could fit the given rating data theoretically by considering the rationale of EM when used for parameter learning of a local structure relevant to the latent variable. Then, we give two kinds of constraints on the initial DAG and CPTs respectively, which avoid the random result and prune a large part of the search space in UPBN construction.

Structural EM (SEM) algorithm [9,21], as the improvement of the classic EM, was proposed to learn the structure of a LVM by using the idea of scoring and hill-climbing search. During the execution of SEM, EM is adopted for parameter learning by a series of iterations, in each of which there is large scale probability computations and intermediate results. The more the parent nodes of the latent variable, or the cardinalities of the latent variable or possible combinations of parent states, the more time consuming for model learning. To overcome the efficiency bottleneck, we incorporate the constraints of DAG and CPTs into SEM, and propose a constraint induced algorithm for UPBN construction, in which Bayesian information criterion (BIC) [22] is adopted as the scoring metric to measure whether a candidate model fit the given dataset or not.

Aiming at the large scale intensive iterations in parameter learning and maximum likelihood estimation during structure learning w.r.t. the large scale intermediate results, we adopt Spark [23] as the distributed computing framework to implement our proposed algorithms. From the algorithm implementation point of view, we present the Spark based method for UPBN construction, in which the operations on the resilient distributed dataset (RDD) are emphasized.

Based on the UPBN constructed by our proposed method, any form of dependence relations implied in rating data could be described qualitatively and quantitatively. Thus, user preferences could be estimated and ratings could be predicted by using probabilistic reasonings [15]. It is worth noting that the latent variable makes it possible to estimate user preferences from the observed values by conditional probabilities and infer other information w.r.t. the given preference by posterior probabilities, which could be fulfilled by the algorithm of probabilistic reasoning with a BN [24,25]. Moreover, the latent variable makes the dependence relations more simplified and interpretable.

Generally, the main contributions of this paper are as follows:

- We use a latent variable to describe user preference in rating data and propose UPBN as the framework of representation and reasoning of dependence relations among the latent and observed variables with uncertainties.
- We give a property that the UPBN should satisfy to theoretically guarantee the fitness of UPBN to rating data when EM algorithm is incorporated into parameter learning in presence of incomplete data w.r.t. the latent variable. Then, we give the constraints and initial structure and parameters to avoid the randomness of model representation and prune the search space in model construction.
- We propose the Spark based and constraint induced algorithms for parallel learning of UPBN, where the operations on RDD are presented. Then, the efficiency of iterative probability computations on large scale intermediate results could be improved.
- We conduct extensive experiments on synthetic datasets and the MovieLens datasets [8] upon Spark clusters to test the efficiency and effectiveness of our proposed method. Experimental results show that our method is efficient for UPBM construction, and also effective in preference estimation and rating prediction compared with some state-of-the-art models.

The remainder of this paper is organized as follows: Section 2 introduces related work. Section 3 defines UPBN and gives the property and constraints of UPBN. Section 4 gives the Spark based and constraint induced algorithms for UPBN construction. Section 5 presents experimental results and performance studies. Section 6 concludes and discusses future work.

2. Related work

2.1. Preference modeling

As a type of representative methods, PGM has been widely used in realistic preference modeling, including BN, topic model, etc. BN is an effective framework to facilitate the representation and reasoning of user preferences by any form of dependence relations among relevant attributes [14,26]. However, unobservable preferences could not be described directly by the BN without latent variables. Topic model, such as probabilistic latent semantic analysis (pLSA) [27] and latent dirichlet allocation (LDA) [28], could be intuitively used to describe the user preferences by latent variables [10], but any form of dependence relations with uncertainties cannot be represented objectively due to the fixed structure.

Different from these methods, we previously [29] presented the preparatory approach for modeling user preference based on BN with a latent variable, but the efficiency still cannot be guaranteed and the effective constraints have not been given. In this paper, we focus on the properties of UPBN to guarantee the fitness of UPBN to rating data theoretically, followed by the design of algorithms for efficient learning of UPBN.

2.2. Rating data analysis

Matrix factorization and deep neural network are two types of classic methods for rating data analysis. In the former kind of methods, Koren [30,31] proposed SVD++ and timeSVD++ to model static and dynamic user preferences by extending the traditional singular value decomposition (SVD). Salakhutdinov and Mnih [32] proposed the probabilistic matrix factorization (PMF), and Tan et al. [33] gave the multi-attribute PMF model for personalized recommendation. Wang et al. [34] proposed Bayesian probabilistic multi-topic matrix factorization (BPMMF) for rating prediction. In the latter kind of methods, convolutional neural network [35] and recurrent neural network [36] were adopted to predict user response respectively. Qu et al. [6,37] proposed a deep neural network model with a novel architecture, namely product-network in network, to predict user response. Guo et al. [12] proposed an attentive long short-term preference model for personalized product search via the attention mechanism. In addition, graph neural network [38] and autoencoder neural network [39] are also used for recommendation systems.

The above methods behave well in rating prediction, but could not be well interpreted due to the nonintuitive description of user preference. In addition, the deep neural network focuses more on parameter training upon the relatively standard model structure, but cannot represent any form of dependence relationships among the attributes in rating data effectively. In particular, we concentrate more on structure construction of UPBN, taking parameter learning as the preliminary, which is consistent with the principle of general BN construction [15].

2.3. Learning of BN with latent variables

BN construction includes DAG construction and CPT learning (i.e, structure construction and parameter learning). Scoring & search based [15] and clique based [40] methods are two types of methods to construct the DAG structure with latent variables. Friedman [9] proposed the method for parameter learning of BN with latent variables based on the EM algorithm [18,19], and extended the classic BIC metric to make the general scoring & search based method suit to the situation with latent variables. Friedman [21] further proposed the structural EM (SEM) algorithm to decrease the complexity of the classic scoring & search based method. Elidan et al. [40] improved the maximal semi-clique (MSC) based idea and proposed the method to detect latent variables by semi-cliques and evaluating each candidate structure to find the best network. He et al. [41] proposed the method to detect latent variables based on ϵ-cliques and determined the number of latent variables by adjusting the value of ϵ.

The former type of methods could be used to represent the dependence relations among the latent and observed variables directly from data, but the efficiency of the construction process and the correctness of the result cannot be guaranteed [20]. The latter type could make general BNs derived efficiently by incorporating latent variables, but the practical sense of latent variables could not be interpreted intuitively. Thus, it is still worthwhile to discuss the efficient and effective method for UPBN construction.

2.4. Parallel learning of BN

Parallel learning of BN usually consists of parallel learning of parameters and parallel construction of the structure. As for the situation with large scale incomplete data, Zhao et al. [42] proposed a parallel algorithm upon MapReduce [43] for parameter learning based on the factor graph. Confronted with the massive, distributed and dynamically changing data, we [44] proposed a parallel and incremental approach for learning BNs by extending the classic scoring & search algorithm based on MapReduce. Gao and Wei [45] provided a parallel structure learning algorithm from multiple local subgraphs to global structure.

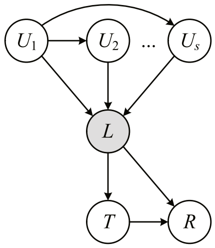


Fig. 1. An UPBN ignoring CPTs.

In these methods for the situation with incomplete data, a large amount of iterations have to be done before convergence to achieve the effective results. However, some MapReduce based techniques are not exactly suitable for intensive iterations, in each of which intermediate results will be materialized to disk. Thus, we adopt Spark [23] as the distributed in-memory computing framework to efficiently fulfill the intensive iterations in UPBN construction.

3. Properties and constraints of UPBN

In this section, we first present the definition of UPBN. Then, we give the properties, by which the initial DAG and CPTs are presented to ensure the effectiveness of UPBN for preference modeling.

3.1. Concept of UPBN

Let $\mathcal{O}$ be the set of observed variables. Let L be the latent (i.e., unobservable) variable to represent the possible preferred item type in rating data. Let $\mathcal{V} = \{V_1, V_2, \ldots, V_n\}$ be the set of all variables, where $\mathcal{V} = \mathcal{O} \cup \{L\}$ and $|\mathcal{V}| = n$. Let T and R be the variables to depict the attributes of items and user ratings respectively. Let $\mathcal{U} = \{U_1, U_2, \ldots, U_s\}$ be the set of variables to describe user's demographic properties, where $\mathcal{O} = \mathcal{U} \cup \{T, R\}$. Following, we give the definition of UPBN.

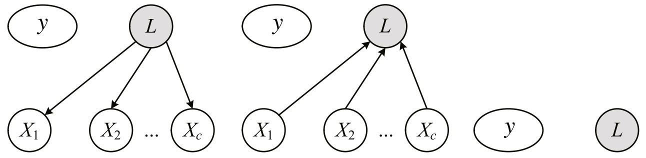
Definition 1. A UPBN is a BN with a latent variable, denoted as (G, θ), where

(1) $G = (\mathcal{V}, E)$ is a DAG, where $\mathcal{V} = \mathcal{U} \cup \{L, T, R\}$ is the set of variables, and E is the set of directed edges representing dependence relations among the variables in $\mathcal{V}$. Specifically, a directed edge $\langle V_a, V_b \rangle$ $(V_a, V_b \in \mathcal{V}, a \neq b)$ in E indicates the dependence relation from V_a to V_b.

(2) θ is the set of probability parameters consisting of the CPTs of all variables, where

- $\pi(V_i)$ denotes the set of parent nodes of V_i $(V_i \in \mathcal{V})$.
- $\theta_{ijk} = P(V_i = k|\pi(V_i) = j)$ denotes the parameter of V_i when the value of V_i is k and the values of its parent nodes take the jth combination.

An UPBN without CPTs is shown as Fig. 1. To construct the UPBN from rating data, the classic EM algorithm is adopted to learn parameters in the presence of latent variable, in which we focus on the fitness of UPBN to the rating data, and further discuss the constraints of parameters and structure of the UPBN.



(a) Local structure 1 (b) Local structure 2 (c) Local structure 3

Fig. 2. Possible local structures w.r.t. the latent variable.

Table 1
Notations.

Notation	Description
D	The training dataset
$\mathcal{X} = x_i$	The ith combination of values of $\mathcal{X}$ $(1 \leq i \leq N)$
$\mathcal{Y} = y_j$	The jth combination of values of $\mathcal{Y}$ $(1 \leq j \leq M)$
$n(x_i)$	Number of samples in D satisfying $\mathcal{X} = x_i$
$n(x_i, y_j)$	Number of samples in D satisfying $\mathcal{X} = x_i$ and $\mathcal{Y} = y_j$
C	Cardinality of L
$L = l_q$	The qth value of L $(1 \leq q \leq C)$
θ^t	Set of current parameters

3.2. Fitness of parameters of the latent variable

Let $\mathcal{X} = \{X_1, X_2, \ldots, X_c\}$ be the set of observed variables with dependence relations to the latent variable L, and $\mathcal{Y}$ be the set of observed variables without dependence relations to L. Without loss of generality, there may exist three possible forms of local structures w.r.t. L in UPBN, shown in Fig. 2, where the dependence relations among observed variables are ignored.

Parameters of UPBN cannot fit the given dataset if the CPTs of L have no changes by EM iterations. This means that the results of parameter learning will not fit the rating data if the CPTs w.r.t. L are kept the same as the initial values in the inferior initial DAG. Theorem 1 guarantees that the UPBN can fit the given rating data based on the initial DAG theoretically.

Theorem 1. *CPTs corresponding to the latent variable L can fit the given rating data by EM iterations if and only if there is at least one directed edge from L to an observed variable in $\mathcal{O}$.*

Proof. First, we give the notations in Table 1.

The maximum likelihood of parameters calculated by EM is represented as

$$\theta^{t+1} = \arg\sup_{\theta} \sum_{q=1}^{C} P(L = l_q | D, \theta^t) \log P(D, L = l_q | \theta) \tag{1}$$

Upon the local structure in Fig. 2(a), we show the process of EM iterations.

E-step: Based on the set of current parameters θ^t, the posterior probability $P(L|\mathcal{X}, \theta^t)$ and expected log-likelihood are obtained. Specifically, we have

$$\begin{aligned} & P(L = l_q | \mathcal{X} = x_i, \mathcal{Y} = y_j, \theta^t) \\ =& P(L = l_q | \mathcal{X} = x_i, \theta^t) \\ =& \frac{P(L = l_q, \theta^t) P(\mathcal{X} = x_i | L = l_q, \theta^t)}{P(\mathcal{X} = x_i, \theta^t)} \\ =& \frac{P(L = l_q, \theta^t) P(\mathcal{X} = x_i | L = l_q, \theta^t)}{\sum_{q=1}^{C} P(L = l_q, \theta^t) P(\mathcal{X} = x_i | L = l_q, \theta^t)} \end{aligned} \tag{2}$$

Meanwhile, we have the following log-likelihood

$$l(\theta|D) = \sum_{i=1}^{N} \sum_{j=1}^{M} n(x_i, y_j) logP(\mathcal{X} = x_i, \mathcal{Y} = y_j) \tag{3}$$

Then, the expected log-likelihood can be obtained from the posterior probability in Eq. (2) and log-likelihood in Eq. (3) as follows:

$$\begin{aligned} & E\left(l(\theta|D)\right) \\ =& \sum_{i=1}^{N} \sum_{j=1}^{M} n(x_i, y_j) \sum_{q=1}^{C} [P(L = l_q | \mathcal{X} = x_i, \mathcal{Y} = y_j, \theta^t) \\ & \log P(\mathcal{X} = x_i, \mathcal{Y} = y_j, L = l_q)] \end{aligned} \tag{4}$$

Since $P(\mathcal{X} = x_i, \mathcal{Y} = y_j, L = l_q) = P(\mathcal{X} = x_i | L = l_q) P(L = l_q) P(\mathcal{Y} = y_j)$, Eq. (4) can be reduced to

$$\begin{aligned} & E\left(l(\theta|D)\right) \\ =& \sum_{i=1}^{N} \sum_{j=1}^{M} n(x_i, y_j) \sum_{q=1}^{C} [P(L = l_q | \mathcal{X} = x_i, \mathcal{Y} = y_j, \theta^t) \\ & \log P(\mathcal{X} = x_i | L = l_q)] \\ +& \sum_{i=1}^{N} \sum_{j=1}^{M} n(x_i, y_j) \sum_{q=1}^{C} [P(L = l_q | \mathcal{X} = x_i, \mathcal{Y} = y_j, \theta^t) \\ & \log P(L = l_q)] \\ +& \sum_{i=1}^{N} \sum_{j=1}^{M} n(x_i, y_j) \sum_{q=1}^{C} [P(L = l_q | \mathcal{X} = x_i, \mathcal{Y} = y_j, \theta^t) \\ & \log P(\mathcal{Y} = y_j)] \end{aligned} \tag{5}$$

M-step: The expected log-likelihood is maximized and the parameters are updated as θ^{t+1} including $P(\mathcal{X} = x_i | L = l_q)$, $P(L = l_q)$ and $P(\mathcal{Y} = y_j)$. The task in M-step could be reduced to obtain the extreme value of Eq. (5) w.r.t. the following constraints:

$$\sum_{i=1}^{N} P(\mathcal{X} = x_i | L = l_q) = 1$$

$$\sum_{q=1}^{C} P(L = l_q) = 1$$

$$\sum_{j=1}^{M} P(\mathcal{Y} = y_j) = 1$$

Correspondingly, we adopt the method of Lagrange multiplier by constructing the Lagrange function (LF) as follows:

$$\begin{aligned} & E\left(l(\theta|D)\right) + \sum_{q=1}^{C} \gamma_q \left[1 - \sum_{i=1}^{N} P(\mathcal{X} = x_i | L = l_q)\right] \\ +& \mu \left[1 - \sum_{q=1}^{C} P(L = l_q)\right] + \delta \left[1 - \sum_{j=1}^{M} P(\mathcal{Y} = y_j)\right] \end{aligned} \tag{6}$$

where γ_q, μ and δ are Lagrange multipliers.

Then, we obtain the first-order partial derivatives of LF w.r.t. $P(\mathcal{X} = x_i | L = l_q)$, $P(L = l_q)$ and $P(\mathcal{Y} = y_j)$ respectively.

$$\frac{\sum_{j=1}^{M} n(x_i, y_j) P(L = l_q | \mathcal{X} = x_i, \mathcal{Y} = y_j, \theta^t)}{P(\mathcal{X} = x_i | L = l_q)} - \gamma_q = 0 \tag{7}$$

$$\frac{\sum_{i=1}^{N} \sum_{j=1}^{M} n(x_i, y_j) P(L = l_q | \mathcal{X} = x_i, \mathcal{Y} = y_j, \theta^t)}{P(L = l_q)} - \mu = 0 \tag{8}$$

$$\frac{\sum_{i=1}^{N} n(x_i, y_j)}{P(\mathcal{Y} = y_j)} - \delta = 0 \tag{9}$$

By Eq. (7) and the constraint $\sum_{i=1}^{N} P(\mathcal{X}=x_i|L=l_q)=1$, we have

$$P(\mathcal{X}=x_i|L=l_q) = \frac{\sum_{j=1}^{M} n(x_i,y_j)P(L=l_q|\mathcal{X}=x_i,\mathcal{Y}=y_j,\theta^t)}{\sum_{i=1}^{N}\sum_{j=1}^{M} n(x_i,y_j)P(L=l_q|\mathcal{X}=x_i,\mathcal{Y}=y_j,\theta^t)} \tag{10}$$

By Eq. (8) and the constraint $\sum_{q=1}^{C} P(L=l_q)=1$, we have

$$P(L=l_q) = \frac{\sum_{i=1}^{N}\sum_{j=1}^{M} n(x_i,y_j)P(L=l_q|\mathcal{X}=x_i,\mathcal{Y}=y_j,\theta^t)}{\sum_{i=1}^{N}\sum_{j=1}^{M} n(x_i,y_j)} \tag{11}$$

By Eq. (9) and the constraint $\sum_{j=1}^{M} P(\mathcal{Y}=y_j)=1$, we have

$$P(\mathcal{Y}=y_j) = \frac{\sum_{i=1}^{N} n(x_i,y_j)}{\sum_{i=1}^{N}\sum_{j=1}^{M} n(x_i,y_j)} \tag{12}$$

Thus, we obtain θ^{t+1}. From Eqs. (10) and (11), we note that the parameters are dependent of the statistics $n(x_i,y_j)$ of D, which means that the CPTs of L can fit D in the process of EM iterations.

Similar to the above derivation upon the local structure in Fig. 2(a), the parameters $P(\mathcal{X}=x_i)$, $P(\mathcal{Y}=y_j)$ and $P(L=l_q|\mathcal{X}=x_i,\mathcal{Y}=y_j)$ upon the local structure in Fig. 2(b) can be obtained. The CPTs of L, such as $P(L=l_q|\mathcal{X}=x_i,\mathcal{Y}=y_j)$, do not include the statistics $n(x_i,y_j)$ from D. Upon the local structure in Fig. 2(c), the CPTs of L, such as $P(L=l_q)$, do not include the statistics $n(x_i,y_j)$ from D as well. ■

Theorem 1 guarantees that an UPBN can fit the given rating data, since the CPTs w.r.t. the latent variable are changed in EM iterations. For instance, the CPTs of L in the local structure in Fig. 2(a) can be changed in EM iterations, but those in Figs. 2(b) and 2(c) cannot.

3.3. Constraints

We know from Theorem 1 that a poor UPBN may be generated if an inferior initial DAG is adopted when using the scoring & search method [9,15], since the UPBN cannot fit the given rating data without an initial dependence relation from the latent variable to an observed variable. We suppose that a user only rates the preferred items and gives them high rating levels. Based on the sense of variables in $\mathcal{V}$ and Theorem 1, we give the constraints of DAG and CPTs respectively.

As for the DAG of an UPBN, Constraints 1 and 2 are presented to indicate the directed edges in the initial DAG and those not in the UPBN, such that the impossible intermediates could be eliminated.

Constraint 1. *The initial DAG of an UPBN includes $\langle L,T\rangle$, $\langle L,R\rangle$ and $\langle T,R\rangle$, indicating that the frequency and level of ratings are influenced by user preference.*

Constraint 2. *The DAG of an UPBN cannot include $\langle L,U_a\rangle$ $(U_a\in\mathcal{U})$, indicating that user's demographic properties should not be influenced by user preference.*

The initial CPTs are presented based on the initial DAG satisfying Constraint 1, in which the conditional probabilities w.r.t. the frequency and level of ratings are assigned larger values if the item's attribute is consistent with user preference, and the CPTs of other variables are assigned by their mean values.

Constraint 3. *The initial CPTs of an UPBN should satisfy the following conditions:*

(1) A user rates the preferred items more frequently, formalized as

$$P(T=t|L=l,t=l) > P(T=t|L=l,t\neq l)$$

(2) High rating level is likely to be given when the item's attribute is consistent with user preference, formalized as

$$P(R=r_1|T=t,L=l,t=l) > P(R=r_2|T=t,L=l,t=l)$$
$$P(R=r_1|T=t,L=l,t\neq l) < P(R=r_2|T=t,L=l,t\neq l)$$

where $r_1>r_2$.

(3) For the node V_i in $\mathcal{U}\cup\{L\}$, the initial conditional probability of $P(V_i=k|\pi(V_i)=j)$ is $\frac{1}{|V_i|}$, where $|V_i|$ is the cardinality of V_i.

4. Constraint induced UPBN construction

In this section, we first present the constraint induced algorithms for learning the DAG and CPTs of an UPBN by focusing on the Spark based implementation of iterative computations. Then, we give the constraint induced algorithm for UPBN construction accordingly.

4.1. Parameter learning

By the EM algorithm upon the currently constructed structure, parameters could be obtained until convergence or reaching the maximal number of iterations. We first denote the UPBN constructed by h times of iterations as $(G^h,\theta^{h,0})$. By t times of parameter learning upon $(G^h,\theta^{h,0})$, the currently learned UPBN is denoted as $(G^h,\theta^{h,t})$.

An iteration of the EM algorithm consists of the following three parts. The weighted dataset is generated in parallel and the conditional probability of L is adopted as the weight of each complete sample. Next, the expected statistics are obtained by counting the weighted dataset in the E-step. Finally, the maximum likelihood of parameters is calculated from the expected statistics in the M-step. To make efficient iterative computations by the EM algorithm, the process for parameter learning of UPBN is as follows.

- **Generation of the weighted dataset:**

(1) The whole dataset D consisting of the values of variables in $\mathcal{V}$ except L is converted to a RDD, such that each *Worker* in the Spark cluster keeps a partition of D.

(2) The ***map*** function is employed to parallelly generate the complete samples for each sample in D, in which a complete sample $(D_d, L=l_q)$ can be obtained from D_d, where D_d $(1\leq d\leq |D|)$ is the dth sample in D, l_q $(1\leq q\leq C)$ is the qth value of L, and C is the cardinality of L. Following, the ***flatMap*** function is employed to make each record correspond to one complete sample. In the first pass of parameter learning, the ***persist*** function is implemented to store the complete dataset in memory to avoid repeated generation.

(3) The ***map*** function is employed to generate the weighted dataset $D^{h,t}$, in which the conditional probability of L, denoted as $P(L=l_q|D_d,\theta^{h,t})=\frac{P(L=l_q,D_d|\theta^{h,t})}{\sum_{q=1}^{C}P(L=l_q,D_d|\theta^{h,t})}$, could be calculated in parallel and adopted as the corresponding weight for each $(D_d,L=l_q)$.

(4) $D^{h,t}$ is persisted in memory for the following structure construction by using the ***persist*** function, in which each record in $D^{h,t}$ corresponds to a weighted sample.

- **E-step:**

(1) According to Eq. (13), the ***map*** function is employed to parallelly generate the labels $i-j-k$ of the expected statistics for each weighted sample in $D^{h,t}$.

(2) $\langle$key, value$\rangle$ pairs are generated parallelly by the ***map*** and ***flatMap*** function, in which each record is $\langle i-j-k, P(L = l_q|D_d, \theta^{h,t})\rangle$.

(3) According to Eq. (14), the ***reduceByKey*** function is employed to generate a set of expected statistics $m^{h,t}$, in which $m^{h,t}_{ijk}$ is the sum of weights of all weighted samples in $D^{h,t}$ satisfying $V_i = k$ and $\pi(V_i) = j$.

(4) $m^{h,t}$ is collected into *Driver* of the Spark cluster by the ***collect*** function to calculate the maximum likelihood of parameters.

$$F(i,j,k;D_d,L=l_q)=\begin{cases}1, & \text{if } V_i = k \text{ and } \pi(V_i)=j\\ 0, & \text{otherwise}\end{cases} \tag{13}$$

$$m^{h,t}_{ijk}=\sum_{d=1}^{|D|}\sum_{q=1}^{C}P(L=l_q|D_d,\theta^{h,t})F(i,j,k;D_d,L=l_q) \tag{14}$$

• M-step:

The maximum likelihood of parameters, denoted as $\theta^{h,t+1}$, could be calculated by Eq. (15).

$$\theta^{h,t+1}_{ijk}=\begin{cases} m^{h,t}_{ijk}\Big/\sum\limits_{k=1}^{|V_i|}m^{h,t}_{ijk}, & \text{if } \sum\limits_{k=1}^{|V_i|}m^{h,t}_{ijk}>0\\ 1/|V_i|, & \text{if } \sum\limits_{k=1}^{|V_i|}m^{h,t}_{ijk}=0\end{cases} \tag{15}$$

where $|V_i|$ is the cardinality of V_i.

Note that the conditional probability is computed independently for each sample in D, and the expected statistics are also computed independently for each weighted sample in $D^{h,t}$. In the process of parameter learning by EM iterations upon Spark, computations for generating the weighted dataset and those in the E-step could be distributed to *Workers* for parallel operations. The computations in the M-step are fulfilled by centralized processing on *Driver*.

To guarantee the efficient convergence, we further define the difference degree of parameters based on two adjacent EM iterations.

Definition 2. The difference degree between $\theta^{h,t}$ and $\theta^{h,t+1}$ is formalized as

$$\begin{aligned} &diff(\theta^{h,t},\theta^{h,t+1})\\ &\qquad = |logP(D|G^h,\theta^{h,t}) - logP(D|G^h,\theta^{h,t+1})| \end{aligned}$$

where the log-likelihood function $logP(D|G^h, \theta^{h,t})$ describes the consistency between D and $(G^h, \theta^{h,t})$.

For fast convergence to reduce the number of iterations, the EM iterations are repeated until $diff(\theta^{h,t}, \theta^{h,t+1})$ is less than a threshold δ ($\delta > 0$).

The complete procedure of parameter learning is summarized in Algorithm 1. It is clear that the complexity of parameter learning is $O(N_1|D|C)$ if the size of CPTs is a constant, where N_1 ($N_1 > 0$) is the maximal number of iterations in parameter learning. Suppose that the Spark cluster consist of two *Workers*, $D = \{D_1, D_2\}$ and all variables are binary variables, an example of Spark based implementation of parameter learning is illustrated as Fig. 3.

4.2. Structure construction

Using the same strategy of SEM [9], the currently optimal model could be selected for the next iteration among a series of candidate models. We suppose that the current UPBN is $(G^h, \theta^{h,t})$ obtained by h times of iterative structure construction and additional t times of parameter learning. In the $t+1$-th pass of structure construction, the currently optimal model and the set of candidate models are denoted as $(G^{h+1}, \theta^{h+1,0})$ and $\mathcal{G}^h$ respectively.

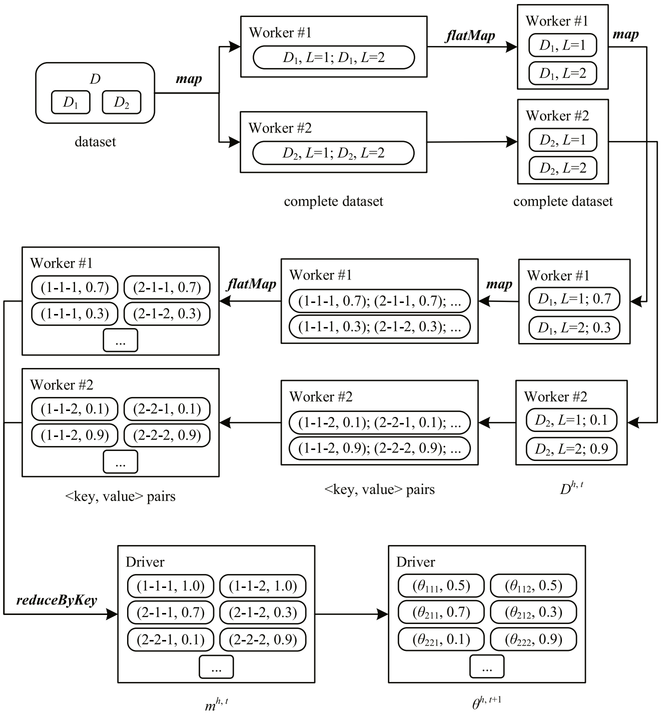


Fig. 3. An example of parameter learning based on Spark.

To make efficient iterative computations, the parallel method for structure construction of UPBN based on Spark is presented as follows.

• Generation of candidate models:

A series of candidate models $\mathcal{G}^h$ are generated for every node by modifying the local structure of the current model with three operators: edge addition, deletion and reversal, in which we only retain the candidate models satisfying Constraints 1 and 2.

• Scoring based evaluation of candidate models:

With the identical calculations of E-step and M-step in parameter learning, the maximum likelihood of parameters of each candidate model in $\mathcal{G}^h$ can be estimated from $D^{h,t}$ that has been persisted in memory. To select the currently optimal model w.r.t. the dataset D, the BIC scoring metric is incorporated to evaluate each candidate model (G^*, θ^*), given in Eq. (16).

$$BIC(G^*,\theta^*|D)=\log P(D|G^*,\theta^*)-\frac{\log|D|}{2}\sum_{i=1}^{n}q_i(|V_i|-1) \tag{16}$$

where $\log P(D|G^*, \theta^*)$ is the log-likelihood to measure whether (G^*, θ^*) fit D. The latter term is the penalty term to avoid overfitting, in which $\sum_{i=1}^{n} q_i(|V_i| - 1)$ is used to measure the size of parameters in G^*, q_i is the number of combinations of values w.r.t. the parent nodes of V_i, $|V_i|$ is the cardinality of V_i, and n is the number of all variables.

• Selection of the currently optimal model:

The candidate model with the highest score is selected as the currently optimal one by

$$(G^{h+1},\theta^{h+1,0})=\arg\max_{(G^*,\theta^*)\in\mathcal{G}^h} BIC(G^*,\theta^*|D) \tag{17}$$

Then, the selected model will be adopted as the current model for the next iteration if the score of the currently optimal model is higher than that of the current model (i.e., $BIC(G^{h+1}, \theta^{h+1,0}|D) > BIC(G^h, \theta^{h,t}|D)$).

Algorithm 1 Parameter learning

```
Input:
    $D$, the training rating dataset
    $(G^h, \theta^{h,0})$, the current UPBN
    $N_1$, the maximal number of iterations
    $\delta$, the threshold
Output:
    $(G^h, \theta^{h,t+1})$, the UPBN
Steps:
if $h = 0$ then
    completeDataset ← D.map{ generating the complete samples for each sample }
    completeDataset.flatMap // making each record correspond to one complete sample
    completeDataset.persist // storing the complete dataset in memory
end if
$t \leftarrow 0$
while true do
    $D^{h,t}$ ← completeDataset.map{ calculating the weight for each complete sample }
    $D^{h,t}$.persist // storing the weighted dataset in memory
    E-step:
    labelDataset ← $D^{h,t}$.map{ generating the labels of expected statistics for each weighted sample }
    keyValuePairs ← labelDataset.map{ generating the pair consisting of the label and weight for each label }
    keyValuePairs.flatMap // making each record correspond to one ⟨key, value⟩ pair
    $m^{h,t}$ ← keyValuePairs.reduceByKey // by Eq. (14)
    $m^{h,t}$.collection
    M-step: calculating $\theta^{h,t+1}$ by $m^{h,t}$ and Eq. (15)
    if $(diff(\theta^{h,t}, \theta^{h,t+1}) < \delta)$ or $(t + 1 = N_1)$ then
        return $(G^h, \theta^{h,t+1})$
    else
        $t \leftarrow t + 1$
    end if
end while
```

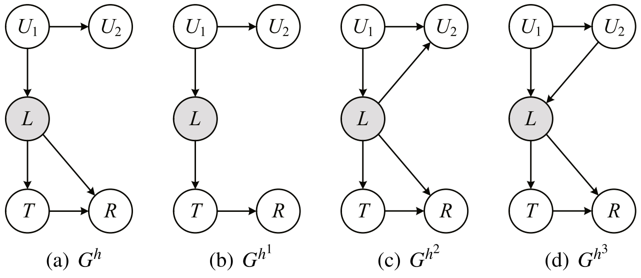
A large amount of candidate models will be discarded while their scores will not be calculated if the constraints are not satisfied. Suppose N_2 is the number of candidate models, it can be seen that the complexity of structure construction is $O(N_2|D|C)$, where C is the cardinality of the latent variable. For instance, the current model is shown in Fig. 4(a), and the candidate models (i.e., G^{h^1}, G^{h^2} and G^{h^3}) are shown in Figs. 4(b)–4(d) respectively. G^{h^1} does not satisfy Constraint 1 and G^{h^2} does not satisfy Constraint 2, since the directed edge $\langle L, R\rangle$ is not included in G^{h^1} and the unreasonable edge $\langle L, U_2\rangle$ is included in G^{h^2}. Thus, it is only necessary to calculate the BIC score of G^{h^3}.

4.3. Constraint induced UPBN construction

To learn an UPBN for preference modeling, we establish our method by incorporating the proposed Spark based algorithms for parameter learning and structure construction. The idea of UPBN construction is summarized as follows:

(1) The initial UPBN is generated based on the constraints given in Section 3, where the initial DAG satisfies Constraint 1 and the initial CPTs are generated randomly satisfying Constraint 3.

(2) The parameters of UPBN are iteratively learned and optimized by Algorithm 1.



(a) G^h (b) G^{h^1} (c) G^{h^2} (d) G^{h^3}

Fig. 4. Current model and candidate models of UPBN.

(3) By the idea of structure construction given in Section 4.2, the candidate model with the highest score is selected as the currently optimal one, and then adopted as the current model in the next iteration.

The above steps (2)-(3) are repeated until the BIC score of the currently optimal model is not greater than that of the current model or the number of iterations achieves the given upper bound. Following, we give the constraint induced and Spark based parallel algorithm for UPBN construction, summarized in Algorithm 2 presenting an iterative process, in which parameter learning and structure construction are implemented in each iteration. Thus, the complexity of Algorithm 2 is $O(N_3 \times (N_1|D|C + N_2|D|C))$, where N_3 ($N_3 > 0$) is the maximal number of iterations. Further, the complexity can be simplified to $O(|D|C)$ if the number of candidate models satisfying constraints (i.e., N_2) is regarded as a constant.

Then, by the probabilistic reasoning with a BN, such as variable elimination [25] and clique tree propagation [24], we can fulfill preference estimation and rating prediction based on the constructed UPBN. Revisiting the UPBN fragment in Fig. 1, the UPBN based results of preference estimation and rating prediction are as follows.

- For preference estimation, we first obtain the set of probabilities $\{P(L = l_1|\mathcal{U}), \ldots, P(L = l_C|\mathcal{U})\}$ by probabilistic reasoning with the UPBN. Then, l_q $(1 \le q \le C)$ is regarded as the preferred item type given the user's demographic properties, according to descending order of the probabilities.
- For rating prediction, we adopt $\arg\max_r P(R = r|\mathcal{U}, T)$ as the predicted rating given the item type and user's demographic properties.

5. Experimental studies

We present an extensive experimental study of our proposed method of preference modeling. By using the synthetic and MovieLens datasets, we conducted two sets of experiments to evaluate the effectiveness and efficiency of our algorithms for UPBN construction.

5.1. Experimental settings

5.1.1. Datasets

- **MovieLens-1M** [8] is a movie rating dataset, including 6040 users, 3952 movies and 1,000,209 ratings records, where each rating record consists of five attributes: *gender*, *age*, *occupation*, *genre* and *rating*. Since the *genre* of a rating record may contain more than one movie type, we made each record only correspond to a certain movie type and generated 2,101,815 records. For example, a rating record (*gender* =“*M*”, *age* =29, *occupation* =“*teacher*”, *genre* =“*action and science fiction*”, *rating* =“5 *stars*”) would be divided into two records (*gender* =“*M*”, *age* =29, *occupation* =“*teacher*”,

Algorithm 2 Constraint induced UPBN construction

```
Input:
    $D$, the training dataset
    $N_3$, the maximum number of iterations
Output:
    $(G^{h+1}, \theta^{h+1,0})$, the UPBN
Steps:
Generating $(G^0, \theta^{0,0})$ by Constraints 1 and 3
$h \leftarrow 0$
while true do
    Obtaining $(G^h, \theta^{h,t})$ by using Algorithm 1 to optimal the parameters of $(G^h, \theta^{h,0})$
    if $h = N_3$ then
        return $(G^h, \theta^{h,t})$
    end if
    Generating the set of candidate models $\mathcal{G}^h$ w.r.t. $G^h$ by Constraints 1 and 2
    for each $(G^*, \theta^*)$ in $\mathcal{G}^h$ do
        $\theta^*$ is optimized by E-step and M-step
    end for
    $(G^{h+1}, \theta^{h+1,0}) \leftarrow \arg \max_{(G^*,\theta^*)\in\mathcal{G}^h} BIC(G^*, \theta^*|D)$
    if $BIC(G^{h+1}, \theta^{h+1,0}|D) \leq BIC(G^h, \theta^{h,t}|D)$ then
        return $(G^h, \theta^{h,t})$
    else
        $h \leftarrow h + 1$
    end if
end while
```

genre =“*action*”, *rating* =“5 *stars*”) and (*gender* =“*M*”, *age* =29, *occupation* =“*teacher*”, *genre* =“*science fiction*”, *rating* =“5 *stars*”).

- **Synthetic Datasets.** The synthetic UPBN is shown in Fig. 1, which is manually constructed by Netica toolkit[1] and the constraints given in Section 3. The synthetic UPBN and corresponding sample dataset were adopted to test UPBN's expression of dependence relations for preference modeling from rating data. In addition, the synthetic MovieLens-1M datasets were generated by adding additional attributes to test the efficiency of UPBN construction.

5.1.2. Evaluation metrics

- **Execution time, speedup and parallel efficiency.** We adopted the execution time, speedup and parallel efficiency upon the Spark cluster to test the efficiency of Algorithms 1 and 2 respectively.
- **Structural difference.** To test the effectiveness of our proposed method for UPBN construction, we compared the structural differences between the synthetic UPBN and the UPBNs constructed by MSC [40] and our proposed method.
- **BIC score.** BIC scores of UPBNs under various constraints and algorithms are compared to further test the effectiveness of our proposed method for UPBN construction.
- **Precision, recall and F1-score.** To test the effectiveness of UPBN for preference modeling, we tested the precision, recall, F1-score of UPBN based preference estimation, calculated by the top-*k* preferences obtained according to the results of probabilistic reasoning of UPBN. Precision, recall and F1-score are defined as

$$precision = \frac{TP}{TP + FP} \tag{18}$$

$$recall = \frac{TP}{TP + FN} \tag{19}$$

$$F1 - score = \frac{2 \times precision \times recall}{precision + recall} \tag{20}$$

where *TP*, *FP* and *FN* are the correctly estimated preferences, incorrectly estimated ones and undiscovered ones, respectively. Intuitively, precision is the proportion of correctly estimated preferences in the estimated ones, recall is the proportion of correctly estimated preferences in the statistics, and F1-score is the weighted average of precision and recall.

- **Mean absolute error (MAE) and root mean squared error (RMSE).** MAE and RMSE were adopted to measure the accuracy (i.e., magnitude of errors) in UPBN based rating prediction, defined as

$$MAE = \frac{\sum_{d=1}^{|D|} |r_d - \hat{r}_d|}{|D|} \tag{21}$$

$$RMSE = \sqrt{\frac{\sum_{d=1}^{|D|} (r_d - \hat{r}_d)^2}{|D|}} \tag{22}$$

where D, r_d and $\hat{r}_d$ is the test dataset, the dth actual rating and the dth predicted rating, respectively.

5.1.3. Comparison models

- **MSC.** To test the effectiveness of our proposed method for UPBN construction, we adopted MSC [40] as the comparison method on structural difference.
- **LAD and pLSA.** To test the effectiveness of UPBN for preference estimation, LDA [28] and pLSA [27] were adopted as the comparison models on precision, recall and F1-score. The user–movie matrix was established upon LDA and pLSA, where user, movie, preference on movie type and rating of each user on movies were regarded as document, word, topic and word frequency respectively. The top-*k* preferred movie types were obtained from the probability distributions of the user-preference matrix.
- **PMF and BPMMF.** To test the effectiveness of UPBN for rating prediction, PMF [32] and BPMMF [34] were adopted as the comparison models on MAE and RMSE.

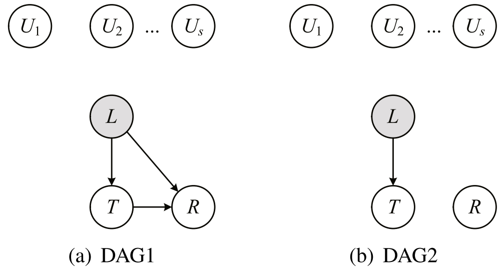
5.1.4. Implementation

- **Environment.** The experiments were conducted upon a TS860 server with 4 Intel(R) Xeon(R) E7-4830 v2 @ 2.20 GHz CPUs and 755 GB memory, in which each CPU has 10 physical cores. In the Spark cluster, 4 *Worker* processes are configured and each *Worker* contains a CPU. The version of Spark and Linux is 1.6.1 and Ubuntu 14.04.5 LTS server respectively. All codes were written in Scala.
- **Parameters.** The maximal number of iterations of EM for parameter learning and SEM for structure construction (i.e., N_1 and N_3) was fixed to 100. The convergence threshold of EM based parameter learning (i.e., δ) was set to 1×10^{-5}. Each experiment was repeated for 5 times, and the average is reported here.

5.2. Experimental results of efficiency

In the experiments, we first gave 2 initial DAGs shown in Fig. 5, denoted as DAG1 and DAG2, satisfying Constraint 1 and Theorem 1 respectively. Then, we used CD to denote the dependence relations satisfying Constraint 2, and CCPT and RCPT to respectively denote the initial CPTs satisfying Constraint 3 and those generated randomly.

[1] https://norsys.com/netica.html.



(a) DAG1 (b) DAG2

Fig. 5. Two initial DAGs.

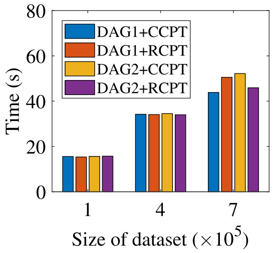
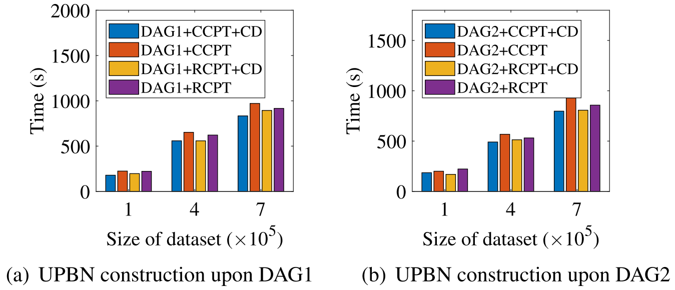


Fig. 6. Execution time of parameter learning upon various constraints.



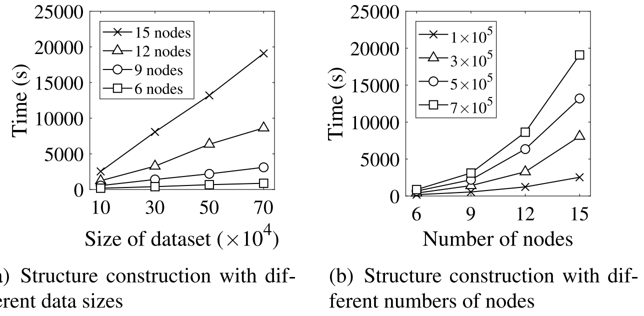
(a) UPBN construction upon DAG1 (b) UPBN construction upon DAG2

Fig. 7. Execution time of UPBN construction upon various constraints.

5.2.1. *Efficiency of constraint induced UPBN construction*

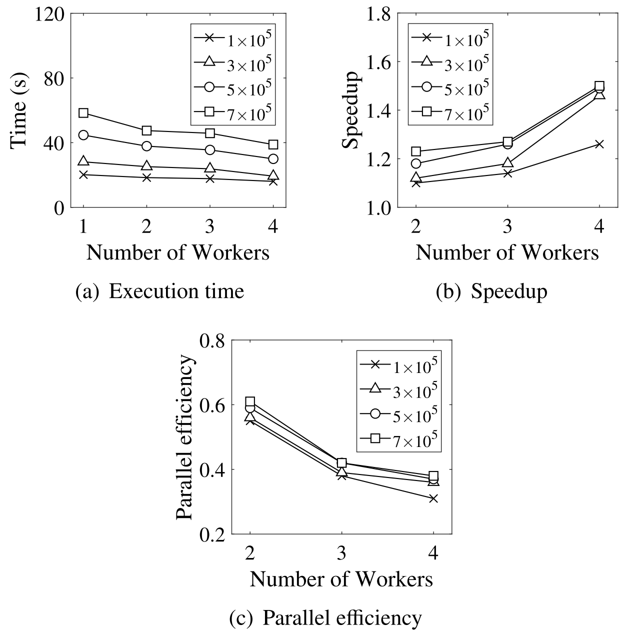
Upon the Spark platform, we tested the influence of constraints on the execution time of Algorithm 1 for parameter learning under various combinations of constraints on DAG and CPTs with the increase of samples from the MovieLens-1M dataset, shown in Fig. 6. Meanwhile, we tested the influence of constraints on the execution time of Algorithm 2 for UPBN construction with the identical configuration upon DAG1 and DAG2, shown in Fig. 7. It can be seen that the execution time of parameter learning with the strong constrains (i.e., DAG1+CCPT) is slightly less than those in the other 3 cases for the situation with 7 × 10^5 samples. Moreover, the execution time of UPBN construction with CD is less than those without CD, which holds for DAG1 and DAG2. Both the execution time of parameter learning and that of UPBN construction are increased linearly with the increase of data size, but the strong constraints can reduce the execution time of UPBN construction to a certain extent.

To test the efficiency of UPBN construction with different data sizes and numbers of nodes, we tested the execution time of Algorithm 2 with the strong constraints (i.e, DAG1+CCPT+CD) by the synthetic MovieLens-1M datasets, shown as Fig. 8. Note that the execution time of UPBN construction increases linearly with the increase of data size for all situations, while increases significantly with the increase of nodes in the UPBN. This is consistent with the complexity of Algorithm 2, since the structural constraints greatly limit the number of candidate models although the increase of nodes makes the candidate models increased exponentially.



(a) Structure construction with different data sizes (b) Structure construction with different numbers of nodes

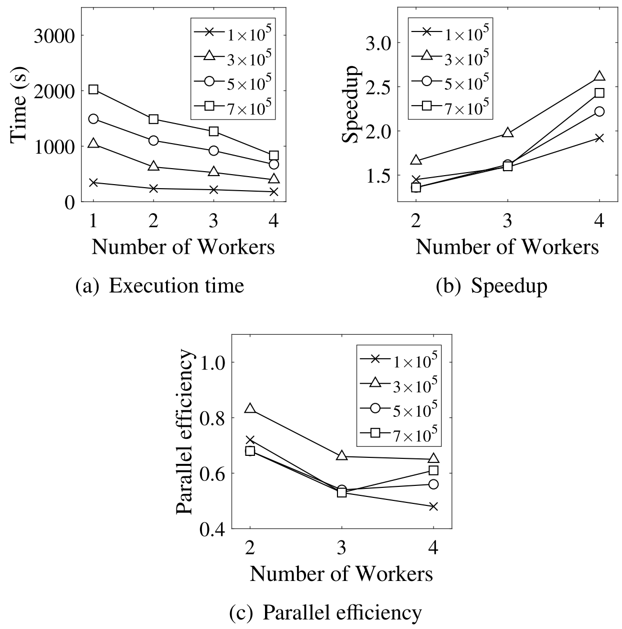
Fig. 8. Execution time of UPBN construction with different data sizes and numbers of nodes.



(a) Execution time (b) Speedup

(c) Parallel efficiency

Fig. 9. Execution time, speedup and parallel efficiency of parameter learning.



(a) Execution time (b) Speedup

(c) Parallel efficiency

Fig. 10. Execution time, speedup and parallel efficiency of UPBN construction.

Table 2
UPBNs with various numbers of different edges.

Constraint	Number of different edges								
	0	1	2	3	4	5	6	7	8
DAG1+CCPT+CD	18		27	3	2				
DAG1+CCPT	18		27	3	2				
DAG1+RCPT+CD			2	1	47				
DAG1+RCPT			2	1	47				
DAG2+CCPT+CD			12		11	5	13	9	
DAG2+CCPT			5		10	16	15	2	2
DAG2+RCPT+CD			1		1	4	17	27	
DAG2+RCPT			1		1	4	17	18	9

Table 3
Precision of estimated user preferences.

Number of users	Method	Top-*k*				
		3	5	7	9	11
100	UPBN	**0.703**	**0.722**	**0.796**	**0.798**	**0.822**
	LDA	0.650	0.690	0.756	0.779	0.812
	pLSA	0.632	0.671	0.734	0.757	0.790
900	UPBN	**0.690**	**0.738**	**0.798**	**0.811**	**0.829**
	LDA	0.674	0.707	0.767	0.791	0.820
	pLSA	0.655	0.687	0.745	0.768	0.796

5.2.2. Efficiency of Spark based UPBN construction

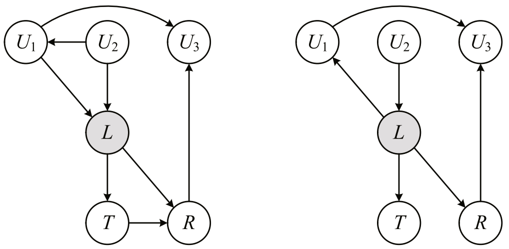
To test the efficiency of Spark based UPBN construction, we tested the execution time, speedup and parallel efficiency of our proposed algorithms for parameter learning and UPBN construction by varying the size of MovieLens-1M dataset from 1×10^5 to 7×10^5, shown in Figs. 9 and 10 respectively. It can be seen that the execution time and parallel efficiency of parameter learning and those of UPBN construction decrease with the increase of the number of *Workers* under various data sizes, while the speedup increases linearly. Although the speedup is increased from 1.10 to 1.50 for parameter learning and from 1.45 to 2.43 for UPBN construction, the speedup does not approach to the ideal values (e.g., speedup 2 for 2 cores) and the parallel efficiency is decreased with the increase of the number of *Workers*. The reason is that the communication cost among *Workers* exceeds the benefits by increasing the number of *Workers*. It can be seen from Fig. 10(c) that the benefits can be improved with the increase of data size, which guarantees the scalability of our method for UPBN construction.

5.3. Experimental results of effectiveness

5.3.1. Constraint induce UPBN construction

To test the effectiveness of the learned UPBN to represent the dependence relations, we compared the UPBN constructed by our method with that by MSC upon the same initial DAG, shown in Figs. 11(a) and 11(b) respectively. By comparing the directed edges (i.e., dependence relations), it can be seen that both of the UPBNs in Fig. 11 include the directed edges consisting of $\langle U_1(gender), U_3(occupation)\rangle$, $\langle U_2(age), L(preference)\rangle$, $\langle L(preference), T(type)\rangle$ and $\langle L(preference), R(rating)\rangle$. These dependence relations are intuitively correct in terms of the specific sense of the variables in MovieLens-1M. However, it is clearly unreasonable that the UPBN in Fig. 11(b) does not include $\langle T(type), R(rating)\rangle$, but it contains $\langle L(preference), U_1(gender)\rangle$ except for $\langle R(rating), U_3(occupation)\rangle$. Thus, the UPBN constructed by our method is more expressive than that by MSC.

To further test the effectiveness of the constraints, we constructed 400 UPBNs with various combinations of constraints by the synthetic datasets, in which the number of different edges is counted by summing the numbers of additional, missing or reverse edges between the synthetic UPBN and the UPBN constructed by our method. The UPBN will be considered better if there are fewer different edges, by which the quality of the UPBNs can be assessed intuitively. We recorded the numbers of different edges of the UPBNs constructed with various constraints respectively, shown in Table 2.



(a) UPBN constructed by our proposed method

(b) UPBN constructed by MSC

Fig. 11. UPBNs constructed by our method vs. maximal semi-cliques.

It can be seen that the UPBNs constructed with DAG1 have fewer different edges than those with DAG2 under the same conditions, which indicates that Constraint 1 is effective for UPBN construction. Specifically, all UPBNs with DAG1 have no more than 4 different edges and 36% of the UPBNs is structurally consistent with the synthetic UPBN, but the UPBNs with DAG2 have more than 4 different edges and there is no UPBN that is structurally consistent with the synthetic UPBN. Moreover, the UPBNs with DAG1+CCPT+CD have one different edge on average, but those with DAG1+RCPT+CD have 4 different edges, which indicates that Constraint 3 is effective for UPBN construction. Similarly, the UPBNs with DAG2+RCPT+CD have 6 different edges on average, but those with DAG2+RCPT have 7 different edges, which indicates that Constraint 2 is also effective. Thus, the constraints given in Section 3 are effective to guarantee the reliability of UPBN constructed from rating data.

In addition, we first compared the BIC scores of our constructed UPBNs with various combinations of constraints by the MovieLens dataset to measure whether our proposed constraints can make UPBN fit the given dataset better, shown in Fig. 12(a). Then, we also compared the BIC scores of the UPBNs constructed by our method with those by MSC, shown in Fig. 12(b). On the one hand, the BIC scores of UPBNs with DAG1+CCPT are larger than those with DAG1+RCPT, which indicates that the constraint

Table 4
Recall of estimated user preferences.

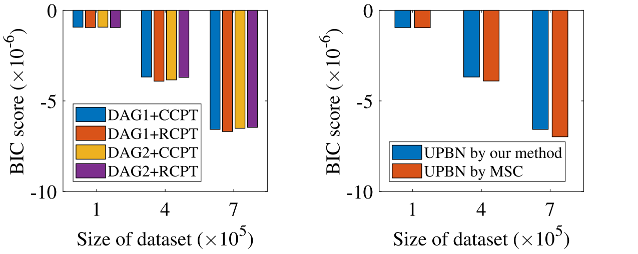
Number of users	Method	Top-k				
		3	5	7	9	11
100	UPBN	**0.192**	**0.329**	**0.507**	**0.654**	**0.822**
	LDA	0.178	0.314	0.482	0.638	0.812
	pLSA	0.177	0.313	0.480	0.636	0.809
900	UPBN	**0.189**	**0.337**	**0.510**	**0.665**	**0.829**
	LDA	0.185	0.322	0.490	0.649	0.820
	pLSA	0.184	0.321	0.488	0.646	0.816

Table 5
F1-score of estimated user preferences.

Number of users	Method	Top-k				
		3	5	7	9	11
100	UPBN	**0.302**	**0.452**	**0.620**	**0.719**	**0.822**
	LDA	0.279	0.432	0.588	0.702	0.812
	pLSA	0.279	0.431	0.588	0.701	0.811
900	UPBN	**0.296**	**0.462**	**0.622**	**0.731**	**0.829**
	LDA	0.290	0.443	0.598	0.713	0.820
	pLSA	0.289	0.442	0.597	0.712	0.818

Table 6
Preferences by UPBN and statistics.

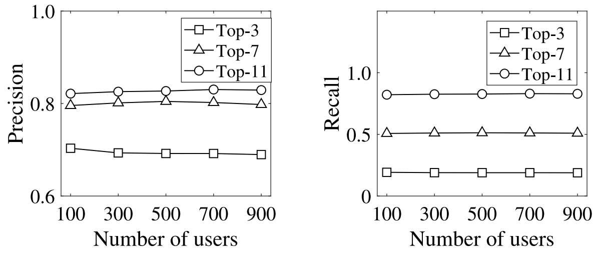
UserID	U_1	U_2	U_3	Preference(Probability)	
				UPBN	Statistics
1815	1	1	1	5 (21.2%)	5(25.0%)
878	1	2	2	8(19.6%)	5(18.1%)
1825	1	3	7	8(16.5%)	8(17.0%)
141	1	4	13	15(22.7%)	15(17.3%)
560	1	5	15	1(20.1%)	1(18.0%)
1043	2	6	11	8(19.2%)	5(21.3%)
1398	2	7	16	14(21.2%)	8(28.1%)
5411	2	1	19	4(23.4%)	4(28.9%)
798	2	3	20	5(24.3%)	8(21.8%)
704	2	4	8	16(23.6%)	16(16.5%)



(a) BIC scores of UPBNs under various constraints

(b) BIC scores of UPBN by our method and MSC

Fig. 12. BIC scores of UPBNs.



(a) Precision

(b) Recall

Fig. 13. Precision and recall with the increase of users.

of parameters can make UPBNs fit the given dataset well. On the other hand, the BIC scores of UPBNs with DAG1+CCPT are slightly lower than those with DAG2+CCPT, which is consistent with the BIC metric given in Eq. (16) since the structural constraint may increase the number of dependence relations so that the size of parameters is also increased. Meanwhile, the BIC scores of UPBNs constructed by our method are larger than those by MSC upon various data sizes. Therefore, we conclude that our proposed constraint induced method is effective for UPBN construction.

5.3.2. UPBN based preference estimation

To test the effectiveness of UPBN based preference estimation, we first adopted 80% (about 168×10^4 samples) of the MovieLens dataset as training data and 20% (about 42×10^4 samples) as test data. Then, we compared the precision, recall and F1-score of the top-k preferred movie types estimated by UPBN, LDA and pLSA. Note that the results of $P(L = l|\mathcal{U})$ in descending order were adopted as the top-k preferred movie types, and the movie types in descending order of weighted rating were adopted as the top-k statistic movie types.

As for 100 and 900 users, the precision, recall and F1-score of the top-k preferred movie types by UPBN, LDA and pLSA are shown and compared in Tables 3–5 respectively by varying the value of k from 3 to 11, where the best results are highlighted in bold. It can be seen from Table 3 that even the top-3 precisions are larger than 60%, and the precision, recall and F1-score by UPBN are greater than those by LDA and pLSA. This indicates that the estimated preferences can reflect the statistic preferences to a certain extent. Moreover, the trends of precision, recall and F1-score are increased with the increase of k. The precision and recall by UPBN are evaluated with different numbers of users, shown in Fig. 13. As can be seen from Fig. 13, the precision and recall by UPBN are not sensitive to the number of users, which confirms that the UPBN constructed in this paper is scalable even for the situations with massive users or rating data.

Intuitively, user preference is not only influenced by rating frequency, but also by rating levels. Therefore, 2 additional experiments were conducted to further test the effectiveness of UPBN based preference estimation. In view of rating frequency, $\arg\max_t P(T = t|\mathcal{U})$ by the probabilistic reasoning with UPBN was adopted as the top-1 estimated rating preference. By direct statistics from test data, the movie type with the largest probability (i.e., rating frequency) was adopted as the top-1 statistic rating preference. Then, we compared these 2 rating preferences for 10 randomly selected users, shown in Table 6. It can be seen

Table 7
Top-5 preferences by different methods.

UserID	Preferred movie(Probability)		
	UPBN	Rating frequency	Average rating
1815	15(20.3%)	15(25.4%)	12(2.0%)
	8(18.9%)	1(15.7%)	9(2.9%)
	1(18.3%)	5(13.7%)	17(5.9%)
	17(9.3%)	8(9.8%)	8(9.8%)
	5(7.5%)	2(6.8%)	3(2,0%)
560	1(25.9%)	1(30.7%)	18(1.1%)
	2(18.1%)	2(15.0%)	16(8.6%)
	16(17.5%)	15(14.4%)	5(3.2%)
	15(17.1%)	16(8.6%)	17(4.0%)
	8(8.1%)	8(6.1%)	2(15.0%)
798	14(21.5%)	5(14.5%)	18(0.6%)
	8(16.1%)	1(12.0%)	14(7.0%)
	5(15.6%)	8(11.7%)	9(5.2%)
	9(15.2%)	2(9.9%)	4(7.4%)
	4(9.8%)	11(9.1%)	12(2.5%)

Table 8
MAE of predicted ratings by MLBN, PMF and BPMMF.

Method	Number of users						
	100	300	500	700	900	1100	1300
UPBN	**0.905**	**0.892**	**0.895**	**0.897**	**0.900**	**0.917**	**0.912**
PMF	1.346	1.262	1.294	1.312	1.298	1.305	1.391
BPMMF	1.281	1.146	1.187	1.180	1.094	1.182	1.080

Table 9
RMSE of predicted ratings by MLBN, PMF and BPMMF.

Method	Number of users						
	100	300	500	700	900	1100	1300
UPBN	1.091	1.090	1.098	1.092	1.082	1.104	1.104
PMF	1.463	1.521	1.397	1.327	1.414	1.395	1.446
BPMMF	**0.987**	**1.013**	**1.062**	**0.976**	**0.994**	**1.035**	**1.014**

that the top-1 estimated rating preferences for 60% of users are consistent with their statistics.

In view of user preference, the results of $P(L = l|\mathcal{U})$ in descending order were adopted as the top-5 preferred movie types, and the top-5 statistic movie types with high rating frequency and average rating level were obtained respectively by direct statistics for 3 users randomly selected, shown in Table 7. It is worth noting that 100% of the top-5 preferred types are included in the top-5 statistic movie types for 3 users, although just some preferred types are included in the top-5 statistical movie types only with high rating frequency or average rating levels. Basically, the preferred movie types in top rankings correspond to the types with higher rating frequency or average rating level than others. In addition, the estimated preferences are not completely consistent with the statistics, since the estimated preferences are based on the users with identical demographic properties, but statistics are based on individual users. Thus, we conclude that user preferences estimated by UPBN are relatively reasonable by considering both rating frequency and rating level.

5.3.3. UPBN based rating prediction

To test the effectiveness of the learned UPBN for rating prediction, we compared the results of rating prediction by UPBN, PMF and BPMMF on MAE and RMSE by varying the number of users in test date from 100 to 1300, shown in Tables 8 and 9, where the best results are highlighted in bold. In terms of MAE, UPBN outperforms PMF and BPMMF by 29.3%−34.4% and 15.5%−29.3% respectively. In terms of RMSE, BPMMF is slightly better than UPBN by 3.3% − 10.6%, while UPBN is much better than PMF by 17.7% − 28.3%. The above results verify that UPBN is effective for rating prediction with reasonable errors.

5.4. Summary

From these experimental results on different datasets, we find the following:

- UPBN can be used to represent the dependence relations among the observed and latent variables effectively. Specifically, the UPBN constructed by our proposed constraint induced method are basically consistent with the UPBN manually constructed on graphic structure. Furthermore, the BIC scores of UPBN by our method are slightly larger than those by MSC.
- By incorporating latent variable and dependence relations into preference modeling, our proposed method outperforms other state-of-the-art methods. Specifically, UPBN outperforms LDA and pLSA on F1-score by 3.6% and 3.8% for preference estimation. Moreover, UPBN outperforms PMF and BPMMF on MAE by 31.3% and 22.2% for rating prediction.
- By designing Spark based and constraint induced algorithms, our proposed method for learning UPBN is scalable to the size of training data and the number of nodes (i.e, variables). In particular, the constraints can reduce execution time of UPBN construction. Meanwhile, the speedup of UPBN construction increases from 1.45 to 2.43 when the number of *Workers* is increased from 2 to 4.

6. Conclusions and future work

Leveraged on the underlying framework of Bayesian network, latent variable model, EM and SEM algorithms, we proposed the concept of UPBN by regarding user preference as a latent variable, as well as the constraint induced and Spark based parallel algorithms for UPBN construction and preference modeling. The advantages of the method given in this paper are summarized as follows:

(1) Theoretically, UPBN is well interpretable upon the probability theory and graphical model. Practically, UPBN is straightforward to represent and infer uncertainties generally implied in real-world rating data and corresponding data-centered applications or personalized services.

(2) The proposed constraints formally reflect the practical characteristics in rating data, intuitive sense of user preference, and specialty of the latent variable. As well, the constraints make it possible to fuse the characteristics of BN and latent variable with the inherence of EM, so that the feasibility and applicability of UBPN in rating data analysis with real applications could be guaranteed.

(3) Spark based parallel algorithms facilitate the large amount of iterations in parameter learning and maximum likelihood estimation, and make these time-consuming computations be implemented and thus make the proposed methodology be practical accordingly.

(4) Extensive experiments verify the efficiency and effectiveness of UPBN, which also embodies the practical roadmap of UPBN construction from realistic rating data, as well as the applications to preference estimation and rating prediction.

However, more datasets and baseline methods should be incorporated in experiments to refine the proposed methods. Definitely, our work in this paper establishes the basis for some other open research issues. First, UPBN only concerns one latent variable to represent one dimensional preference (i.e., movie *type*), which should be further extended since multiple dimensions of preferences are ubiquitous in the real world, such as *genre*, *year*, *country* etc. Then, in response to the ever-increasing rating data that reflect the dynamic evolving preference or behavior of users, it is necessary to develop incremental algorithms to model construction and knowledge reasoning. Meanwhile, it is necessary to develop online algorithms for learning parameters and structures considering the sparse data or long-tail situations. Furthermore, it is worthwhile to fuse the uncertain knowledge by our proposed method with knowledge graphs in a certain domain to enrich the constraints for model construction. These exactly are our future work.

CRediT authorship contribution statement

Kun Yue: Conceptualization, Methodology, Writing - original draft. **Xinran Wu:** Investigation, Software, Validation, Writing - review & editing. **Liang Duan:** Supervision, Writing - review & editing. **Shaojie Qiao:** Data curation, Resources. **Hao Wu:** Visualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This paper was supported by the National Natural Science Foundation of China (U1802271, 61772091, 61802035), the Science Foundation for Distinguished Young Scholars of Yunnan Province (2019FJ011), the Cultivation Project of Donglu Scholar of Yunnan University, and the Foundation for Undergraduates of Educational Department of Yunnan Province (2020Y0010).

References

[1] L. Boratto, E. Vargiu, Data-driven user behavioral modeling: from real-world behavior to knowledge, algorithms, and systems, J. Intell. Inf. Syst. 54 (1) (2020) 1–4.

[2] R. Bagher, H. Hassanpour, H. Mashayekhi, User preferences modeling using dirichlet process mixture model for a content-based recommender system, Knowl. Based Syst. 163 (2019) 644–655.

[3] Q. Liu, A. Reiner, A. Frigessi, I. Scheel, Diverse personalized recommendations with uncertainty from implicit preference data with the Bayesian Mallows model, Knowl. Based Syst. 186 (2019) URL: https://doi.org/10.1016/j.knosys.2019.104960.

[4] L. Wu, Y. Ge, Q. Liu, E. Chen, R. Hong, J. Du, M. Wang, Modeling the evolution of users' preferences and social links in social networking services, IEEE Trans. Knowl. Data Eng. 29 (6) (2017) 1240–1253.

[5] J. Chen, W. Zeng, J. Shao, G. Fan, Preference modeling by exploiting latent components of ratings, Knowl. Inf. Syst. 60 (1) (2019) 495–521.

[6] Y. Qu, B. Fang, W. Zhang, R. Tang, M. Niu, H. Guo, Y. Yu, X. He, Product-based neural networks for user response prediction over multi-field categorical data, ACM Trans. Inf. Syst. 37 (1) (2019) 5:1–5:35.

[7] G. Zhao, X. Qian, C. Kang, Service rating prediction by exploring social mobile users' geographical locations, IEEE Trans. Big Data 3 (1) (2017) 67–78.

[8] GroupLens, GroupLens, Movielens-1m dataset, 2020, http://grouplens.org/datasets/movielens/1m/.

[9] N. Friedman, Learning belief networks in the presence of missing values and hidden variables, in: Proc. 14th Int. Conf. Mach. Learn. (ICML), Nashville, Tennessee, USA, 1997, pp. 125–133.

[10] Z. Cheng, J. Shen, On effective location-aware music recommendation, ACM Trans. Inf. Syst. 34 (2) (2016) 13:1–13:32.

[11] K. Zhao, G. Cong, Q. Yuan, K. Zhu, SAR: A sentiment-aspect-region model for user preference analysis in geo-tagged reviews, in: Proc. 31st IEEE Int. Conf. Data Engineering (ICDE), Seoul, South Korea, IEEE Computer Society, 2015, pp. 675–686.

[12] Y. Guo, Z. Cheng, L. Nie, Y. Wang, J. Ma, M. Kankanhalli, Attentive long short-term preference modeling for personalized product search, ACM Trans. Inf. Syst. 37 (2) (2019) 19:1–19:27.

[13] A. Constantinou, N. Fenton, W. Marsh, L. Radlinski, From complex questionnaire and interviewing data to intelligent Bayesian network models for medical decision support, Artif. Intell. Med. 67 (2016) 75–93.

[14] F. Hsu, Y. Lin, T. Ho, Design and implementation of an intelligent recommendation system for tourist attractions: The integration of EBM model, Bayesian network and Google Maps, Expert Syst. Appl. 39 (3) (2012) 3257–3264.

[15] D. Koller, N. Friedman, Probabilistic Graphical Models - Principles and Techniques, MIT Press, 2009.

[16] Y. Li, H. Chen, J. Zheng, A. Ngom, The max-min high-order dynamic Bayesian network for learning gene regulatory networks with time-delayed regulations, IEEE/ACM Trans. Comput. Biol. Bioinform. 13 (4) (2016) 792–803.

[17] B. Cobb, L. Li, Bayesian network model for quality control with categorical attribute data, Appl. Soft Comput. 84 (2019) URL: https://doi.org/10.1016/j.asoc.2019.105746.

[18] A. Dempster, N. Laird, D. Rubin, Maximum likelihood from incomplete data via the EM algorithm, J. R. Stat. Soc. Ser. B Stat. Methodol. 39 (1) (1977) 1–38.

[19] S. Lauritzen, The EM algorithm for graphical association models with missing data, Comput. Statist. Data Anal. 19 (2) (1995) 191–201.

[20] C. Jin, Y. Zhang, S. Balakrishnan, M. Wainwright, M. Jordan, Local maxima in the likelihood of Gaussian mixture models: Structural results and algorithmic consequences, in: Proc. Annual Conf. Neural Inf. Processing Syst. (NIPS), Barcelona, Spain, 2016, pp. 4116–4124.

[21] N. Friedman, The Bayesian structural EM algorithm, in: Proc. 14th Uncertainty Artif. Intell. (UAI), Madison, Wisconsin, USA, 1998, pp. 129–138.

[22] G. Schwarz, Estimating the dimension of a model, Ann. Statist. 6 (2) (1978) 461–464.

[23] Apache, Apache spark, 2020, http://spark.apache.org.

[24] G. Shafer, P. Shenoy, Probability propagation, Ann. Math. Artif. Intell. 2 (1990) 327–351.

[25] N. Zhang, D. Poole, Exploiting causal independence in Bayesian network inference, J. Artificial Intelligence Res. 5 (1996) 301–328.

[26] N. Garg, S. Dhurandher, P. Nicopolitidis, J. Lather, Efficient mobility prediction scheme for pervasive networks, Int. J. Commun. Syst. 31 (6) (2018) URL: https://doi.org/10.1002/dac.3520.

[27] T. Hofmann, Probabilistic latent semantic analysis, in: Proc. 15th Uncertainty Artif. Intell. (UAI), Stockholm, Sweden, 1999, pp. 289–296.

[28] D. Blei, A. Ng, M. Jordan, Latent Dirichlet allocation, J. Mach. Learn. Res. 3 (2003) 993–1022.

[29] R. Gao, K. Yue, H. Wu, B. Zhang, X. Fu, Modeling user preference from rating data based on the Bayesian network with a latent variable, in: Proc. 17th WAIM Workshops, Nanchang, China, 2016, pp. 3–16.

[30] Y. Koren, Factorization meets the neighborhood: a multifaceted collaborative filtering model, in: Proc. 14th ACM Int. Conf. Knowl. Discovery and Data Min. (SIGKDD), Las Vegas, Nevada, USA, 2008, pp. 426–434.

[31] Y. Koren, Collaborative filtering with temporal dynamics, Commun. ACM 53 (4) (2010) 89–97.

[32] R. Salakhutdinov, A. Mnih, Probabilistic matrix factorization, in: Proc. Annual Conf. Neural Inf. Processing Syst. (NIPS), Vancouver, British Columbia, Canada, 2007, pp. 1257–1264.

[33] F. Tan, L. Li, Z. Zhang, Y. Guo, A multi-attribute probabilistic matrix factorization model for personalized recommendation, Pattern Anal. Appl. 19 (3) (2016) 857–866.

[34] K. Wang, W. Zhao, H. Peng, X. Wang, Bayesian probabilistic multi-topic matrix factorization for rating prediction, in: Proc. 25th Int. Joint Conf. Artif. Intell. (IJCAI), New York, NY, USA, IJCAI/AAAI Press, 2016, pp. 3910–3916.

[35] Q. Liu, F. Yu, S. Wu, L. Wang, A convolutional click prediction model, in: Proc. 24th ACM Conf. Inf. Knowl. Management (CIKM), Melbourne, VIC, Australia, ACM, 2015, pp. 1743–1746.

[36] Y. Zhang, H. Dai, C. Xu, J. Feng, T. Wang, J. Bian, B. Wang, T. Liu, Sequential click prediction for sponsored search with recurrent neural networks, in: Proc. 28th Conf. Artif. Intell. (AAAI), San Québec City, Québec, Canada, AAAI Press, 2014, pp. 1369–1375.

[37] Y. Qu, H. Cai, K. Ren, W. Zhang, Y. Yu, Y. Wen, J. Wang, Product-based neural networks for user response prediction, in: Proc. 16th Int. Conf. Data Min. (ICDM), Barcelona, Spain, 2016, pp. 1149–1154.

[38] R. Yin, K. Li, G. Zhang, J. Lu, A deeper graph neural network for recommender systems, Knowl. Based Syst. 185 (2019) URL: https://doi.org/10.1016/j.knosys.2019.105020.

[39] J. Zhao, X. Geng, J. Zhou, Q. Sun, Y. Xiao, Z. Zhang, Z. Fu, Attribute mapping and autoencoder neural network based matrix factorization initialization for recommendation systems, Knowl. Based Syst. 166 (2019) 132–139.

[40] G. Elidan, N. Lotner, N. Friedman, D. Koller, Discovering hidden variables: A structure-based approach, in: Proc. Annual Conf. Neural Inf. Processing Syst. (NIPS), Denver, CO, USA, MIT Press, 2000, pp. 479–485.

[41] C. He, K. Yue, H. Wu, W. Liu, Structure learning of Bayesian network with latent variables by weight-induced refinement, in: Proc. 5th Int. Workshop on Web-Scale Knowl. Representation Retrieval & Reasoning (Web-KR@CIKM), Shanghai, China, ACM, 2014, pp. 37–44.

[42] Y. Zhao, J. Xu, Y. Gao, A parallel algorithm for Bayesian network parameter learning based on factor graph, in: Proc. 25th IEEE Int. Conf. Tools with Artif. Intell. (ICTAI), Herndon, VA, USA, IEEE Computer Society, 2013, pp. 506–511.

[43] R. Lämmel, Google's mapreduce programming model - revisited, Sci. Comput. Program. 70 (1) (2008) 1–30.

[44] K. Yue, Q. Fang, X. Wang, J. Li, W. Liu, A parallel and incremental approach for data-intensive learning of Bayesian networks, IEEE Trans. Cybern. 45 (12) (2015) 2890–2904.

[45] T. Gao, D. Wei, Parallel Bayesian network structure learning, in: Proc. 35th Int. Conf. Mach. Learn. (ICML), StockholmsmäSsan, Stockholm, Sweden, in: Proceedings of Machine Learning Research, vol. 80, PMLR, 2018, pp. 1671–1680.