Neurocomputing 386 (2020) 208–220

Contents lists available at ScienceDirect

Neurocomputing

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

Joint Personalized Markov Chains with social network embedding for cold-start recommendation

Yijia Zhang[a,b], Zhenkun Shi[a,b,d,*], Wanli Zuo[a,b,*], Lin Yue[a,c], Shining Liang[a,b], Xue Li[d,e]

[a] Key Laboratory of Symbol Computation and Knowledge Engineering of Ministry of Education, Changchun 130012, China
[b] College of Computer Science and Technology, Jilin University, Jilin 130000, China
[c] School of Computer Science and Information Technology, Northeast Normal University, Changchun 130117, China
[d] School of Information Technology and Electrical Engineering, The University of Queensland, Brisbane 4072, Australia
[e] Dalian Neusoft University of Information, China

ARTICLE INFO

Article history:
Received 5 June 2019
Revised 26 September 2019
Accepted 3 December 2019
Available online 19 December 2019

Communicated by Dr. Guan Ziyu

Keywords:
Markov chains
User cold-start
Temporal information
Social network embedding

ABSTRACT

The primary objective of recommender systems is to help users select their desired items, where a key challenge is providing high-quality recommendations to users in a "cold-start" situation. Recent advances in tackling this problem combine social relations and temporal information and integrate them into a unified framework. However, these methods suffer from a limitation that there not always exist links for the newcomers, thus these users are filtered in related studies. To break the boundary, in this paper, we propose a Joint Personalized Markov Chains (JPMC) model to address the cold-start issues for implicit feedback recommendation system. In our study, we first utilize user embedding to mine Network Neighbors, so that newcomers without relations can be represented by similar users, then we designed a two-level model based on Markov chains at both user level and user group level respectively to model user preferences dynamically. Experimental results on three real-world datasets have shown that our model can significantly outperform the state-of-the-art models.

1. Introduction

Recommender systems (RS) play a vital role in daily lives as they assist prospective buyers in making proper decisions. RS aim at providing users with effective recommendations based on their intuitions and preferences. In literature, there are two commonly used techniques for recommendation: *Collaborative Filtering (CF)* based methods [1–3] and *Content-based (CB)* methods [4,5]. However, these methods suffer the most known problem, the "user cold-start (UCS)" problem. The UCS is related to recommendations for novel users lacking explicit information (e.g., ratings). This problem may lead to the loss of new users due to the low accuracy of recommendations at the early stage [6].

With the achievement in social media, many social-based methods have been proposed to deal with the UCS problem. These methods predict new users' preferences based on their friends by introducing social relations, and they usually improve the performances of traditional recommendation techniques such as matrix factorization (MF) [7–10] and Bayesian personalized ranking (BPR) [11,12]. However, these methods have two drawbacks: (1) Social relations are usually sparse in most real-life systems, leading to explicit social links are not always available, especially for cold-start users. For a more intuitive understanding, we first investigate the distribution of social relations for both full users and cold-start users (users with fewer than 5 feedbacks). As we can see from Fig. 1, most of the full users have less than 5 explicit social links in Epinions and Last.fm network. Meanwhile, cold-start users have fewer social links than full users, even in the network of Ciao, over half of cold-start users have less than 10 explicit social links. (2) Not all users share preferences with his/her friends. For example, an active user always connects with many users so that some friends do not have common interests with he/she. Thus, it is not appropriate to use social relations directly for the recommendation. Taking these into consideration and inspired by the success of network embedding in related works [13–16], we employ network embedding in our model. By extending existing network embedding methods, we select a group of users for each user as their Network Neighbors to replace friends.

Another important factor we should not neglect is that both of the user preference and social influence are drifting over time. So, one practical way to deal with UCS problem is integrating temporal information into the recommendation. By utilizing

* Corresponding authors at: Key Laboratory of Symbol Computation and Knowledge Engineering of Ministry of Education, Changchun 130012, China.
E-mail addresses: shizk14@mails.jlu.edu.cn (Z. Shi), zuowl@jlu.edu.cn (W. Zuo).

https://doi.org/10.1016/j.neucom.2019.12.046
0925-2312/

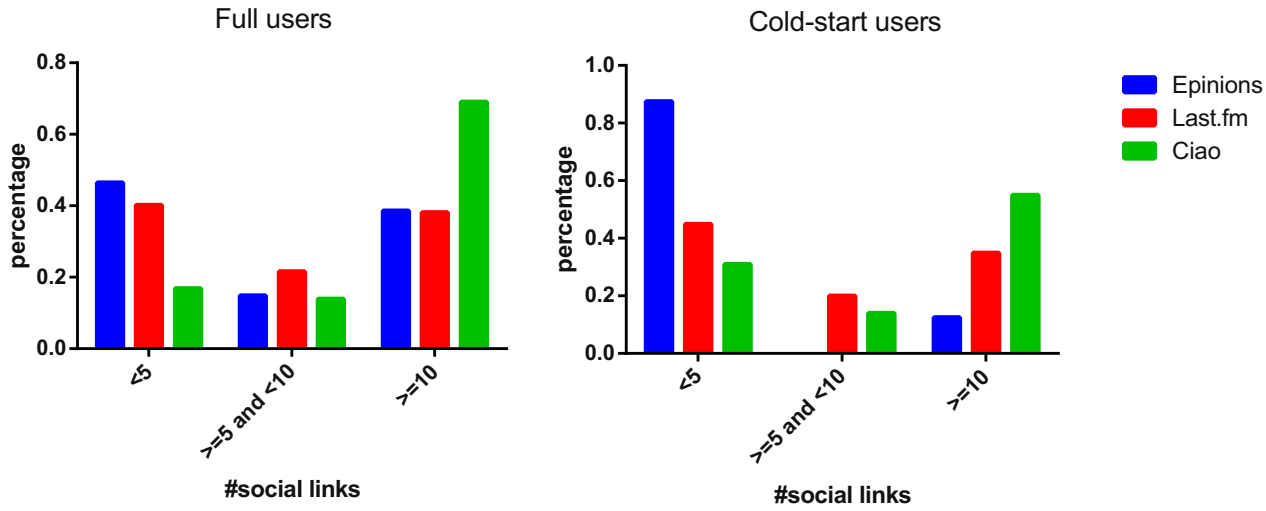


Fig. 1. The illustration of social connections number. The horizontal axis represents the range of social connections, and the vertical axis represents the percentage of the user.

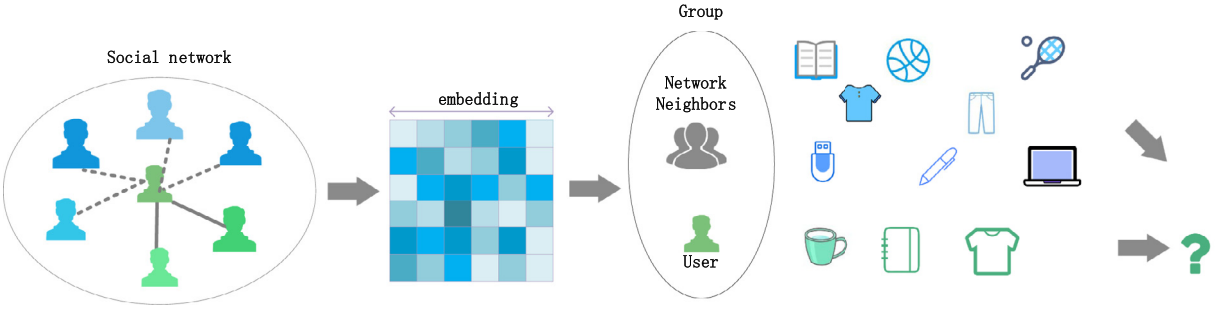


Fig. 2. The schematic illustration of our model. On the left, the solid arrows indicate the connection between friends, and the dotted arrows indicate the connection we added. In the middle, there are users embedding representations pre-trained. On the right, there are consumed sequences of users and Network Neighbors and items in different locations mean they are consumed at different time.

temporal information to enhance UCS in the recommendation, a lot of efforts have been made by previous researchers [17–19]. These methods are based on Markov chains to model transitions probabilities between two items, which can capture dynamic user preference, but they lack other side information such as social information, and as such, they help little for addressing UCS problem. Furthermore, methods combining temporal information and social information are proposed to make recommendation [20,21], these methods have verified that combining temporal information and social information can help to deal with UCS problem effectively, but they also cannot solve the problem that social relations are sparse in real systems.

Based on the considerations above, we propose a novel joint model based on Markov Chains. Fig. 2 shows the main idea of our model. We first create links between users with high relevance in addition to explicit social links. Then we utilize Node2vec model to pre-train the representations of the users. According to the representations, we select Network Neighbors for each user. Finally, we model the user's both static preference and dynamic preference at the user level and group level. Our contributions are summarized as follows:

1. To address the recommendations for cold-start users lacking explicit social relations, we apply network embedding technologies to select Network Neighbors instead of using social relations directly. Particularly, we use both explicit and implicit links to train the embeddings of users.
2. To model user's both static and dynamic preferences according to Network Neighbors, we design a joint model to capture these preferences, we fully considered the temporal information and integrate it with user preference into a single framework at both user and group level.
3. We evaluate the proposed method on three real-world datasets, and empirical results show that the proposed model improves recommendation performance over all the baselines significantly. Our model performs better in user cold-start issues compared to state-of-the-art methods, especially for users who have less social links.

2. Related work

2.1. User cold-start

The user cold-start problem refers to providing recommendations for users who have no history of consumption, it is pervasive in various recommendation applications, thus addressing user cold-start issues is critically significant for improving the accuracy of recommendation. At the early stages, there are several studies proposed to mitigate the UCS problem by extending traditional collaborative filtering methods [22–25]. Zhou et al. [22] proposed a novel cold-start recommendation method which solves the problem of initial interview construction within the context of learning user and item profiles. Rashid et al. [23] studied six techniques that collaborative filtering can use to learn about new users. Later, they extended it and studied the feasibility of many item selection measures for the new user problem [24]. Kim et al. [25] proposed a

collaborative filtering method to provide an enhanced recommendation quality derived from user-created tags.

The advent of online social network stimulates the working on addressing user cold-start issues [9,10,12,26–28]. Jamali and Ester [9] introduced a novel probabilistic matrix factorization model SocialMF based on the assumption that users' latent feature vectors are dependent on their social relations. Chaney et al. [10] presented social Poisson factorization, a Bayesian model that incorporates users' latent preferences for items with the latent influences of her friends. Wang et al. [12] proposed model to approximate tie strength and the popular Bayesian Personalized Ranking (BPR) model to incorporate the distinction between strong and weak ties. Pan and Chen [26] presented a method called GBPR that aggregates groups of users' preferences on items to reduce modeling uncertainty. Sedhain et al. [27] proposed an effective learning-based liner approach with social information for the user cold-start problems. In particular, Zhao et al. [11] proposed a Social Bayesian Personalized Ranking (SBPR) model, they used social connections to better estimate users' rankings of products, which significantly increased the accuracy in a cold-start scenario.

2.2. Temporal recommendation

In the temporal recommendation domain, many methods based on Markov Chain are widely used to tackle the next-item prediction problem [18,19,21,29–31]. Markov Chain considers temporal data as a stochastic process over discrete random variables, it is a popular tool in the recommender system community. Based on it, Rendle et al. [17] proposed Factorizing Personalized Markov Chains (FPMC) that combines MF and (factorized) Markov chains to capture personalized and sequential patterns. He and McAuley [18] proposed a model named Factorized Sequential Prediction with Item Similarity Model (Fossil). They viewed users as a combination of the factors of the items they have interacted with, which in turn allows Fossil to provide sequential recommendations for cold-start users. They also proposed a Socially-Aware Personalized Markov Chains (SPMC) to deal with UCS problem [21], which is a combination of personalization, sequential dynamic, and social information. Song et al. [29] proposed a unified model called States Transition Pair-Wise Ranking Model. They combine LDA with first-order Markov chains to simultaneously model the users' long and short-term favorites. Chen et al. [31] put forward a framework of personalized interest-forgetting Markov model to better simulate the decisions of intelligent agents for personalized recommendations.

In conclusion, there are two main differences between our work and previous work: (1) We use Network Neighbors instead of friends to make up for the limitations of social relations. (2) We also model dynamic user preference at the group level in addition to the single user level.

3. Proposed model

In this section, we first present some definitions of our model. Secondly, we will pre-train the user embedding to gain Network Neighbors. Thirdly, we will introduce two structures we design according to Network Neighbors. Finally, we will show our proposed model in this paper.

3.1. Problem formulation

In this paper, we focus on user cold-start task in implicit feedback recommendation. In addition to the implicit feedback information, both timestamps and social relations are also available. Let $\mathcal{U}$ and $\mathcal{I}$ denotes user set and item set respectively. For each $u \in \mathcal{U}$, we use $\mathcal{I}_u{}^+$ to denote the items the user has performed.

Table 1
Notations.

Symbol	Description
$\mathcal{U}$, $\mathcal{I}$	User/item set
u, i	User $u \in \mathcal{U}$, item $i \in \mathcal{I}$
S_u	Sequence of user
$\mathcal{I}_u{}^+$	Positive item set of user u
$v_u{}^{U,I}$, $v_i{}^{I,U}$	User/item latent feature vectors
$v_i{}^{I,L}$, $v_i{}^{L,I}$	Item latent feature vectors
$\hat{x}_{u,i,l}$, $\hat{y}_{u,i,l}$	Estimated score user u gives item i given last item l
W	Matrix of user-item interaction
D_T	Subset of (u, i, l) at user level
D_G	subset of (u, i, l) at group level
Θ	Parameter set
λ	Regularization parameter
α	Balance parameter
K	Dimension of latent feature space
N_g	The number of Network Neighbors

According to the timestamp, each $u \in \mathcal{U}$ is observed with a feedback sequence $S_u = \{x_1, x_2, x_3, \ldots, x_t\}, \forall x_t \in \mathcal{I}_u{}^+$, where x_t is the consumption item of user u at time t. Our objective is to predict the next feedback x_{t+1} of each user in the consumption sequence and improve user cold-start issues accordingly. To better describe our methods, we have the following definitions, and the notation used throughout this paper is summarized in Table 1.

Definition 1. *Network Neighbors*. Given a user u, let $\mathcal{G}_u$ suggests the Network Neighbors of the user u. We define $\mathcal{G}_u$ the set of users with whom the user u has similar preferences.

Definition 2. *Group*. Given a user u, the group consists u and Network Neighbors $\mathcal{G}_u$, we define a group as $u \cup \mathcal{G}_u$.

Definition 3. *Group items*. Given a user u and Network Neighbors $\mathcal{G}_u$, the group items refer to the items consumed by $\mathcal{G}_u$ and user u.

Definition 4. *User sequence*. Given a user u, we define $S_u = \{x_1, x_2, x_3, \ldots, x_t\}$ as user sequence.

3.2. Selecting Network Neighbors from social network

As mentioned before, it is not realistic to integrate social information into recommendation directly. Therefore, instead of explicit social relations, we consider utilizing network embedding methods to select a group of users called Network Neighbors ($\mathcal{G}_u$) to represent the user preference at group level. Among the network embedding methods, Node2vec is a state-of-the-art method which preserves higher-order proximity between nodes [16]. It has a more flexible sampling strategy so that it is essential in network embedding domain. Therefore, we adopt node2vec to learn high-level network representations. Formally, the proposed function is derived as follows:

$$\max_f \sum_{n \in V} \log Pr((N_S(u)|f(u)) \tag{1}$$

Where f is the mapping function from nodes to feature representations. For every source node $n \in V$, they define $N_S(u) \subset V$ as a network neighborhood of node generated through a neighborhood sampling strategy S.

When training Node2vec model, each user is viewed as a node. Considering the sparsity of social relations, we use two types of links in term of edges to enhance the process of training: (1) The explicit social links in the social network. (2) Added links according to the PMI (Pointwise Mutual Information) [32] between two users. PMI is widely used in Information Research domain to measure the co-occurrence probability between two entries, it is defined as Eq. (2). We argue that users with higher PMI value are closer in their preference. Thus, for each user, we calculate the PMI values with other users and create links between users with top 10

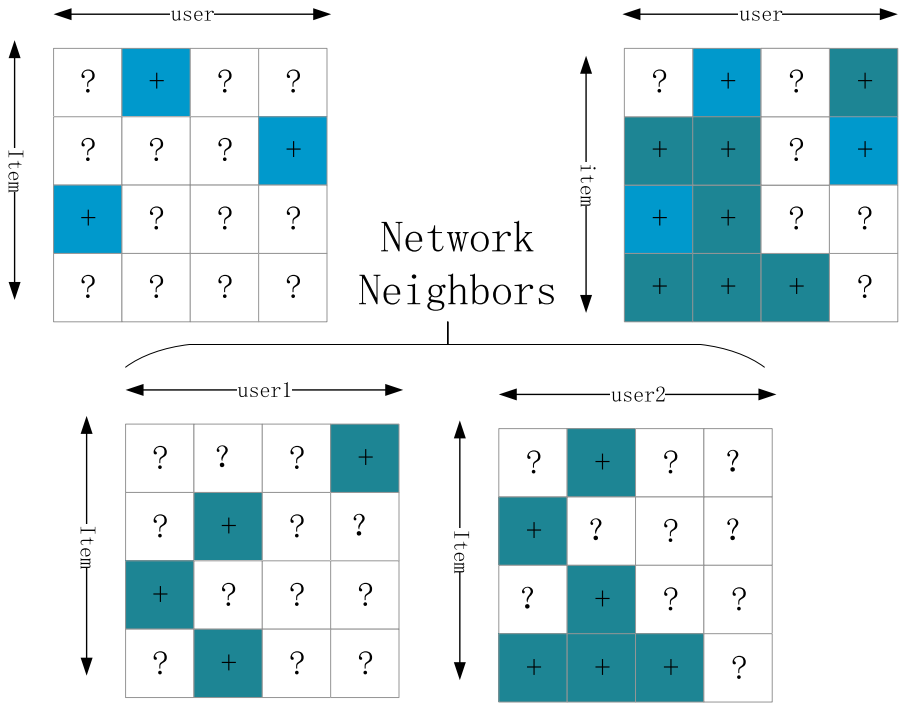


Fig. 3. The simplified diagram shows the process of constructing the SNA matrix according to the feedback of Network Neighbors. The left part is the original user-item matrix, while the right part is SNA matrix. Plus (+) indicates that a user prefers the item, and question mark (?) indicates unknown feedback.

PMI values. We added these links to explicit social links to train the user embedding together.

$$PMI(u,v) = \frac{\#(u_1,u_2)\cdot\mathcal{D}}{\#(u_1)\cdot\#(u_2)} \tag{2}$$

Where $\#(u_1,u_2)$ denotes the number of feedback both user u_1 and user u_2 have performed, $\mathcal{D} = \sum_{(u_1,u_2)}\#(u_1,u_2)$, $\#(u_1) = \sum_{u_2}\#(u_1,u_2)$ denotes the number of feedback user u_1 have performed, $\#(u_2) = \sum_{u_1}\#(u_1,u_2)$ denotes the number of feedback user u_2 have performed. After pre-training the node2vec model, we calculate the cosine similarity between users according to their embedding representations as follows:

$$cosine(p,q) = \frac{p^T q}{\|p\|\|q\|} \tag{3}$$

Where p and q are vector representations of two users. Ultimately, we select top-N users as for each user as his/her $\mathcal{G}_u$. Note that N value will be evaluated in later experiments.

3.3. Static Network Neighbors Augmented matrix

To integrate network embedding information into the recommendation, we first construct the static Network Neighbors Augmented matrix (SNA matrix). We assume that a user tends to show preference to the items that $\mathcal{G}_u$ like, thus we utilize feedback of $\mathcal{G}_u$ to fill the missing data in the original user-item matrix. Fig. 3 illustrates how we construct the SNA matrix. As we deal with implicit feedback where each entry is a binary value 1 or 0, we fill the missing entry in the original matrix with value 1 if the interaction is performed by $\mathcal{G}_u$. Through this method, we transformed the original sparse user-item matrix into an SNA matrix which is more denser. Formally, the SNA matrix $W \in \mathbb{R}^{M\times N}$ is designed as Eq. (4), it can be easily achieved by matrix multiplication.

$$w_{ui} = \begin{cases} 1, & \text{if interaction}(u,i)\text{is observed directly or indirectly.} \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

The NA matrix provides a static method to integrate user embedding information into the recommendation, which brings additional information to improve the UCS problem. It is used to capture static preference of the user along with the original user-item matrix.

3.4. Dynamic neighbors sequence

To better explore user preference over time, we further construct a dynamic neighbors sequence (DN sequence) to model users' dynamic preferences. We argue that users in a group may have the same preference at the same time, thus we can merge user sequences in a group by temporal information. Fig. 4 determines how to construct the DN sequence. To get the DN sequence, we first collect the feedback of all the users in a group (user and $\mathcal{G}_u$), then we reorder these feedback by time. Noted that we only keep the first occurrence for the same feedback.

As can be seen, the DN sequence can be viewed as a dynamic expression of NA matrix, which provides a dynamic method to integrate network embedding information into our model. Different from previous studies, DN sequence considers the time consistency of consume behaviors of all users in a group, so that a group can be viewed as a special user. By constructing the DN sequence, we can build successive relations for all the group items according to temporal information, and further model dynamic user preference at group level.

3.5. Joint Personalized Markov Chains Model (JPMC)

In this part, we present our joint model based on SNA matrix and DN sequence. Since FPMC incorporates matrix factorizing with Markov chains to learn static and dynamic latent factors respectively, we build our model under this framework. We will have a brief introduction of FPMC at first, then we will present our proposed joint model.

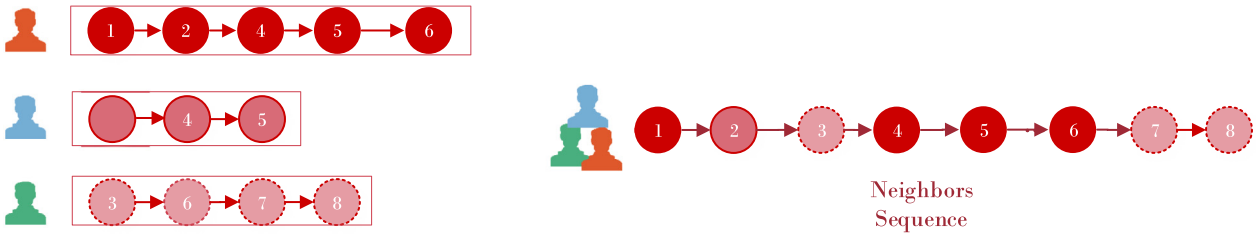


Fig. 4. The procedure for constructing the DN sequence. On the left side, there are three users in a group, where the number indicates the order in user's feedback sequence. On the right side, there is the DN sequence we constructed.

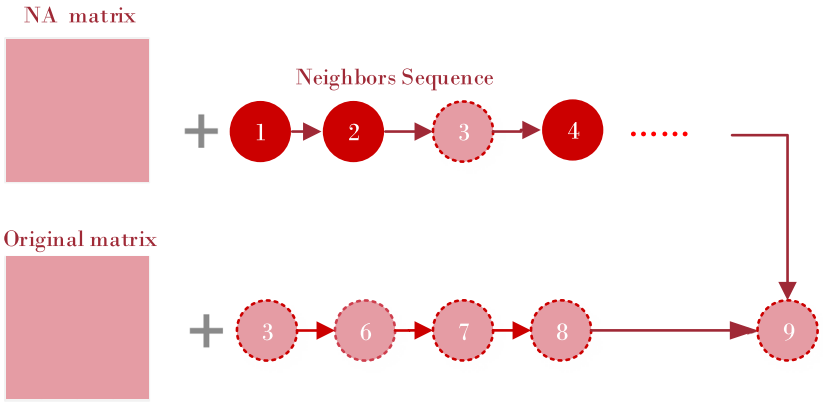


Fig. 5. The overall architecture of our model.

3.5.1. First-order FPMC

In FPMC, given a user and item, we only consider the last item, and the transitions probability to the item is proportional to:

$$p_u(j|i) \propto \langle v_u^{U,I}, v_i^{I,U} \rangle + \langle v_i^{I,L}, v_i^{L,I} \rangle \tag{5}$$

Where the first inner product denotes user u's static preference for item i, while the second models the similarity between item j and the last item l. And the estimated score $\hat{x}_{u,i,l}$ is equal to the probability. We choose FPMC as a basic model due to its effectiveness in modeling complex temporal data.

3.5.2. Joint FMPC

We now discuss how to build our joint model. So far, we have developed two components of our model: SNA matrix and DN sequence. To integrate them, we propose a co-factorizing method to model the complex interplay between group influence and user preference over time. We assume that: (1) A user tends to show preferences for the feedback which $\mathcal{G}_u$ have performed. (2) A user's next feedback prediction is affected by the latest feedback from both user sequence and DN sequence. Fig. 5 shows the overall of our model. We aim to capture the user's static and dynamic preference at both user level and group level. At user level, we use original user-item matrix and user sequence to model user preference, which is the same as classical FPMC methods. At group level, we model user preference based on the SNA matrix and DN sequence. This part brings network embedding information into Markov chains structure, which plays an important role in dealing with UCS problem. Moreover, the two levels can complement each other to learn static user and item factors better, due to we can learn these latent factors according to two kinds of interactions jointly. Also our model is capable of capturing the relationships between more adjacent items performed by users in a group.

$$\begin{aligned} \arg\max_{\theta} = & \sum_{(u,i,j,l)\in D_T} \ln p\left(i >_{u,l} j|\Theta\right) \\ & + \alpha \sum_{(u,i,j,l)\in D_G} ln p\left(i >_{u,l} j|\Theta\right) + \ln p(\Theta) \\ = & \sum_{(u,i,j,l)\in D_T} \ln \sigma\left(\hat{x}_{u,i,l} - \hat{x}_{u,j,l}\right) \\ & + \alpha \sum_{(u,i,j,l)\in D_G} \ln \sigma\left(\hat{y}_{u,i,l} - \hat{y}_{u,j,l}\right) - \lambda \|\Theta\|_F^2 \end{aligned} \tag{6}$$

Where D_T and D_G are the training corpuses, for each (u, i, j, l) in D_T, $u \in \mathcal{U}, i \in \mathcal{I}_u^+, i \neq j$, $\mathcal{I}_u^+$ is the items user have positive feedback in original user-item interaction matrix, and l is the last item in user u's sequence. For each (u, i, j, l) in D_G, $u \in \mathcal{U}, i \in \mathcal{I}_G^+, i \neq j$, $\mathcal{I}_G^+$ is the items user show positive feedback in SNA matrix, it is consisted of the items both user and $\mathcal{G}_u$ have consumed, and l is the last item in user's DN sequence. We use L_2 regularization to prevent over-fitting, α is a balance parameter, and its value will be evaluated in Section 4.6.

Our model jointly learns user and item factors in two levels. Krohn-Grimberghe et al. [33] proved that Factorizing the relations jointly is at least as good as the sequential approach. Moreover, Cao et al. [34] proposed CoFactor to jointly decompose the user-item interaction matrix and the item-item co-occurrence matrix. He and co-workers [34] also provided a joint model for song recommendation. Our model extends them to capture not only static preferences but also dynamic preferences. In addition, since we deal with implicit feedback, it makes sense for us to learn a personalized ranking for static preference as follows,

$$\hat{x}_{ui} > \hat{x}_{uj}, \hat{y}_{uk} > \hat{y}_{uj} \tag{7}$$

Where $\hat{x}$ is u's estimated score for item by factorizing the original user-item matrix, $\hat{y}$ is u's estimated score for item by factorizing the SNA matrix, $i \neq j$, $k \neq j$.

In sum, we integrate user embedding information and temporal information to improve the accuracy of the recommendation. It is worth to notice that our model can tackle both sparsity and user cold-start issues, due to the SNA matrix and DN sequence enrich the feedback. An additional benefit of our model is that it is more time-sensitive because we consider the temporal relationships between items from both user sequence and DN sequence. And by

Table 2
The learning algorithm of our model.

Algorithm 1. The optimization for our model
1.random initialize θ
2.Repeat
3. Repeat
4. Draw (u, i, j, l) from D_T:
5. $\theta \leftarrow \Theta + \beta\left(\left(1-\sigma\left(\hat{x}_{u,i,l}-\hat{x}_{u,j,l}\right)\right)\frac{\partial}{\partial\Theta}\left(\hat{x}_{u,i,l}-\hat{x}_{u,j,l}\right)-2\lambda\Theta\right)$
6. Until convergence
7. Repeat
8. Draw (u, i, j, l) from D_G:
9. $\theta \leftarrow \Theta + \beta\left(\alpha\left(1-\sigma\left(\hat{y}_{u,i,l}-\hat{y}_{u,j,l}\right)\right)\frac{\partial}{\partial\Theta}\left(\hat{y}_{u,i,l}-\hat{y}_{u,j,l}\right)-2\lambda\Theta\right)$
10. Until convergence
11.Until convergence or max-iteration has been reached

utilizing a joint framework, we can effectively predict the next-item for both full users and cold-start users.

3.6. Learning algorithm

In this paper, we apply stochastic gradient descent (SGD) strategy to optimize the loss function, and it draws a stochastic sample from all training instances. Since we design a joint model in this paper, we randomly sample training instance from D_T and D_G at each iteration, then the gradient descent on our model for all related parameters is performed. The complete algorithm is detailed in Table 2 and the gradients of our model towards parameters Θ are:

$$\begin{aligned}
&\frac{\partial}{\partial\theta}\left(\ln\sigma\left(\hat{x}_{u,i,l}-\hat{x}_{u,j,l}\right)-\lambda\Theta^2\right)\\
&=\left(1-\sigma\left(\hat{x}_{u,i,l}-\hat{x}_{u,j,l}\right)\right)\frac{\partial}{\partial\Theta}\left(\hat{x}_{u,i,l}-\hat{x}_{u,j,l}\right)-2\lambda\Theta\\
&\times\frac{\partial}{\partial\theta}\left(\ln\sigma\left(\hat{y}_{u,i,l}-\hat{y}_{u,j,l}\right)-\lambda\Theta^2\right)\\
&=\left(1-\sigma\left(\hat{y}_{u,i,l}-\hat{y}_{u,j,l}\right)\right)\frac{\partial}{\partial\Theta}\left(\hat{y}_{u,i,l}-\hat{y}_{u,j,l}\right)-2\lambda\Theta
\end{aligned}\tag{8}$$

The derivatives of $\hat{x}_{u,i,l}-\hat{x}_{u,j,l}$ are:

$$\begin{aligned}
\frac{\partial}{\partial v_{u,f}^{U,I}}\left(\hat{x}_{u,i,l}-\hat{x}_{u,j,l}\right)&=v_{i,f}^{I,U}-v_{j,f}^{I,U}\\
\frac{\partial}{\partial v_{i,f}^{I,U}}\left(\hat{x}_{u,i,l}-\hat{x}_{u,j,l}\right)&=v_{u,f}^{U,I}\\
\frac{\partial}{\partial v_{j,f}^{I,U}}\left(\hat{x}_{u,i,l}-\hat{x}_{u,j,l}\right)&=-v_{u,f}^{U,I}\\
\frac{\partial}{\partial v_{l,f}^{L,I}}\left(\hat{x}_{u,i,l}-\hat{x}_{u,j,l}\right)&=v_{i,f}^{I,L}-v_{j,f}^{I,L}\\
\frac{\partial}{\partial v_{i,f}^{I,L}}\left(\hat{x}_{u,i,l}-\hat{x}_{u,j,l}\right)&=v_{l,f}^{L,I}\\
\frac{\partial}{\partial v_{j,f}^{I,L}}\left(\hat{x}_{u,i,l}-\hat{x}_{u,j,l}\right)&=-v_{l,f}^{L,I}
\end{aligned}\tag{9}$$

The derivatives of $\hat{y}_{u,i,l}-\hat{y}_{u,j,l}$ are the same with $\hat{x}_{u,i,l}-\hat{x}_{u,j,l}$.

4. Experiment

In this section, we conduct experiments on the three real-world datasets to demonstrate the effectiveness of the proposed method. Especially, we conduct our experiment in a cold-start scenario to investigate the ability of our model in dealing with UCS problem. We also do extensive experiments to evaluate the ability of our method with different settings.

Table 3
The statistics of our datasets.

Statistics	Epinions	Ciao	Last.fm
#of Users	22,152	1796	1407
# of Items	296,275	16,600	12,373
# of Feedback	912,417	35,080	86,122
Density	0.014%	0.12%	0.49%
# of Trusters	18,089	2342	1892
# of Trustees	18,089	2342	1892
# of Trusts	573,420	87,380	25,434
Density	0.0017%	0.015%	0.71%

4.1. Datasets

We choose three datasets including implicit feedback, timestamp, and social relations from different domains to evaluate our model: Epinions, Ciao, and Last.fm. These datasets also vary significantly in terms with density.

Epinions. Epinions is a popular website where people can register for free and review many different types of items. It includes all actions of users and trust relationships among users on the website with the temporal information maintained. This dataset spans from January 2001 to November 2013, it is available online.[1]

Ciao. Ciao is a website where users give ratings and reviews on various items. This dataset was crawled from Ciao's official site.[2] The dataset is also available online[1].

Last.fm. Last.fm is an online music system. This dataset is collected from the site[3] which contains social network, tag, and music artist listening information ranging from 1956 to 2011. The dataset is available at[4].

For all the datasets, we remain one feedback for testing, one feedback for validating, leaving others for training. We filter the datasets to leave the users having at least 4 feedback to satisfy the requirements for training, testing and validating. The statistics of these three datasets after the above filtering process are shown in Table 3.

4.2. Evaluate metric

In this paper, we adopt two popular top-K metrics for implicit feedback recommendation to evaluate all the models: *Precision@K* and *NDCG@K*.

Precision@K. Precision measures the probability that the user is interested in the item, *Precision@K* for each user is defined as,

$$precision@K=\frac{|S(K;u)|}{K}\tag{10}$$

Where $|S(K;u)|$ indicates the set of already consumed items in the test set that appear in the top-K list.

NDCG@K. Normalized DCG (Discount Cumulative Gain), it accounts for the position of the hit by assigning higher scores to hits at top ranks. the *DCG@K* for each user is:

$$DCG@K=\sum_{1}^{K}\frac{2^{1\{u(i)=1\}}-1}{\log(i+1)}\tag{11}$$

Where $1\{\}$ is the indicator function, and $u(i)=1$ returns 1 if u has consumed the item i.

[1] http://www.cse.msu.edu/~tangjili/trust.html.
[2] http://www.ciao.co.uk/.
[3] http://www.last.fm.
[4] http://ir.ii.uam.es/hetrec2011.

Table 4
Parameters settings of respective methods.

Methods	Parameters
BPR	$\lambda_1 = \lambda_2 = \lambda_2 = 0.01, \beta_1 = \beta_2 = \beta_3 = 0.1$
SBPR	$\lambda_1 = \lambda_2 = \lambda_2 = 0.01, \beta_1 = \beta_2 = \beta_3 = 0.1, constant = 1$
FMC	$\lambda_1 = \lambda_2 = \lambda_2 = 0.01, \beta_1 = \beta_2 = \beta_3 = 0.1$
FPMC	$\lambda_1 = \lambda_2 = \lambda_2 = 0.01, \beta_1 = \beta_2 = \beta_3 = 0.1$
SPMC	$\lambda_1 = \lambda_2 = \lambda_2 = 0.01, \beta_1 = \beta_2 = \beta_3 = 0.1$
Ours	$\lambda_1 = \lambda_2 = \lambda_2 = 0.01, \beta_1 = \beta_2 = \beta_3 = 0.1, \alpha = 0.5, N_g = 10$

4.3. Baselines

To justify the effectiveness of our model, we compare our model with several representative models which could be divided into three categories: (1) models that are unaware of temporal and social information (Pop, BPR); (2) models considering either social or temporal information (SBPR, FMC, FPMC); (3) models that are both temporal-aware and socially-aware (SPMC).

Pop. This is a simple method recommending items accounts for their rank of popularity in the system.

BPR [35]. This is a sampling-based algorithm that optimizes the pairwise ranking between observed instances and sampled negative instances.

SBPR [36]. This method improves personalized ranking with social connections. The model is developed by Zhao et al. based on the simple observation that users tend to assign higher ranks to items that their friends prefer.

FMC. This model captures the possibility that a user transitions from one item to another by factorizing the item-to-item transition matrix, it is not a personalized method.

FPMC [17]. This model is the combination of Matrix Factorization and first-order Markov Chains. It is presented by Rendle which captures both temporal information and personalized user preference.

SPMC [21]. This method is proposed by He et al. in 2017. It is a state-of-the-art method which leverages feedback from sequences, as well as social interactions in the same model.

4.4. Performance analysis

4.4.1. Experiments settings

To demonstrate the effectiveness of our model proposed in this paper, we conduct our experiments on the three datasets. For a fair comparison, we randomly initialize the latent factors with a Gaussian distribution with a mean of 0 and a standard deviation of 0.01 for all the latent factor models. Gird search is applied to select the hyperparameters for each model. We experiment with the learning rates from [1, 0.1, 0.01, 0.001], regularization hyperparameters from [1, 0.1, 0.01, 0.001] and dimension from [10, 20, 30, 40, 50, 60, 70,80, 90, 100] for all the methods except Pop, and number of $\mathcal{G}_u$ from [5, 10, 15, 20, 25, 30] for our model on the validate set to tune the parameters that resulted in the best performances. Other hyperparameters are selected according to both our experiment and previous. The main parameters of respective methods are given in Table 4, where β_1, β_2, β_3 are the learning rates and λ_1, λ_2, λ_3 are regularization parameters on the Last.fm dataset, Epinions dataset, and Ciao dataset respectively. In addition, we preserve users who have at least one friend for social recommender models. We select the best results of our model and other baselines as the final result, and we conduct top-5 recommendation on the three datasets to evaluate the performance of all the baselines. We use stochastic gradient (SGD) to optimize the model. For each test instance, we randomly choose 99 negative items as negative samples. We implemented all the baselines as described in the original paper. It is worth to notice that our experiments are based on the full user scenario without a special statement. Noted the last columns in the result tables are the improvements in our model relative to the best baselines.

4.4.2. Full user analysis

We first compare the performances of all the models for the full user, where K=60. For our model, the number of $\mathcal{G}_u$ is set as 10, balance parameter α is set as 0.5. The experimental results for *NDCG*@5 and *Precision*@5 are summarized in Table 5. From the results we can make observations and analysis as follows:

Generally, our method is superior over all the baselines on both sparse and dense datasets according to the two metrics, especially has a significant improvement on Ciao dataset. Through this comparison, we find SPMC is the strongest model among the baselines due to it integrates both social information and temporal information into the recommendation. Our model gain 20% *NDCG* and 21.2% *Precision* improvements on average against SPMC model, this is as expected because our model utilizes user embedding to select Network Neighbors, which provides more meaningful information than explicit social relations for the recommendation.

When comparing with other baselines, we find that: (1) Our model shows substantial improvements over Pop, which results from Pop is based on item popularity without considering any global user preference information. (2) Our model also has a significant improvement than BPR, this is because BPR is a simple model which considers only static preference of the users without any other side information. (3) Obviously, our model has a much better performance than FPMC and FMC. Despite FMC and FPMC take temporal information into consideration, they only consider a single sequence and are lack of other auxiliary information such as social information. (4) Our model outperforms SBPR, SBPR takes the social relations into the recommendation to design a ranking method which provides a better recommendation, but it ignores temporal information thus has worse performance.

From the perspective of the baselines, we find that: (1) When comparing BPR with SBPR, we find SBPR performs much better than BPR on both Last.fm and Epinions datasets, which shows social information can help improving recommendation. (2) When comparing BPR with FPMC, we can notice FPMC outperform BPR on Last.fm and Epinions datasets, and similar results can be seen on Ciao datasets, which demonstrates that it is crucial to integrate temporal information into the recommendation. When comparing FMC and FPMC, we can see FPMC performs better than FMC on the three datasets. In addition, BPR outperforms FMC on Ciao and Last.fm datasets, suggesting the importance of personalization. (3) When comparing SPMC and SBPR with FPMC, we find SPMC performs best among them by modeling moth social influence and sequential influence. FPMC and SBPR only consider either temporal information or social information so that their abilities are curbed.

In conclusion, our model beats all the baselines on the three datasets, which indicates combining network embedding information and temporal information is useful for the recommendation. By integrating this information a joint FPMC model, our model outperforms all the baselines and leads to superior accuracy in implicit feedback recommendation.

4.4.3. Cold-start analysis

We further investigate the performances of all the model in dealing with UCS problem. Traditional methods usually choose users whose feedback is fewer than 5 as cold-start users. Since we need at least 4 feedback in our model, leading to less cold-start users achieved, thus we preserve N recent feedback for each user to simulate the cold-start environment, where N is the threshold value. The previous work [21] adopted this protocol. We conduct

Table 5
NDCGs and Precisions of all models on three datasets.

Dataset	User	Metric	Pop	BPR	SBPR	FMC	FPMC	SPMC	Ours	Imp
Last.fm	Full	*NDCG*@5	0.3019	0.4462	0.5089	0.4009	0.4647	0.5259	**0.5512**	+4.8%
	user	*Precision*@5	0.2674	0.4064	0.4748	0.3549	0.4247	0.4824	**0.5084**	+5.3%
Epinions	Full	*NDCG*@5	0.2833	0.3113	0.3701	0.3381	0.3466	0.4100	**0.4284**	+4.5%
	user	*Precision*@5	0.2560	0.2812	0.3392	0.2892	0.3084	0.3779	**0.3969**	+5.0%
Ciao	Full	*NDCG*@5	0.1332	0.1603	0.1633	0.1483	0.1691	0.1963	**0.2918**	+50.7%
	user	*Precision*@5	0.1142	0.1390	0.1410	0.1177	0.1473	0.1713	**0.2631**	+53.5%

Table 6
NDCGs and Precisions for cold-start users with different threshold value.

Dataset	User	Metric	Pop	BPR	SBPR	FMC	FPMC	SPMC	Ours	Imp
Last.fm	Cold	*NDCG*@5	0.2741	0.3048	0.2531	0.2835	0.3169	0.3258	**0.4445**	+11.8%
	5	*Precision*@5	0.2449	0.2891	0.2335	0.2537	0.2991	0.2976	**0.4158**	+39.0%
	Cold	*NDCG*@5	0.3001	0.4251	0.3557	0.3388	0.3859	0.4031	**0.4641**	+15.1%
	10	*Precision*@5	0.2665	0.3992	0.3296	0.2972	0.3635	0.3705	**0.4316**	+8.1%
Epinions	Cold	*NDCG*@5	0.2545	0.1710	0.1917	0.2182	0.1876	0.2587	**0.3478**	+34.4%
	5	*Precision*@5	0.2356	0.1522	0.1665	0.1852	0.1682	0.2345	**0.3115**	+32.2%
	Cold	*NDCG*@5	0.3148	0.3133	0.2767	0.2968	0.2897	0.2972	**0.3335**	+5.9%
	10	*Precision*@5	0.2891	0.2852	0.2482	0.2569	0.2631	0.2694	**0.3007**	+4.0%
Ciao	Cold	*NDCG*@5	0.1214	0.0816	0.1026	0.1090	0.0831	0.1420	**0.2258**	+59.0%
	5	*Precision*@5	0.1059	0.0707	0.0887	0.0854	0.0725	0.1272	**0.2057**	+61.7%
	Cold	*NDCG*@5	0.1502	0.1324	0.1562	0.1283	0.1389	0.1647	**0.2490**	+51.1%
	10	*Precision*@5	0.1292	0.1167	0.1379	0.1049	0.1238	0.1413	**0.2254**	+59.5%

the experiments on three datasets with the threshold values as 5 and 10, respectively. The performances of all the models in dealing with UCS problem with different threshold values are summarized in Table 6. Apparently, our model has a significant improvement than all the baselines varying the threshold value, especially when the threshold value is set as 5, demonstrating our model is good at deal with UCS problem.

From the results shown we also can see: (1) On Last.fm dataset and Ciao dataset, FPMC and SPMC have better performances than other baselines, and SPMC is the strongest among all the baselines, suggesting temporal information and social information are both essential elements for improving UCS problem. (2) On Epinions dataset, we find SPMC still performs best when the threshold value is 5, but BPR and Pop achieve better performances when the threshold value is 10, which is presumably due to the influences of social information and temporal information on this dataset are not effective in cold-start scenario, and we can infer that popularity is also an important factor for UCS problem.

To summarize, our model achieves valid results on three datasets in the cold-start scenario, which is apparently due to Network Neighbors are more effective than explicit social relations in user cold-start issues, and we model dynamic preferences, which is also crucial for this problem. Compared with other baselines, our model shows significant improvements on Ciao dataset and shows better performances on Epinions dataset and Last.fm dataset. Hence, we can say our model can effectively deal with UCS problem.

4.4.4. Trust cold-start analysis

As mentioned before, cold-start users usually have fewer social links in the social network. In this paper, we call these users as trust cold-start users. To explore the performances of all the models for trust cold-start users, we conduct the experiments on the three datasets for users with fewer than 5 and 10 social links based on the cold-start scenario, where the threshold value of cold-start users is 5. The results are summarized in Table 7. From the results we can see that our model achieves significant improvements than other baselines, demonstrating the effectiveness of Network Neighbors information for dealing with trust cold-start users. Among the baselines, SPMC and FPMC perform much better than other baselines, whereas SBPR performs worse than others, which is different from the previous results in Table 6, suggesting the abilities of social methods for UCS problem are limited by sparse social networks when social links are brave. It also proves the importance of temporal information in dealing with this problem.

4.4.5. Dimension of latent feature

To analyze the effect of dimension K value on our model, we first fix the number of $\mathcal{G}_u$ as 10, balance parameter α as 0.5, then we conduct the experiment with different K values from [10, 20, 30, 40, 50, 60, 70, 80, 90, 100] for all the models except for Pop. Fig. 6 illustrates the two metrics with variations in K values on the three datasets. As the results showed, a larger K value always leads to a better performance for our model which indicates our model typically benefits from larger values of K, also our model outperforms all the baselines with different K values. Based on the performances we can see: (1) On Epinions dataset, the results are more stable than other two datasets, which is apparently due to Epinions dataset is larger than Ciao and Last.fm datasets. (2) On Ciao dataset, our model has more significant improvements compared with other baselines, which is probably due to $\mathcal{G}_u$ plays a more important role on Ciao dataset. But the curve is not as stable as it on Epinions dataset, because Ciao dataset is smaller than Epinions dataset. (3) On Last.fm dataset, there are also increasing trends for all the models with the increasing of K values, but the increasing trends are not significant when K=30, because Last.fm dataset is a smaller dataset than Ciao dataset and Epinions dataset and lager K values will result in over-fitting.

Generally, from the curves we can see that a larger K usually leads to better performances of all the models on the three datasets, this is mainly due to higher latent dimensions contains more information. However, the increasing trend of our model is not significant as before after K=60 on both Ciao and Epinions datasets, and most baselines achieve good performances when K=60. Therefore, based on the analysis on the three datasets for all models, we set K=60 as the default setting in this paper to reduce train complexity and avoid over-fitting.

Table 7
NDCGs and Precisions for trust cold-start users with different threshold value.

Dataset	User	Metric	Pop	BPR	SBPR	FMC	FPMC	SPMC	Ours	Imp
Last.fm	Cold	*NDCG@5*	0.1972	0.2465	0.2543	0.2078	0.2586	0.2517	**0.3450**	+33.4%
	5	*Precision@5*	0.1714	0.2324	0.2335	0.1746	0.2434	0.2321	**0.3179**	+30.6%
	Cold	*NDCG@5*	0.2187	0.2569	0.2850	0.2332	0.2822	0.2782	**0.3668**	+29.9%
	10	*Precision@5*	0.1913	0.2422	0.2616	0.2052	0.2651	0.2565	**0.3393**	+32.2%
Epinions	Cold	*NDCG@5*	0.2559	0.1744	0.2048	0.2327	0.2586	0.2775	**0.3122**	+12.5%
	5	*Precision@5*	0.2366	0.1568	0.1758	0.1969	0.2434	0.2531	**0.2816**	+11.2%
	Cold	*NDCG@5*	0.2604	0.1761	0.2668	0.2234	0.2822	0.2337	**0.3151**	+11.6%
	10	*Precision@5*	0.2417	0.1578	0.2386	0.1882	0.2651	0.2102	**0.2840**	+7.1%
Ciao	Cold	*NDCG@5*	0.1487	0.1044	0.1026	0.1261	0.1200	0.1751	**0.2672**	+52.5%
	5	*Precision@5*	0.1269	0.0931	0.0887	0.0992	0.1069	0.1524	**0.2359**	+54.7%
	Cold	*NDCG@5*	0.1453	0.0989	0.1429	0.1095	0.0876	0.1803	**0.2523**	+39.9%
	10	*Precision@5*	0.1238	0.0865	0.1262	0.0839	0.0792	0.1641	**0.2270**	+38.3%

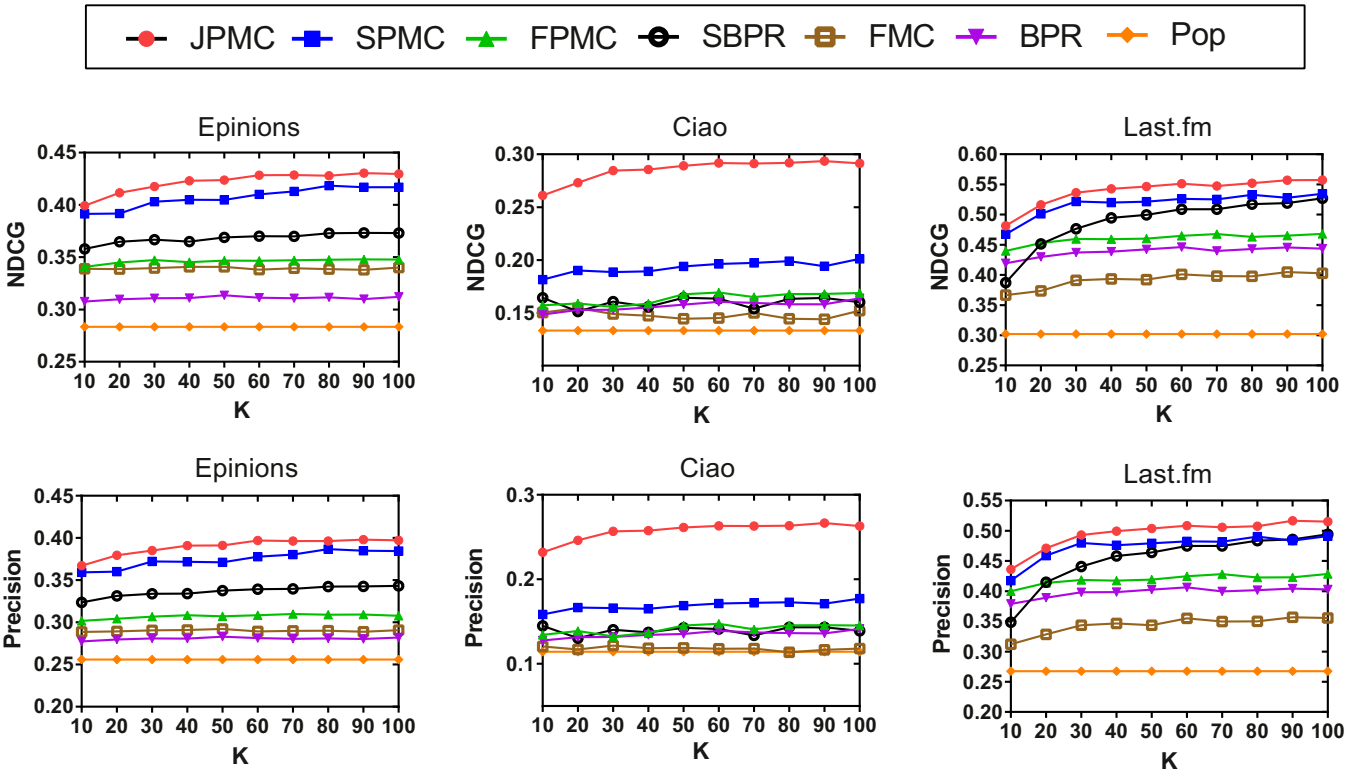


Fig. 6. The NDCGs and Precisions of all the models with different K values.

4.5. Different methods for selecting Network Neighbors

We further conduct the experiments to investigate the effectiveness of different methods to achieve Network Neighbors. First, We only send social relations into Node2vec model to achieve Network Neighbors, we call this method JPMC-os. Fig. 7 shows the results of our model on the three datasets. From the results we can notice: On Epinions and Ciao datasets, JPMC has a significant improvement than JPMC-os with different K values. Despite the improvement is not significant on Last.fm dataset, JPMC still outperforms than JPMC-os. It demonstrates that added links are crucial for selecting more suitable Network Neighbors. We also try to select Network Neighbors with high PMI value without network embedding process, which is called as JPMC-PMI. From the results we can see that JPMC-PMI performs worst among the three kinds of methods on the three datasets. It demonstrates that network embedding method is irreplaceable, due to it can mine deep connection information among users.

4.5.1. Number of Network Neighbors

We also analyze the performances of our model with the different number of Network Neighbors N_g. To investigate when our method makes a better recommendation with different N_g values, we conduct the experiment on all the datasets where the number of Network Neighbors is from the range of [5, 10, 15, 20, 25, 30], we fix K=60, balance parameter $\alpha = 0.5$. Fig. 8 illustrates the results by varying the value of N_g on the two metrics. It is apparent that the curves of two metrics on the three datasets are much alike. We observe from the results that our model performs best when the number of N_g is 10 on Epinions dataset and Ciao dataset, and 5 on Last.fm dataset. There is a decreasing trend when the number of N_g is larger than 10 on both Ciao and Epinions datasets, which is presumably due to top 10 users have much more similarities with the user, while more than 10 users will bring more useless information which will mislead the recommendation. Therefore, we set the default setting of the number of N_g as 10 on the three datasets.

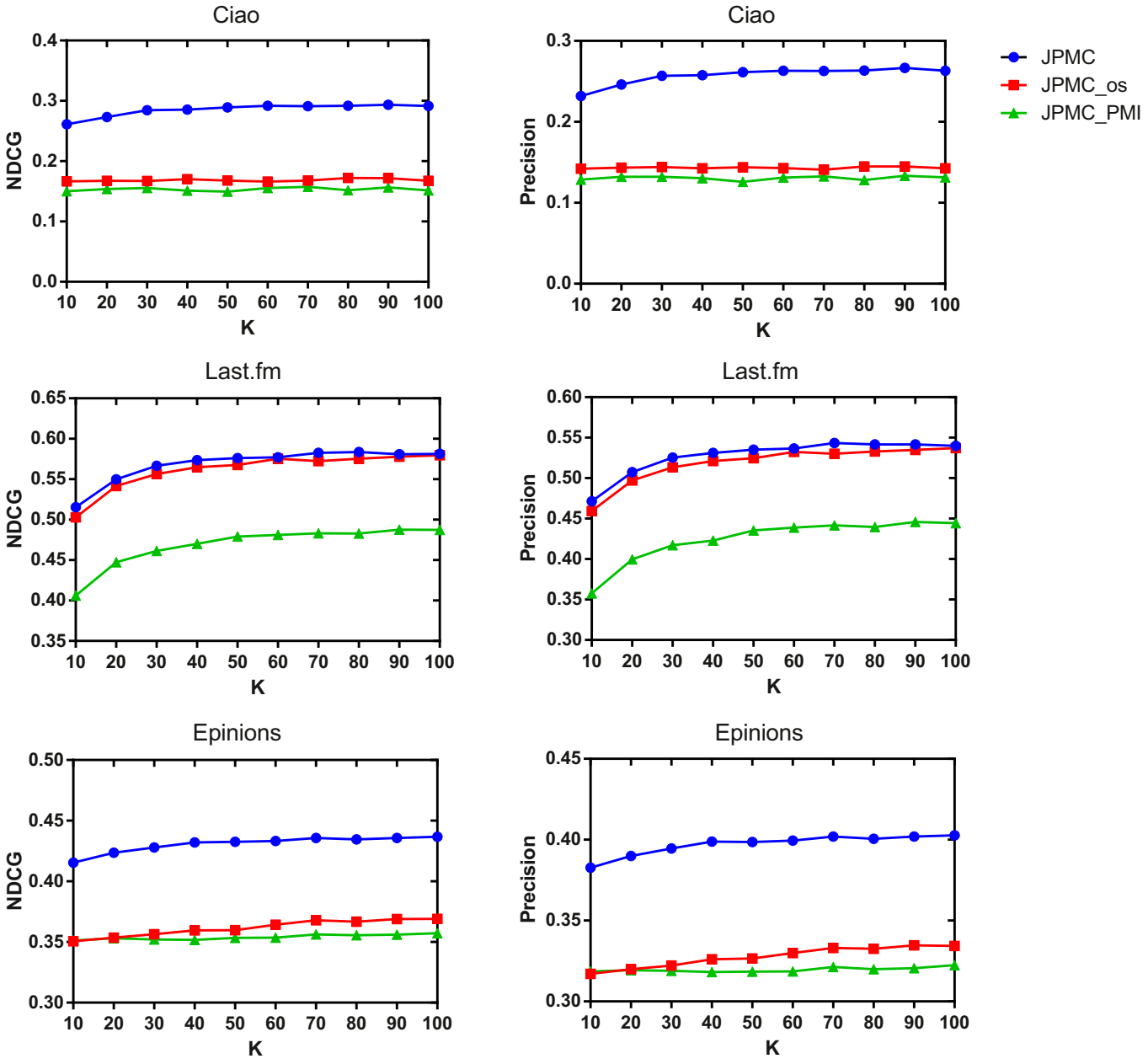


Fig. 7. The NDCGs and Precisions of three methods with different methods for selecting Network Neighbors.

4.6. Influence of balance parameter α

We also analyze the influence of balance parameter α on our model, we conduct the experiments with different α values from [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1], we fix K as 60, N_g as 10. Fig. 9 shows the results of two metrics on the three datasets. From the figure we can see: On Ciao dataset, our model achieves best performance when α is 0.5, and the performance begins to drop when the α value is larger than 0.5. On Last.fm dataset, the performance of our model continues to rise until α is 0.5, when α is larger than 0.5, the trend of rise is not apparent as before. On Epinions dataset, the trend of two metrics is similar to that on Ciao dataset, our model achieves best NDCG when α is 0.6, while achieves best precision when α is 0.5. The results indicate our model performs best when $\alpha = 0.5$ in most cases, thus we set the value of α as 0.5 in our model to achieve best performance.

4.6.1. Converge

At last, we investigate the convergence of our model on the different datasets. Fig. 10 demonstrates the *NDCG* and *Precision* of our model on all datasets in 200 iterations. Noted that at each training iteration we choose all positive feedback in the training set. We also notice that our model converges fast on Ciao and Epinions datasets within 20 iterations, fewer than 80 iterations on Last.fm dataset. This fact indicates our model can achieve promising convergence on different datasets. From the curves we can see: Our model performs best on Last.fm dataset among the three datasets, but worst on Ciao dataset, which is mainly because the user-item interactions and social relations on Last.fm dataset are denser than other two datasets, while Ciao dataset is a small and sparse dataset. On Epinions dataset, the curve is more stable because it is a larger dataset than the other two datasets.

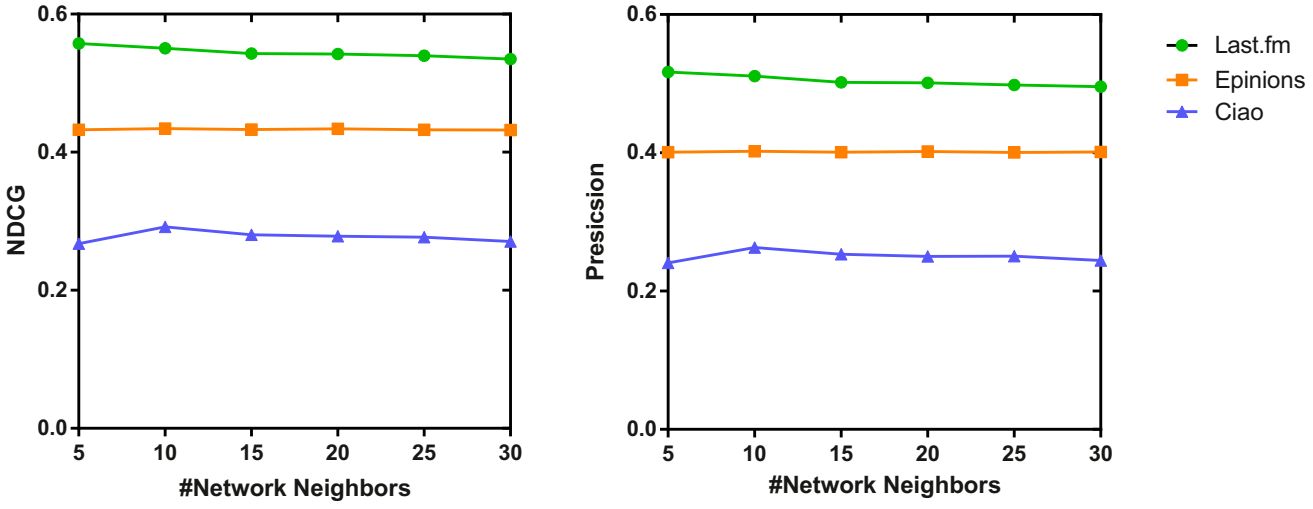


Fig. 8. The NDCGs and Precisions of all the models with different number of Network Neighbors.

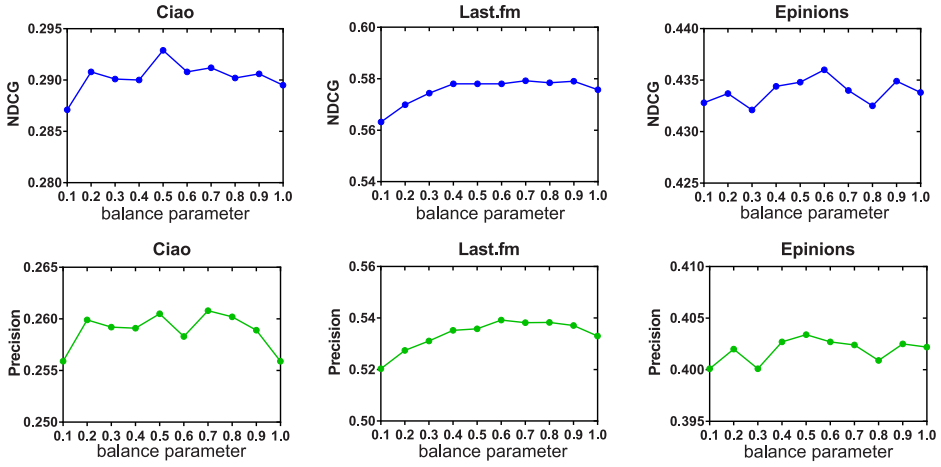


Fig. 9. The NDCGs and Precisions of our model with different α values.

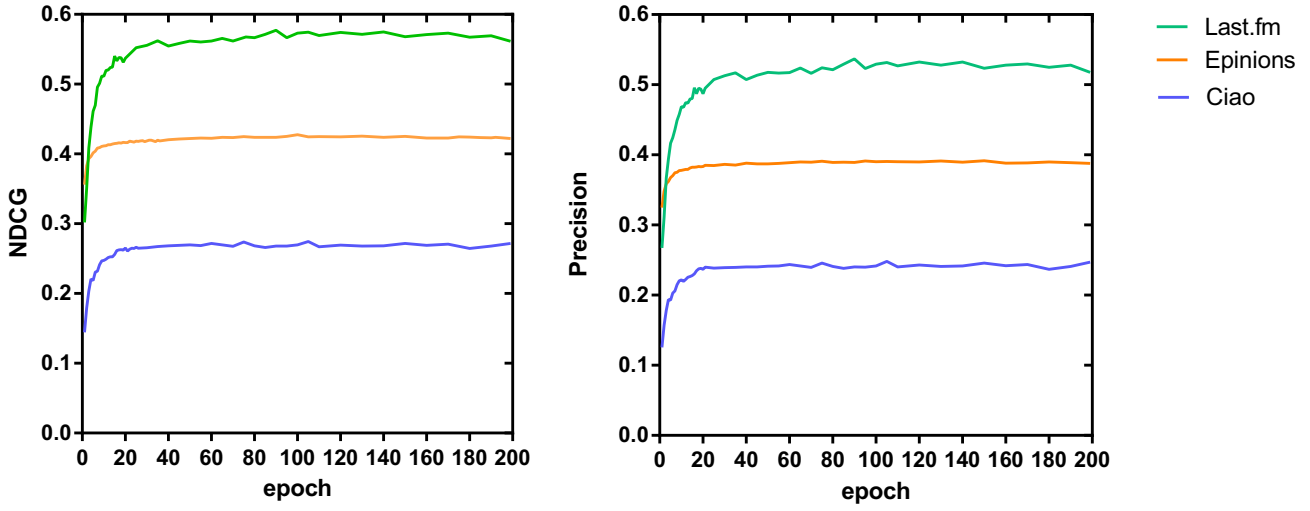


Fig. 10. The NDCGs and Precisions of our model with the increasing of epoch value.

5. Conclusion and future

In the paper, we bring forward a novel joint model combining social network embedding information and temporal information, which offers significant advantages both in terms of improving the recommendation quality and in dealing with the UCS problem as compared to existing work. We first pre-train the representations of all the users by Node2vec method, then we select Network Neighbors with high similarities. To integrate it into recommendation, we secondly construct SNA matrix and DN sequence according to the Network Neighbors. We further propose a joint model under the FPMC framework to capture both user level and

group level preferences. Our model can properly utilize social and temporal information to predict the next-item of the user. Especially, we can deal with cold-start users with less social links. In the future, we will identify deeper information from social network to effectively solve user cold-start issues. In addition, we will explore the impact of temporal information on user consumption behavior to build an incremental recommender system.

Declaration of Competing Interest

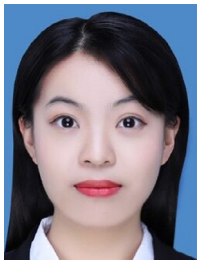
All authors have participated in (a) conception and design, or analysis and interpretation of the data; (b) drafing the article or revising it critically for important intellectual content; and (c) approval of the final version. This manuscript has not been submitted to, nor is under review at, another journal or other publishing venue. The authors have no affiliation with any organization with a direct or indirect financial interest in the subject matter discussed in the manuscript.

Acknowledgments

This work is sponsored by the National Natural Science Foundation of China (61976103 and 61872161), the Nature Science Foundation of Jilin Province (20180101330JC), the Scientific and Technological Development Program of Jilin Province (20190302029GX).

References

[1] Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, Computer 42 (8) (2009) 30–37.
[2] Y. Koren, Factorization meets the neighborhood: a multifaceted collaborative filtering model, in: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2008, pp. 426–434.
[3] R. Pan, Y. Zhou, B. Cao, N.N. Liu, R. Lukose, M. Scholz, Q. Yang, One-class collaborative filtering, in: Proceedings of the Eighth IEEE International Conference on Data Mining, IEEE, 2008, pp. 502–511.
[4] M.J. Pazzani, D. Billsus, Content-based recommendation systems, in: The Adaptive Web, Springer, 2007, pp. 325–341.
[5] T. Chen, L. Hong, Y. Shi, Y. Sun, Joint Text Embedding for Personalized Content-Based Recommendation, arXiv:1706.01084 (2017).
[6] Y. Zhu, J. Lin, S. He, B. Wang, Z. Guan, H. Liu, D. Cai, Addressing the item cold-start problem by attribute-driven active learning, IEEE Trans. Knowl. Data Eng. (2019).
[7] H. Ma, H. Yang, M.R. Lyu, I. King, Sorec: social recommendation using probabilistic matrix factorization, in: Proceedings of the 17th ACM Conference on Information and Knowledge Management, ACM, 2008, pp. 931–940.
[8] H. Ma, I. King, M.R. Lyu, Learning to recommend with social trust ensemble, in: Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2009, pp. 203–210.
[9] M. Jamali, M. Ester, A matrix factorization technique with trust propagation for recommendation in social networks, in: Proceedings of the Fourth ACM Conference on Recommender Systems, ACM, 2010, pp. 135–142.
[10] A.J. Chaney, D.M. Blei, T. Eliassi-Rad, A probabilistic model for using social networks in personalized item recommendation, in: Proceedings of the 9th ACM Conference on Recommender Systems, ACM, 2015, pp. 43–50.
[11] T. Zhao, J. McAuley, I. King, Leveraging social connections to improve personalized ranking for collaborative filtering, in: Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, ACM, 2014, pp. 261–270.
[12] X. Wang, S.C. Hoi, M. Ester, J. Bu, C. Chen, Learning personalized preference of strong and weak ties for social recommendation, in: Proceedings of the 26th International Conference on World Wide Web, International World Wide Web Conferences Steering Committee, 2017, pp. 1601–1610.
[13] A. Grover, J. Leskovec, node2vec: Scalable feature learning for networks, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2016, pp. 855–864.
[14] W.Y. Zhang C Yu L, Collaborative user network embedding for social recommender systems, in: Proceedings of the 2017 SIAM International Conference on Data Mining, Society for Industrial and Applied Mathematics, 2017, pp. 381–389.
[15] C. Shi, B. Hu, W.X. Zhao, S.Y. Philip, Heterogeneous information network embedding for recommendation, IEEE Trans. Knowl. Data Eng. 31 (2) (2019) 357–370.
[16] P. Goyal, E. Ferrara, Graph embedding techniques, applications, and performance: a survey, Knowl.-Based Syst. 151 (2018) 78–94.
[17] S. Rendle, C. Freudenthaler, L. Schmidt-Thieme, Factorizing personalized Markov chains for next-basket recommendation, in: Proceedings of the 19th International Conference on World Wide Web, ACM, 2010, pp. 811–820.
[18] R. He, J. McAuley, Fusing similarity models with Markov chains for sparse sequential recommendation, in: Proceedings of the 16th IEEE International Conference on Data Mining (ICDM), IEEE, 2016, pp. 191–200.
[19] R. He, C. Fang, Z. Wang, J. McAuley, Vista: a visually, socially, and temporally-aware model for artistic recommendation, in: Proceedings of the 10th ACM Conference on Recommender Systems, ACM, 2016, pp. 309–316.
[20] C.H. Liu, J. Xu, J. Tang, J. Crowcroft, Social-aware sequential modeling of user interests: a deep learning approach, IEEE Trans. Knowl. Data Eng. (2018).
[21] C. Cai, R. He, J. McAuley, Spmc: socially-aware personalized Markov chains for sparse sequential recommendation, Proceeding of the 2017 IJCAI'17, 2017.
[22] K. Zhou, S.-H. Yang, H. Zha, Functional matrix factorizations for cold-start recommendation, in: Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2011, pp. 315–324.
[23] A.M. Rashid, I. Albert, D. Cosley, S.K. Lam, S.M. McNee, J.A. Konstan, J. Riedl, Getting to know you: learning new user preferences in recommender systems, in: Proceedings of the 7th International Conference on Intelligent User Interfaces, ACM, 2002, pp. 127–134.
[24] A.M. Rashid, G. Karypis, J. Riedl, Learning preferences of new users in recommender systems: an information theoretic approach, ACM SIGKDD Explor. Newslett. 10 (2) (2008) 90–100.
[25] H.-N. Kim, A.-T. Ji, I. Ha, G.-S. Jo, Collaborative filtering based on collaborative tagging for enhancing the quality of recommendation, Electron. Commerce Res. Appl. 9 (1) (2010) 73–83.
[26] W. Pan, L. Chen, GBPR: group preference based Bayesian personalized ranking for one-class collaborative filtering, in: Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence, 2013, pp. 2691–2697.
[27] S. Sedhain, A.K. Menon, S. Sanner, L. Xie, D. Braziunas, Low-rank linear cold-start recommendation from social data, in: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, 2017, pp. 1502–1508.
[28] Z. Shi, W. Zuo, W. Chen, L. Yue, J. Han, L. Feng, User relation prediction based on matrix factorization and hybrid particle swarm optimization, in: Proceedings of the 26th International Conference on World Wide Web Companion, International World Wide Web Conferences Steering Committee, 2017, pp. 1335–1341.
[29] Q. Song, J. Cheng, T. Yuan, H. Lu, Personalized recommendation meets your next favorite, in: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, ACM, 2015, pp. 1775–1778.
[30] N. Natarajan, D. Shin, I.S. Dhillon, Which app will you use next?: Collaborative filtering with interactional context, in: Proceedings of the 7th ACM Conference on Recommender Systems, ACM, 2013, pp. 201–208.
[31] J. Chen, C. Wang, J. Wang, A personalized interest-forgetting Markov model for recommendations, in: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, 2015, pp. 16–22.
[32] O. Levy, Y. Goldberg, Neural word embedding as implicit matrix factorization, in: Proceedings of the Advances in Neural Information Processing Systems, 2014, pp. 2177–2185.
[33] A. Krohn-Grimberghe, L. Drumond, C. Freudenthaler, L. Schmidt-Thieme, Multi--relational matrix factorization using Bayesian personalized ranking for social network data, in: Proceedings of the Fifth ACM International Conference on Web Search and Data Mining, ACM, 2012, pp. 173–182.
[34] D. Cao, L. Nie, X. He, X. Wei, S. Zhu, T.-S. Chua, Embedding factorization models for jointly recommending items and user generated lists, in: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, 2017, pp. 585–594.
[35] S. Rendle, C. Freudenthaler, Z. Gantner, L. Schmidt-Thieme, BPR: Bayesian personalized ranking from implicit feedback, in: Proceedings of the Twenty–Fifth Conference on Uncertainty in Artificial Intelligence, AUAI Press, 2009, pp. 452–461.
[36] T. Zhao, J. McAuley, I. King, Leveraging social connections to improve personalized ranking for collaborative filtering, in: Proceedings of the 23rd ACM International Conference Information and Knowledge Management, ACM, 2014, pp. 261–270.



Yijia Zhang received the B.Sc. degree from the College of Computer Science and Technology, Jilin University, Changchun, China, where she is pursuing the Ph.D. degree with the Key Laboratory of Symbol Computation and Knowledge Engineering, Ministry of Education. Her research interests include recommender system, deep learning, data mining and machine learning.

Zhenkun Shi received his B.Sc. degree in the College of Computer Science and Technology, Agricultural University of Hebei, China. He is pursuing the Ph.D. degree in Key Laboratory of Symbol Computation and Knowledge Engineering of the Ministry of Education, Jilin University, Changchun, China. He is also a joint Ph.D student of the University of Queensland. His research interests include clinical data mining, social computing, deep learning.

Wanli Zuo is a Professor at Jilin University. He received the B.Sc., M.S. and Ph.D. degrees in the College of Computer Science and Technology, Jilin University, Changchun, China. He has published more than 160 journal papers and conference papers. His research interests include data mining, information retrieval, natural language processing, and machine learning, etc.

Lin Yue received the B.S. and M.S. degrees from Northeast Normal University, the Ph.D. degree from the Key Laboratory of Symbolic Computation and Knowledge Engineering, Ministry of Education, Jilin University, China, all in computer science, and the joint Ph.D. degree from the University of Queensland, Australia. She is currently a Post-Doctoral Research Fellow with Northeast Normal University. Her current main research interests include data mining, natural language processing, and machine learning.

Shining Liang received the B.Sc. degree from the College of Computer Science and Technology, Jilin University, Changchun, China, where he is currently pursuing the Ph.D. degree with the Key Laboratory of Symbol Computation and Knowledge Engineering, Ministry of Education. His research interests include natural language processing, causality mining, deep learning and clinical data mining.

Xue Li received the Ph.D. degree in information systems from the Queensland University of Technology, in 1997. He is a full professor in the School of Information Technology and Electrical Engineering, University of Queensland, Australia. He is an adjunct professor with Chongqing University, China. His research areas are big data analytics, pattern recognition, and intelligent information systems. He is a member of the ACM and the IEEE.