Accepted Manuscript

SDBPR: Social distance-aware Bayesian personalized ranking for recommendation

Feng Zhao, Yu Shen, Xiangyu Gui, Hai Jin

PII: S0167-739X(18)32047-8
DOI: https://doi.org/10.1016/j.future.2018.12.052
Reference: FUTURE 4673

To appear in: *Future Generation Computer Systems*

Received date : 30 August 2018
Revised date : 7 December 2018
Accepted date : 20 December 2018

SDBPR: Social Distance-aware Bayesian Personalized Ranking for Recommendation

Feng Zhao, Yu Shen, Xiangyu Gui, Hai Jin

Services Computing Technology and System Lab & Big Data Technology and System Lab & Cluster and Grid Computing Lab

School of Computer Science and Technology, Huazhong University of Science and Technology, China

Abstract

Recommendation systems recommend new items to users. Because training data contain only binary forms of implicit feedback in many cases, such as in IoT and IoV, one-class collaborative filtering which can be solved by using rating-based methods to estimate the numeric scores of items or ranking-based methods based on the preferences of each user for items, must be addressed. In addition, because of the sparsity of such data, ranking-based methods are often preferred over rating-based methods when only implicit feedback is available. Social information has recently been used to improve the accuracy of rankings. Traditional approaches simply consider the direct friends of users in a social network, but this process fails to consider the propagation of influence along connections in the social network and cannot reveal the complex graph structure of the social network. In this paper, a novel social distance-aware Bayesian personalized ranking model, called SDBPR, is proposed to generate more accurate recommendations. SDBPR uses a random walk to travel the social network and then makes pairwise assumptions about the ranking order based on the distance between users along the random walk. The experimental results on two real datasets show that the proposed approaches significantly outperform the baseline approaches in terms of ranking prediction.

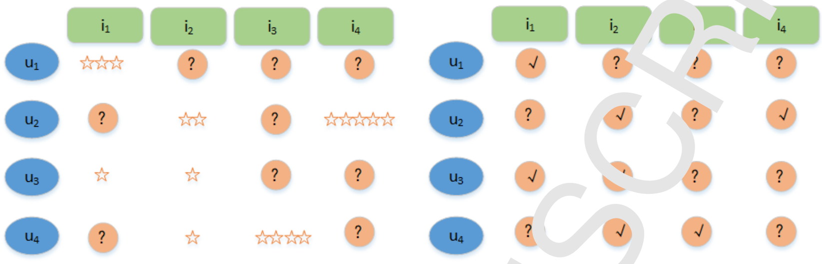
Keywords: Recommendation, Bayesian personalized ranking, Social similarity, Random walk

1. Introduction

Recommendation systems are very popular in people's daily life and are widely used by many Internet services. Amazon and eBay recommend products when users are shopping online, Netflix and YouTube recommend movies to their customers, and the Internet of Vehicles (IoV) recommends automatic driving routes or locations for parking. One important aspect of recommendations is that they must be personalized, which means that the recommendation system must recommend different items in the context of a given user. For example, in the path planning problem for the IoV, a path must be planned from the beginning to the end for users. Traditional methods consider only information about the path itself, such as the length of the path and road conditions. If we can collect information about the historical paths selected by drivers and some user path selection history closely related to the user and then use this information to estimate the driver's preference for certain paths, we can choose the path from the set of potential paths that is most consistent with the driver's preference. Then, as in the automatic parking problem, we need to recommend the best parking location for the user; we can consider not only the driving distance but also inherent parking location attributes, such as price and services. Furthermore, we can provide even more accurate recommendations if we can combine this information with friends' preferences for parking spaces to estimate the user's preference for a certain parking location.

Personalized recommendation systems first track user behavior, which reflects user preferences, then estimate the preference of a user from user feedback, and finally give a personalized ranking of items for the user. This task can be performed by using rating-based methods to estimate the numeric score of an item or by using ranking-based methods to estimate the relative preferences of items for each user. Many proposed rating-based methods, such as k-nearest neighbor (kNN) collaborative filtering [1] and matrix factorization [2], have achieved good performance when explicit feedback, such as rating scores (Figure 1a), is available. However, explicit feedback may be difficult to track in real-world scenarios. Implicit feedback (Figure 1b), such as click actions, number of views and purchase behavior, is more easily tracked than explicit feedback because it does not interfere with normal user actions. Many rating-based methods often fail when only implicit feedback is available. However, some rating methods can be adapted to utilize implicit feedback[3]. In general, ranking-based methods are preferred over rating-based methods when only implicit feedback is available. Implicit feedback often exists in binary form, using "yes" or "no" to represent user actions. We focus on the issue of recommendation using this form of implicit feedback, which is referred to as one-class collaborative filtering [4].

With the growth of online social networks, many recommendation systems require that users sign in to access their services. Users of a recommendation system can also constitute a social network (Figure 2). For example, through social relationships in IoV, it becomes possible to automatically park at the nearest friend's parking lot. Recent studies have also shown that social information can be used to improve the accuracy of recommendation systems because users in a social network often have interests that are similar to those of their friends. A user's preferences can be estimated both from the user's own behavior and the behavior of the user's friends. When a user's feedback is scarce, feedback from the user's friends can provide a large volume of information that can be used to infer the user's preferences. This process is especially important for a user who has only recently joined the system. Moreover, the strength of social ties between users is important because different friends may have different influ-



(a) Explicit feedback (b) Implicit feedback

Figure 1: Two types of feedback. In explicit feedback, a pentagram represents the rating score between 1 and 5 that user u gives item i. In implicit feedback, a tick represents a "yes" action of user u, such as click actions, view times and purchase behavior, with respect to item i. The question mark in both figures indicates that no feedback is tracked for user u with respect to item i. Recommendation models aim to predict the missing values represented by question marks.

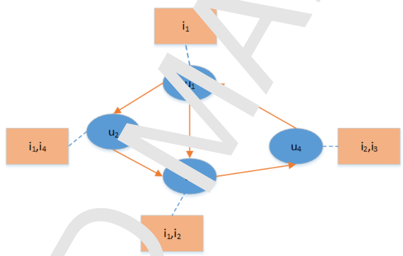


Figure 2: An example of a social network in a recommendation system. Each user is represented by an ellipse. The line pointing from u_i to u_j indicates that u_i trusts u_j. The content in the rectangle represents items consumed by the user who is connected to the rectangle by a dotted line.

ences when a user chooses to acquire new items. In [5], Gee et al. investigated the influence of strong ties and weak ties when users found jobs on Facebook's social network. Weak ties are important in recommendation tasks because the feedback from weak ties provides slightly more novelty to users than strong ties. Based on previous work, we extend the definition of weak ties to include weak relations by considering information from the friends of friends. The information from multistep social connections often has slightly weak relations with the original user but is also useful for recommendations.

The goal of this paper is to build a ranking-based model for recommendations based on the social ties between users when only implicit feedback is available. Many rating-based methods propose to consider social connections when estimating the rating scores of items based on the premise that any two users who are friends in a social network should exhibit similar rating patterns.

Since explicit feedback is not accessible in many practical applications, we do not consider using rating-based methods that rely on explicit feedback. Rating-based methods often fail to utilize implicit feedback because of the inherent data sparsity, but ranking-based methods can overcome this issue. The most popular ranking-based method is the Bayesian personalized ranking (BPR) method proposed by Steffen et al. in [6]. BPR assumes that a user prefers items that have been consumed by the user over items that have not been consumed by the user. In [7], Zhao et al. proposed social Bayesian personalized ranking (SBPR), which incorporates social connections into BPR by further assuming that a user prefers items consumed by their friends over items not consumed by their friends. Moreover, the recommendation model should use a strategy to compute the similarity of the user. The most common approach is mapping each user and item into a latent vector space of fixed dimension. Each dimension of the latent vector represents one aspect of the user and item. For instance, one factor may represent that the user is a food lover or that the item is a type of fruit. Then, the preference of the user for an item is represented by the user and item vectors.

Although many approaches incorporate social information to estimate user preferences, most consider only the direct friends of users. These approaches fail to consider the complex underlying graph structure of the social network, which is useful for examining the item information spread among users. The first challenge we faced involves uncovering the complex graph structure of a large social network. In this paper, we adopt the random walk [8] strategy to travel the social network along social connections. This strategy mimics the item information propagation process in a social network. Random walk has proven to be effective at capturing the information hidden in a graph structure. Because we use a ranking-based method to make recommendations, the second challenge that we faced involves making a reasonable assumption about the preference order of items for each user. We are motivated by the assumptions used in SBPR. To incorporate weak relationships between users, we use the distance along the random walk path between users as a metric to estimate the user's preference order for two items. The distance is weighted by the similarity between two consecutive users in the path. Here, we adopt Jaccard's coefficient [9] to measure the similarity between users. Jaccard's coefficient for two users is defined as the number of common friends divided by the size of the union of the friends. Jaccard's coefficient is widely used in the literature and has proven to be a useful metric for denoting the similarity between users in a social network. We summarize the contributions of our work as follows.

- A novel social distance-aware Bayesian personalized ranking model, called SDBPR, is proposed to generate accurate recommendations. SDBPR recognizes the importance of the graph structure of the social network, which is useful for item information spread among users, and uses the random walk strategy to uncover the complex graph structure of the social network. The multistep distance is also used as the confidence for estimating a user's preference order for two items, thereby incorporating weak relationships in the SDBPR model.

- A novel learning algorithm for learning the parameters of the ranking model is proposed. Each user and item are mapped to a latent vector space that represents features of the user and item, and a stochastic gradient descent algorithm is employed to optimize the parameters. The results from two experiments show that our model outperforms existing methods in various metrics.

The remainder of this paper is organized as follows. Section 2 surveys the related work. Section 3 formulates and analyzes the problem and describes our proposed model and learning algorithm. Section 4 compares the experimental results of our model with other baselines on two real-word datasets and verifies the effectiveness of our method. Section 5 concludes this paper and discusses future considerations.

2. Related Work

2.1. Recommendation Systems

Item recommendation systems, which aim to provide users with personalized ranking lists of items, have been widely used by many internet service platforms. Substantial amounts of work have been done in this area. The most famous methods are collaborative filtering and content-based techniques [10]. Collaborative filtering looks for users with rating patterns similar to those of the active user and makes recommendations for the active user according to the ratings of the most similar users. Content-based techniques proceed in an item-centric manner. These techniques look for items that are similar to the current item and recommend the most similar items to the user. Content-based methods have proven to be more effective than traditional collaborative filtering. In [11, 12], Chen et al. proposed a 5G-smart diabetes system and smart personal health advisor (SPHA) that utilize various personalized information to recommend personalized treatment solutions for patients and provide an analysis of a user's health status. However, with content-based methods, it is difficult to determine what information is used for recommendation in some cases. Both collaborative filtering and content-based methods have memory-based and model-based implementations. Memory-based implementations, such as kNN [1], first compute the similarity and then make recommendations based on the top-K most similar users. Low-rank matrix factorization is a popular model-based implementation that factorizes the original matrix into low-rank matrices. Although the original user-item matrix is sparse, low-rank matrix factorization can produce a dense representation of the data. Matrix factorization maps each user and item into a common latent vector space of fixed dimension. Each dimension describes one feature of an item and a personal interest of the user. The dot product of the user latent vector and item latent vector is used to estimate the preference of users for items. This idea is widely used in recommendation task studies [13, 14]. We also adopt this method in our model.

2.2. Implicit Feedback and One-class Collaborative Filtering

The majority of the work on recommendation systems focuses on methods of using explicit feedback. However, explicit feedback is difficult to track in real-world applications; therefore, recommendation systems must rely on implicit feedback. One technique is to use implicit feedback to predict explicit feedback, such as ratings, and then use a method that utilizes explicit feedback to recommend items. Another technique is to directly model the implicit feedback. In [3], Hu et al. proposed a model to use preferences and confidence levels to represent implicit feedback. The preference of a user for an item is measured at different confidence levels. This model introduces confidence into the final loss function and accounts for all user-item pairs, including observed and nonobserved items. Moreover, implicit feedback often exists in a binary form. The problem of recommendation using this binary form of implicit feedback is called one-class collaborative filtering. In general, two types of methods can be used to solve this problem. The point-wise method attempts to fit numeric scores of items, and the pair-wise method [6, 7, 15] models the ranking order of items. The point-wise method assumes that positive feedback provides a higher preference score than does negative feedback. The matrix factorizing method works in this way. In [3], Hu et al. proposed an efficient alternating-least-squares optimization algorithm for factoring the confidence-based user-item matrix in linear time. Pairwise methods always regard implicit feedback as a flag that indicates that users show higher preferences for certain items over others. [6] presents a BPR model that assumes that users prefer observed items to nonobserved items. This method samples triples containing the user, the observed item and the nonobserved item from the data and then maximizes the probability of the user preferring the observed item over the nonobserved item. Following this process, other research has extended the BPR model by incorporating contextual information.

2.3. Social Recommendation

Social networks have become popular in real-world recommendation systems. The key point of social recommendation is to employ social information to improve the accuracy of the results. A common assumption is that a user may share interests with friends. Therefore, information about a user's friends is useful in estimating a user's preferences. Many previous studies [16, 14, 17, 18] have proved that social information can benefit recommendation tasks. In [16], Ma et al. proposed to incorporate social regularization in a matrix factorization framework. In [14], Jamali et al. exploited trust propagation in a social network for recommendation tasks. In [19], a hierarchical group matrix factorization (HGMF) technique was proposed to learn the user-group feature in a social network for recommendation.

However, the aforementioned studies, with a few exceptions, consider rating-based methods that focus on explicit feedback. In [7], Zhao et al. proposed an SBPR model that extends the BPR model by assuming that users prefer items consumed by their friends over other nonobserved items. In the SBPR model, an item consumed by more friends will have a smaller gap between this item and another item in terms of positive feedback.

In [15], Chen et al. used Jaccard's coefficient to distinguish between strong ties and weak ties; they then extended the BPR model by further assuming that a user prefers weak ties to strong ties. This model ranks items by tie type instead of by the number of friends who consumed the item. We also recognize the importance of assigning different weights to different social ties. In contrast to previous work, we use the distance in the social network to measure the strength of the relationship between any two users. Furthermore, we employ the random walk technique to traverse the social network. The distance between two users is computed along the random path as the summation of the Jaccard distance of the connected users in the path. We introduce the weak relationship between two users in ways that are more general than weak ties.

3. Design

In this section, we first present the general framework of SDBPR and personalized ranking. Next, we discuss the details of our SDBPR model, which employs the random walk strategy to uncover the graph structure of the social network. Finally, we incorporate weak relationships into our model by making the rank order assumption about nodes in the random walk path.

3.1. Overview of SDBPR

Many studies extend the BPR framework by incorporating contextual information. The general framework of our model is presented in Figure 3. Our model is an extension of BPR but differs in the sample process of the item pairs and the assumption of the rank order. First, we use random walk to sample the social network. Then, we make a ranking assumption for the items based on the sampled path produced by the random walk and the user-item matrix. Next, we map each user and item to a latent vector space and employ a stochastic gradient learning algorithm to learn the parameters. Finally, the model can produce personalized rankings of nonobserved items for the users.

In SDBPR, we use U to denote the total number of users and I to denote the total number of items. The social network is denoted as $G = (U, E)$, where $E \subseteq U \times U$. The feedback is denoted as $F \subseteq U \times I$. Given M users, N items, a social network G and feedback F, the problem of social recommendation is to produce a personalized ranking list of items R for every user. Let x_i denote the ith item in R, and let x_{ui} denote the preference of user u for item i. Then, for each item pair (x_i, x_j) in R, $x_{ui} \geq x_{uj}$ if $i < j$. For convenience, we also denote $T(u) = \{v \in U : (u, v) \in E\}$ as the friends of user u and $I(u) = \{i \in I : (u, i) \in F\}$ as the items consumed by user u.

3.2. Personalized Ranking

As mentioned above, the problem is one-class collaborate filtering. In this setting, implicit feedback contains only the item consumed by the user. If we

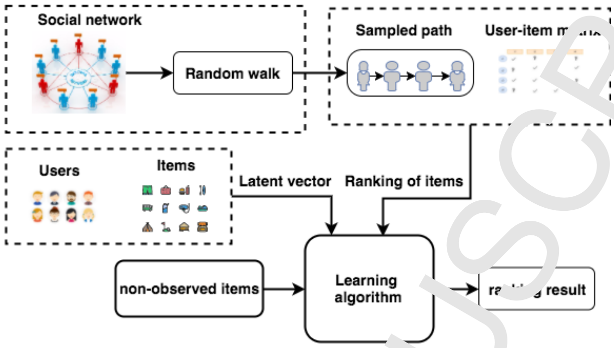


Figure 3: Framework of SDBPR.

assign 1 to observed items and 0 to nonobserved items, then most values in the user-item matrix will be 0. Recommendation aims to rank the nonobserved items for the user. If we consider only the observed items and ignore the nonobserved items, the dataset will be small because the user-item matrix is sparse. Another strategy involves first predicting the rating score of the items, including the observed items and the nonobserved items for each user, and then using the rating score to rank the items. Models, such as matrix factorizing, that use this process to fit the user-item matrix will fail because, to minimize loss, such as the mean squared error (MSE), the model attempts to predict 1 for all observed items and 0 for all nonobserved items. Therefore, these models cannot distinguish the differences among all nonobserved items for the users.

Based on the work in [6, 7], instead of simply ignoring the nonobserved items or predicting the rating score of nonobserved items, we directly optimize the rank of the items for each user. In this way, the model avoids the problem presented by the sparsity of the dataset. For each user u in the dataset, we sample tuples $S(u) = \{(u, i, j)\}$ from the item set and assume user u prefers i over j. Let x_{ui} denote the preference of user u for item i, and let Θ denote all parameters in the model. Then, we attempt to optimize the following objective function:

$$\begin{aligned}
L(\Theta) &= Prob(\Theta| > S) \\
&= \frac{Prob(\Theta)Prob(> S|\Theta)}{Prob(> S)} \\
&= \frac{Prob(\Theta)\prod_{u\in U}\prod_{(u,i,j)\in S(u)} Prob(x_{ui} > x_{uj})}{Prob(> S)} \\
&\propto Prob(\Theta)\prod_{u\in U}\prod_{(u,i,j)\in S(u)} Prob(x_{ui} > x_{uj})
\end{aligned} \tag{1}$$

where $S = \cup_{u\in U} S(u)$ and $> S$ represent that, for each tuple (u, i, j) in S, user u prefers item i over item j. We assume that the rank orders of all item pairs are

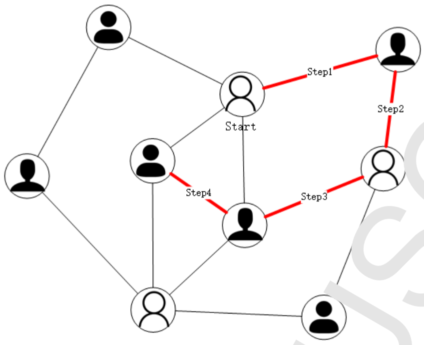


Figure 4: An example of a random walk in a social network.

independent of each other given the parameters Θ. Instead of directly maximizing the likelihood, we often minimize the log-likelihood function. Specifically,

$$lnL(\Theta) = lnProb(\Theta) + \sum_{u\in U} \sum_{(u,i,j)\in C(u)} lnProb(x_{ui} > x_{uj}) \tag{2}$$

We have now presented the general framework of BPR. Many studies extend the BPR framework by incorporating different contextual information. These studies differ in terms of the sample process of the item pairs and the assumptions of the rank order.

3.3. Sampling a Social Network by Random Walk

Social networks often include an extremely large number of users; thus, we cannot consider every user pair in the social network. However, in general, the average number of friends per user is limited. One simple strategy is to consider only user pairs that are composed of users and their friends. Many previous works [7, 15] have adopted this strategy. However, this strategy fails to consider the complex structure of the social graph. Note that information contained in the social network is propagated through the connections between users. We adopt the random walk strategy, seen in Figure 4, to travel the social graph in a random manner. The users along the path are sampled to be used in our model. For every connected user along the random walk path, we adopt the Jaccard coefficient to measure the strength of the connection between two users.

$$strength(u,v) = \frac{|T_u \cap T_v|}{|T_u \cup T_v|} \tag{3}$$

By definition, the strength of the connection between any two users is between 0 and 1. If users u and v are the same user, the strength of the connection between u and v is 1. If users u and v have no friends in common, the strength of the connection between u and v is 0. Because of this property, we can regard the strength between two users as the similarity. For two users u_i and u_j along the random walk path $u_1, u_2 ... u_i ... u_j ... u_{n-1}, u_n$, we define the strength of the connection between u_i and u_i as follows:

$$strength(u_i, u_j) = \prod_{k=i+1}^{k=j} strength(u_k, u_{k-1}) \tag{4}$$

Algorithm 1: Random walk algorithm for sampling

```
Input: user u, social graph G
Output: list of pairs (user, strength)
visited ← ∅;
pairs ← ∅;
max_allowed_hop ← 6;
cur_hop ← 0;
cur_user ← u;
cur_strength ← 1;
while cur_hop < max_allowed_hop do
    visited ← visited ∪ (cur_user, cur_strength);
    found ← false;
    friends = the friends of cur_user in the social graph G;
    shuffle(friends);
    foreach f in friends do
        if f not in visited then
            cur_strength ← cur_strength × strength(f, u);
            cur_user ← f;
            found ← true;
            break;
        end
    end
    if not found then
        break;
    end
    cur_hop ← cur_hop + 1
end
```

The sample strategy used in our model is the random walk. We select a user as the initial node to randomly walk the social graph. The random walk algorithm for sampling the social network is presented in Algorithm 1. First, we initialize the necessary variables. For each current user, we first compute the strength of the connection between the current user and the initial user. Next, we sample a nonvisited user from the friends of the current user as the next user. We terminate the algorithm if no such user exists. For each user along the random walk path, we add the user and strength pair to the final returned set.

3.4. Ranking Assumption on Sampled Paths

We discussed the sampling strategy in the previous section. In this section, we focus on the rank order assumption used in our model. The work of SBPR considered the rank order between a user and the direct friends of the user. Let x_{ui}, x_{uk} and x_{uj} denote the preference of user u for items i, k and j, respectively; the rank assumption of SBPR is as follows:

$$x_{ui} > x_{uk} > x_{uj} \tag{5}$$

where i is an item consumed by u, k is an item consumed by the friends of u, and j is an item consumed by neither u nor the friends of u.

In contrast to SBPR, which considers only the rank order between a user and the direct friends of the user, our SDBPR model computes the strength of the connection between an initial user and every user along the random walk path. The rank order of items for a user is estimated according to the strength of the connection between the users in the random walk path. Formally, we assume that user u prefers items consumed by a user with a high-strength connection over items consumed by a user with a low-strength connection. The rank order assumption in SDBPR is as follows:

$$x_{ui} > x_{uj} \tag{6}$$

where i is a randomly chosen item in $I(v)$, j is a randomly chosen item in $I(w)$, and v and w are any two directly connected users in the random walk path. We also introduce a coefficient c_{uij} to measure the confidence of this assumption. We define the probability of this assumption as follows:

$$Prob(x_{ui} > x_{uj}) = \sigma(\frac{x_{ui} - x_{uj}}{c_{uij}}) \tag{7}$$

where $\sigma(x) = \frac{1}{1+e^{-x}}$ and $c_{uij} = 1 + e^{strength(u,v) - strength(u,w)}$.

3.5. Parameter Learning

We adopt the latent factor strategy in our model. Every user and item is projected into a d-dimensional latent vector space. The dot product between the user vector and the item vector is used to measure the preference of a user for an item. The latent vectors of user u and item i are denoted as $\alpha(u)$ and $\beta(i)$. Thus, the preference of user u for item i is

$$x_{ui} = \alpha(u)^T \beta(i) \tag{8}$$

Our aim is to find the parameters that maximize $lnL(\Theta)$. Moreover, we assume that the prior of the parameters Θ follows a Gaussian distribution. We obtain the following final objective function of our SDBPR model:

$$lnL(\Theta) = \sum_{u \in U} \sum_{(u,i,j) \in S(u)} ln\sigma(\frac{x_{ui} - x_{uj}}{c_{uij}}) - \lambda_u \sum_{u \in U} \alpha(u)^T \alpha(u) - \lambda_i \sum_{i \in I} \beta(i)^T \beta(i) \tag{9}$$

Algorithm 2: Learning algorithm

```
Input: users U, items I, feedback F, social graph G
Output: parameters Θ
for iteration ← 1 to max_iteration do
    foreach user u in U do
        pairs ← RandomWalk(u,G);
        Sort(pairs); //sort pairs according to strength of user by
          descending ;
        length ← the length of pair ;
        for p ← 2 to length do
            v ← user in p position of pairs;
            w ← user in p − 1 position of pairs;
            i ← randomly pick an item from I(v);
            j ← randomly pick an item from I(w);
            Compute the gradient of the parameter Θ = {alpha(u),
              beta(i),beta(j)}; Θ ← Θ + ∂lnL(Θ)/∂Θ;
        end
    end
end
```

We employ a stochastic gradient descent algorithm to learn the parameters of our model. For each iteration, we first sample the training instances using the random walk algorithm. Then, for each sampled instance, we compute the derivative of $lnL(\Theta)$ for all parameters in the model. Finally, we adjust the parameters based on the positive gradient of the parameter. We present the learning algorithm in Algorithm 2.

4. Experiments

4.1. Experiment Setup

We use two datasets from two popular websites. These two datasets have different forms, so we transform them into a uniform form for use with our model. Both datasets contain social network information of users. The feedback in the datasets indicates whether a user u consumed an item.

- **Epinions Dataset.** This dataset comes from Epinions, an online consumer review website. Two files are included in this dataset. One file contains positive trust statements: each line in the file represents a source user who trusts another target user. The other file contains feedback: each line in the file contains a rating between 1 and 5 given by a user for an item. We convert the dataset into four separate files. The first two files record all users and all items, the third file records the trust relationships of users, and the fourth file contains binary feedback. We treat scores higher than 3 as positive feedback, representing that a user consumes an item. The trust file is used to build a directed social graph, and we randomly select 80% of the feedback data as training data and use the remaining data as testing data.

- **Ciao Dataset.** Users of the ciao website can write reviews and comments about products to help others make purchasing decisions. Customers can read these reviews to consider the opinions of other customers about a product. Three files are included in this dataset: we consider only two of them. One file contains trust information: each line in the file contains a trustor id and a trustee id. The other file contains the ratings of movies: each line in the file contains a rating between 1 and 5 given by a customer to a movie. We convert the dataset into four separate files. Two of them are used to record all customers and movies, one records the trust relationships of the customers, and the final contains the binary feedback produced by the same rules as those used for the Epinions dataset. Similar to the Epinions dataset, we use the trust file to build a directed social graph and split the feedback data into two parts: 80% is used to train the model, and the rest is used to evaluate the model.

To demonstrate the effectiveness of our approach, we apply our method and other four baseline methods to the two datasets.

- **Random.** Randomly sample items from the list of items the user has not consumed to build a ranked list for each user.
- **Most Popular.** In this method, all items are ranked based on their prevalence, that is, the frequency of appearance of items in the feedback.
- **BPR.** The BPR method assumes that users prefer observed items over nonobserved items.
- **SBPR.** The SBPR method extends the BPR method by ranking nonobserved items behind items consumed by the user's friends.

We use three popular metrics to evaluate the recommendation quality of our model and make comparisons among all methods.

- **Recall@K.** This metric is an extension of the traditional recall metric. For each user in the dataset, Recall@K measures the fraction of all actually consumed items in the test set found in the list of top K items predicted by the model. A higher value of this metric means that the model has the power to find more nonobserved items that will be consumed by the user in the future. The final recall value is the mean of all recall values of a user. We define $C(u)$ as the list of consumed items for user u in the test set. $C(K, u)$ represents the items in the test set already consumed by u that appear in the list of top K ranked items produced by the model. The formula of Recall@K is defined as follows.

$$Recall@K = \frac{|C(K,u)|}{|C(u)|}$$

- **Precision@K.** This metric is an extension of the traditional precision metric. For each user in the dataset, Precision@K corresponds to the number of actually consumed items in the list of top K predicted items divided by the K list produced by the model. A higher value of this metric means that the model makes more correct predictions. The final precision value is the mean of all precision values of a user. $C(K,u)$ has the same meaning as in Recall@K. The formula of Precision@K is defined as follows.

$$Precision@K = \frac{|C(K,u)|}{|K|}$$

- **AUC (Area under the curve).** This metric measures the probability that the recommendation model will assign a higher preference to a randomly chosen consumed item than to a randomly chosen nonconsumed item. A higher value of this metric means that the model will rank more positive items ahead of negative items. The final AUC value is the mean of all AUC values of a user. The formula of AUC is defined as follows.

$$AUC = \frac{1}{|U|} \sum_{u \in U} \frac{1}{|M(u)|} \sum_{(i,j) \in M(u)} I(x_{ui} > x_{uj})$$

where $M(u) = \{(i,j) : j \in C(u)\ and\ j \in N(u)\}$,$C(u)$ denotes the items consumed by user u, $N(u)$ denotes the nonconsumed items of user u, and I is an indicator function that outputs 1 if item i is ranked ahead of item j.

- **NDCG (Normalized discounted cumulative gain).** This metric considers the weighted rankings of recommended items. Items ranked higher have larger weights in the final result. For each user in the dataset, NDCG is computed by accumulating the weighted rankings of items for the user. A higher value of this metric means that more positive items are ranked at the top of the ranking list. To define NDCG, we first define DCG as follows.

$$DCG(u) = \sum_{j=1}^{n} \frac{2^{r(j)} - 1}{log_2(1 + j)}$$

where $r(j)$ is the score of the item in the jth position of the recommendation list of user u. The value of $r(j)$ is equal to 1 if item j is consumed by u and is otherwise 0. The NDCG of user u is the normalized result of DCG and is define as follows.

$$NDCG(u) = \frac{DCG(u)}{Z(u)}$$

where $Z(u)$ is the ideal value of DCG(u) for user u, which is computed as the best possible ranking of items for the user. The final NDCG value is the mean of all NDCG values of a user.

$$NDCG = \frac{1}{|U|} \sum_{u \in U} NDCG(u)$$

Table 1: Performance evaluations on the Ciao dataset

Method	Random	Most Popular	BPR	SBPR	SDBPR
Prec@10	0.0001	0.0025	0.0033	0.0046	[illegible]
Rec10	0.0003	0.003	0.02677	0.0354	[illegible]
AUC	0.4999	0.6531	0.6525	[illegible]	[illegible]
NDCG	0.0838	0.1158	0.1164	[illegible]	[illegible]

Table 2: Performance evaluations on the Epinions dataset

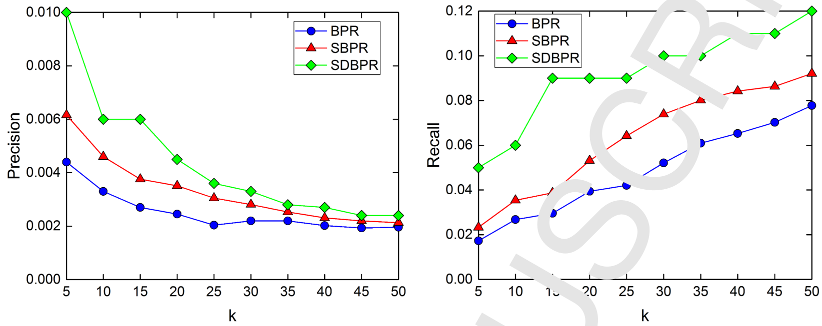
Method	Random	Most Popular	BPR	SBPR	SDBPR
Prec@10	0.00005	0.0004	0.0035	[illegible]	0.0052
Rec@10	0.00047	0.012	0.011	[illegible]	0.01811
AUC	0.5068	0.6436	[illegible]	[illegible]	0.7072
NDCG	0.07811	0.0933	[illegible]	[illegible]	0.1136

4.2. Results and Analysis

We implement our model and four other models with factor sizes ranging from 2 to 20. Table 1 and Table 2 detail the recommendation performance of the five models with a factor size of 20 on two datasets in terms of four different metrics. From Table 1 and Table 2, we can conclude that our SDBPR outperforms the other four models in all cases.

SDBPR vs Baselines. The performance of the Random method is generally the worst among all the methods on both datasets. The gap between SDBPR and Random for Recall@K, Precision@K, AUC and NDCG is large. Because the Random method randomly selects an item from the nonobserved items, it ignores all information contained in both the feedback and the social graph. The Most Popular method selects items according to their global popularity; this method is a strong baseline method compared to the Random method. The smallest gap between the Most Popular method and SDBPR is found for AUC and NDCG. Because many users show high preferences for a small set of items, the Most Popular method assigns higher ranks to these items. The Most Popular method is expected to achieve good performance in terms of AUC and NDCG. However, because the Most Popular method does not consider the personality of users, it suffers from poor performance in terms of Recall@K and Precision@K when we restrict K to a small number. Both BPR and SBPR achieve good performance in terms of AUC and NDCG because these two metrics are analogous to the ranking objective. SBPR is superior to BPR in all cases because SBPR distinguishes between social feedback and negative feedback, whereas BPR treats them equally.

Our SDBPR model also outperforms both SBPR models in all cases. The largest gap is in the Recall@K metric on both datasets. The main reason for this result is that, in contrast to SBPR, our model distinguishes the influence of different users in the social network. On the Ciao dataset, SDBPR outperforms SBPR by as much as 30%, 69%, 4%, and 4.2% in terms of Precision@K, Recall@K, AUC and NDCG, respectively. On the Epinions dataset, SDBPR



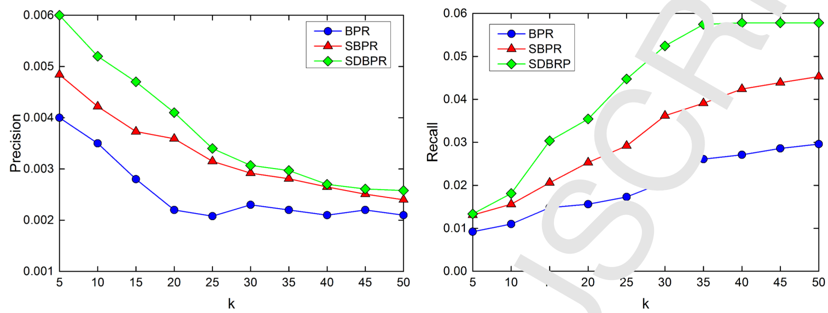
(a) Precsion (b) Recall

Figure 5: Precision and recall on Ciao.

outperforms SBPR by 23%, 16%, 5.3%, and 4.1% in terms of Precision@K, Recall@K, AUC and NDCG, respectively. The performance of models that consider the social network is better than the performance of those that do not because the model can learn more about users from the social information in the dataset to improve performance. Thus, social information is useful for item recommendation.

Recall and Precision. In Figure 5 and Figure 6, we evaluate the Precision@K and Recall@K of three models, namely, BPR, SBPR, and SDBPR, when varying K from 5 to 50. We exclude the Random model and the Most Popular model because the precision and recall of these two methods do not have meaningful trends when we vary K. In both datasets, the precision continues to decrease as K increases. The precision changes rapidly when K is small but more slowly when K is large. The gap in precision between our model and the other two models decreases with increasing K. When K is approximately 50, the precision of the three models on both datasets stops changing at nearly the same value. The recall of the three models on both datasets continues to increase at a steady rate as K increases from 5 to 50. We might expect the recall to stop changing at some point. From Figure 5 and Figure 6, we conclude that our SDBPR significantly improves the precision of the other two baseline models when K is small and the recall when K is large. In real recommendation systems, the value of K is often set to a medium number between 5 and 20; thus, we expect our model to achieve good performance in practice.

Performance with Respect to N (Number of Latent Factors). We evaluate three models, namely, BPR, SBPR, and SDBPR, when varying the size of the latent vector N from 1 to 20. As shown in Figure 7a and Figure 7b, when N is small, the gap between these models is small. The gap of AUC and NDCG on Ciao increases as N increases. However, when N exceeds a threshold, all models converge. We can interpret this phenomenon as follows. When N is small, the latent vector is unable to capture information from either the user-



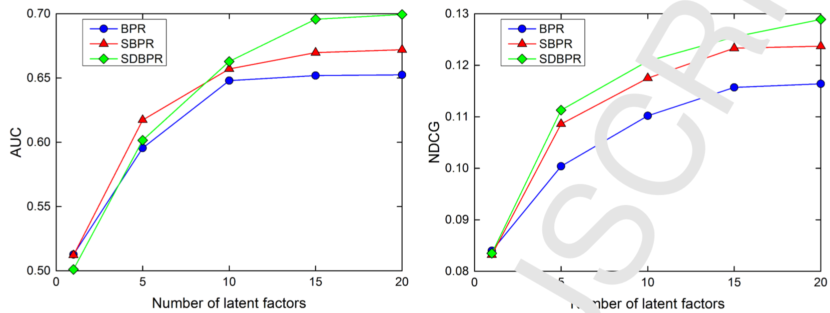
(a) Precision (b) Recall

Figure 6: Precision and recall on Epinions.

item relationship or from the social network. When N becomes large, SBPR and SDBPR are able to capture the information contained in the social network; therefore, the gap between the social model and the nonsocial model increases. Moreover, our SDBPR model consistently outperforms SBPR with varying N, and the gap between these models becomes large as N increases. The reason for this change is that, when N is sufficiently large, SBPR learns only one-step strength in the social network but SDBPR has enough parameters to learn the multistep strength in the social network. Therefore, in contrast to SBPR, our SDBPR model has the power to learn the complex social network structure. The trends of AUC and NDCG on the Epinions dataset in Figure 8a and Figure 8b are similar to those on the Ciao dataset shown in Figure 7a and Figure 7b, but the performance on Epinions is worse than that on Ciao. There are more irrelevant items in the total item set on Epinions because the number of items in the Epinions data set is large. Both AUC and NDCG consider the rank of all possible items in the data set, and so the performance on Epinions is not as good as that on Ciao. We note that, when N is increased to approximately 20, the models show minimal improvement; thus, for many models, a small N is sufficient.

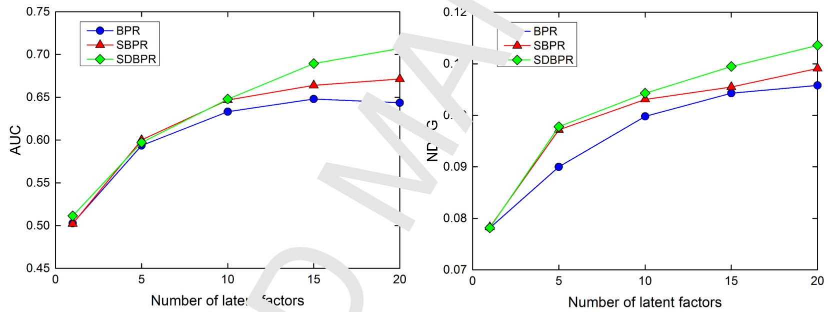
5. Conclusion

In this paper, we studied a method of exploiting social network information to improve the performance of recommendation systems. We designed a model called SDBPR based on previous work. To uncover the complex graph of a social network, we employed the random walk method to sample users in the social network. We also incorporated weak relationships into the SDBPR model by making a rank order assumption between users at multistep distances. Compared to existing recommendation approaches, our approach achieves good performance when applied to the one-class collaborative filtering problem. Our work creates



(a) AUC (b) NDCG

Figure 7: Correlation of AUC and NDCG with the number of latent factors on Ciao.



(a) AUC (b) NDCG

Figure 8: Correlation of AUC and NDCG with the number of latent factors on Epinions.

opportunities for future research. First, we can consider other methods, such as preferential attachment and Adamic-Adar, to replace Jaccard's coefficient to measure the strength between two users. Second, content-based methods often show good performance; we could consider how to combine the properties of users and items into social recommendations. Third, to process data produced by an extremely large number of users and items in large recommendation systems in a reasonable time, we could consider scaling the SDBPR model for use in the MapReduce framework.

Acknowledgment

This work was supported in part by National Natural Science Foundation of China under Grants No.61672256 and Guandong Science and Technology Plan under Grants no. 2017B030305003.

References

[1] M. Deshpande, G. Karypis, Item-based top-N recommendation algorithms, ACM Trans. Inf. Syst. 22 (1) (2004) 143–177.

[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, 2008, pp. 426–434.

[3] Y. Hu, Y. Koren, C. Volinsky, Collaborative filtering for implicit feedback datasets, in: Proceedings of the 8th IEEE International Conference on Data Mining, 2008, pp. 263–272.

[4] Y. Li, C. X. Zhai, Y. Chen, Exploiting rich user information for one-class collaborative filtering, Knowledge & Information Systems 38 (2) (2014) 277–301.

[5] L. K. Gee, J. J. Jones, M. Burke, Social networks and labor markets: How strong ties relate to job finding on facebook's social network, Journal of Labor Economics 35 (2) (2017) 485–518.

[6] S. Rendle, C. Freudenthaler, Z. Gantner, L. Schmidt-Thieme, Bayesian personalized ranking from implicit feedback, in: Proceedings of the 25 Conference on Uncertainty in Artificial Intelligence, 2009, pp. 452–461.

[7] T. Zhao, J. J. McAuley, I. King, Leveraging social connections to improve personalized ranking for collaborative filtering, in: Proceedings of the 23rd ACM International Conference on Conferenceon Information and Knowledge Management, 2014, pp. 261–270.

[8] Y. Li, C. Constantin, C. Du Mouza, Sgvcut: A vertex-cut partitioning tool for random walks-based computations over social network graphs, in: Proceedings of the 29th International Conference on Scientific and Statistical Database Management, 2017, pp. 1–4.

[9] D. Liben-Nowell, J. Kleinberg, The link-prediction problem for social networks, Journal of the American Society for Information Science and Technology 58 (7) (2007) 1019–1031.

[10] G. Adomavicius, A. Tuzhilin, Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions, IEEE Trans. Knowl. Data Eng. 17 (6) (2005) 734–749.

[11] M. Chen, J. Yang, J. Zhou, Y. Hao, J. Zhang, C. H. Youn, 5g-smart diabetes: Toward personalized diabetes diagnosis with healthcare big data clouds, IEEE Communications Magazine 56 (4) (2018) 16–23.

[12] M. Chen, Y. Zhang, M. Qiu, N. Guizani, Y. Hao, Spha: Smart personal health advisor based on deep analytics, IEEE Communications Magazine 56 (3) (2018) 164–169.

[13] C. Zhang, X. Zhao, K. Wang, J. Sun, Content + attributes: A latent factor model for recommending scientific papers in heterogeneous academic networks, in: European Conference on Information Retrieval, Springer, 2014, pp. 39–50.

[14] M. Jamali, M. Ester, A matrix factorization technique with trust propagation for recommendation in social networks, in: Proceedings of the 2010 ACM Conference on Recommender Systems, 2010, pp. 135–142.

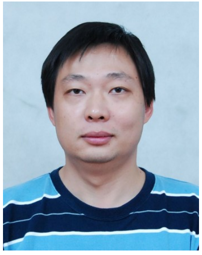
[15] X. Wang, W. Lu, M. Ester, C. Wang, C. Chen, Social recommendation with strong and weak ties, in: Proceedings of the 25th ACM International Conference on Information and Knowledge Management, 2016, pp. 5–14.

[16] H. Ma, D. Zhou, C. Liu, M. R. Lyu, I. King, Recommender systems with social regularization, in: Proceedings of the Forth International Conference on Web Search and Web Data Mining, 2011, pp. 287–296.

[17] M. Ye, X. Liu, W. Lee, Exploring social influence for recommendation: a generative model approach, in: Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval, 2012, pp. 671–680.

[18] P. Wang, B. Xu, Y. Wu, X. Zhou, Link prediction in social networks: the state-of-the-art, SCIENCE CHINA Information Sciences 58 (1) (2015) 1–38.

[19] X. Wang, W. Pan, C. Xu, Hgmf:hierarchical group matrix factorization for collaborative recommendation (2014) 769–778.



Feng Zhao received his M.S. and Ph.D degrees in Computer Science from Huazhong University of Science and Technology in 2003 and 2006. He is currently a professor of School of Computer Science and Technology, Huazhong University of Science and Technology, China. Zhao is a member of the IEEE and a member of the ACM. He has published more than 60 papers at various conferences and journals, including SRDS, TSC, TBD, ISJ, INF, FGCS and CMA. His research interests include information retrieval, data mining, security and distributed computing.

Yu Shen received the B.S. degree in Software Engineer from Xiamen University, China, in 2016. He is currently a postgraduate student in computer science of Huazhong University of Science and Technology. His research interests include information retrieval and machine learning.

Xiangyu Gui received the B.S. degree in Optical Information Science and Technology from Huazhong University of Science and Technology, China, in 2015. He is currently a doctoral student in computer science of Huazhong University of Science and Technology. His research interests include information retrieval and knowledge graph.

Hai Jin is a Cheung Kung Scholars Chair Professor of computer science and engineering at Huazhong University of Science and Technology (HUST) in China. Jin received his PhD in computer engineering from HUST in 1994. In 1996, he was awarded a German Academic Exchange Service fellowship to visit the Technical University of Chemnitz in Germany. Jin worked at The University of Hong Kong between 1998 and 2000, and as a visiting scholar at the University of Southern California between 1999 and 2000. He was awarded Excellent Youth Award from the National Science Foundation of China in 2001. Jin is the chief scientist of ChinaGrid, the largest grid computing project in China, and the chief scientists of National 973 Basic Research Program Project of Virtualization Technology of Computing System, and Cloud Security.

Jin is a senior member of the IEEE and a member of the ACM. He has co-authored 15 books and published over 500 research papers. His research interests include computer architecture, virtualization technology, cluster computing and cloud computing, peer-to-peer computing, network storage, and network security.

HIGHLIGHTS

We summarize our contributions as follows.

- A novel social-distance-aware Bayesian personalized ranking model, named SDBPR, is proposed to generate more accurate recommendations. SDBPR recognizes the importance of the graph structure of the social network, which is useful for item information spread among users, and it proposes to use the random walk strategy to uncover the complex graph structure of the social network. The multistep distance is also used as the confidence for estimating a user's preference order for two items, therein incorporating weak relations in SDBPR model.
- A novel learning algorithm for learning the parameters for the ranking model is proposed. Each user and item are mapped to a latent vector space that represents different aspects of the user and item, and a stochastic gradient descent algorithm is employed to learn the parameters. Experimental results from two experiments demonstrate that our algorithm outperforms existing methods in terms of various metrics.