Research on Recommendation Models for One-class Collaborative Filtering

Zhang Qian

A Thesis submitted to Tokushima University in partial fulfillment of the requirements for the degree of Doctor of Philosophy

March 2022

Department of Information Science and Intelligent Systems
Graduate School of Advanced Technology and Science
Tokushima University, Japan

Contents

List of Figures

List of Tables

Acknowledgment

It is a good chance for me to express gratitude for everything and everyone in three years through this thesis. A1 group provides me with a good research environment to focus on studying research content at any time. I will always remember the nice learning atmosphere created by teachers and classmates in the A1 group.

First, I would like to give my greatest thanks to my supervisor Professor Fuji Ren for the careful guidance and enthusiastic support on my studies on data mining and recommender systems. Professor Ren is a good teacher with a piece of profound knowledge, a rigorous style. Professor Ren listened carefully to my research content, answered my doubts and provided valuable suggestions many times during these years. With Professor Ren's full support and patient guidance, I will continue my research on artificial intelligence.

I would like to express my sincere gratitude to Professor Kenji Terada and Professor Masami Shishibori for their time and effort in reviewing my thesis and thank for their valuable suggestions. Meanwhile, I would like to thank Shun Nishide and Xin Kang for their help throughout my Ph.D. course. It is an honor for me to know those respected teachers.

Furthermore, I would like to thank my classmates in the A1 group. Thanks to Xudong Zhang, Tianhao She, Zheng Liu, Ziyun Jiao and Yangyang Zhou for their valuable suggestions and shared experience in research. Thanks to the graduated seniors Ning Liu, Zhichao Cui, Wenjie Liu, Duo Feng and Mengjia He for their guidance early in my research. Especially, I would like to thank my labmates, Jiawen Deng and Siyuan Xue, for their encouragement and help in my study and life. In my heart, they are not only excellent seniors but also good friends who have companied with me for two years.

Finally, thanks to my parents for their support and understanding of my choice to pursue the Ph.D. course.

Abstract

Recommender systems have become an indispensable tool for real-world applications. One-class collaborative filtering has attracted much attention in recommendation communities because the “one-class” is more suitable to describe data of many applications. Many recommendation methods have been proposed for realizing personalized ranking with one-class feedback (implicit feedback). Pairwise ranking methods with relative preference assumptions are widely used for dealing with the one-class problem due to their high performance. Bayesian Personalized Ranking (BPR) is one of the most popular pairwise methods, assuming users prefer the observed item to the unobserved item.

BPR assumes the equal importance of each user’s unobserved items. However, existing some items that users have not seen yet. It is not appropriate to treat each user’s all unobserved items equally. Additionally, the parameters in BPR are learned by the stochastic gradient descent (SGD) optimization algorithm. The previous work has shown that the vanishing gradient problem exists in the learning process when the user’s preference difference between the observed item and the unobserved item is very large.

In order to alleviate the problems of the previous model, three recommendation models are studied for one-class collaborative filtering in this thesis, including PBPR (Prior-based Bayesian Pairwise Ranking), PBPR* (Improving PBPR) and DBPL (Double Bayesian Pairwise Learning). All three recommendation models consider users’ preference differences between their unobserved items and can be realized without any additional social information. In addition, the users’ potential preference scores on their unobserved items are calculated based on users’ historical interactions for further distinguishing the relative preference of each user’s any two unobserved items. The key contributions in this thesis are summarized as below:

(1) Motivated by the discovery that the user may be interested in items that their like-minded users have observed, users’ potential preference scores on their unobserved items could be calculated by the similarities between users and the similarities between items. The similarities between users and the similarities between items are measured at the item level and the entity level considering that the user might like the item or entity. Experiments on the real-world dataset demonstrate the results of the *UIIU* (user-based item similarity and item-based user similarity) method are the best in most cases.

The potential preference scores calculated by the *UIIU* method are used for further studying.

(2) With the observation that each user has his/her own chosen intention on different service systems and most people's chosen intention about items have continuity and do not change suddenly, the Latent Dirichlet Allocation (LDA) model is used to realize this observation. The user's chosen intention is considered as the hidden variable, and two distributions (user-chosen intention distribution and item-chosen intention distribution) are updated during the learning process of the model. The users' potential preference scores can be obtained by the inner product of two distributions. Experimental results of the LDA-based method are better than BPR across all evaluation metrics on three datasets.

(3) For alleviating the assumption in BPR that equal importance of the huge unobserved items, the novel model PBPR is proposed. It relaxes the simple pairwise preference assumption in BPR by further considering the pairwise preference between any two unobserved items. PBPR considers the situation of existing fine-grained preference difference between any two unobserved items of a user. It assumes the user prefers an unobserved item with a higher potential preference score over another unobserved item. PBPR* is proposed to enhance the performance of PBPR by conducting several strategies to overcome shortcomings in PBPR, for more accurate recommendation results.

(4) With the consideration that the user's preference difference between the observed item and the unobserved item can be reduced by fusing a relatively smaller preference difference between another pair of items, DBPL is proposed by taking two pairwise preferences into the previous pairwise learning model. DBPL also takes into account each user's fine-grained preference differences between unobserved items. For each user, the unobserved item, which has a higher potential preference score, is assumed to have a smaller preference difference with the observed item of the user. Theoretically, DBPL could alleviate the vanishing gradient problem in the previous algorithm's learning procedure and obtain more accurate recommendations.

(5) A series of experiments over three real-world datasets are conducted to validate three recommendation models. Experimental results show the effectiveness of recommendation models for solving the one-class collaborative filtering problem. The experimental results of PBPR, PBPR* and DBPL are better than BPR, showing the effectiveness of assumptions proposed for recommendation models. Experimental

results of the PBPR*-based method are better than the PBPR-based method in most cases. Experimental results of the DBPL-based recommendation method outperform other recommendation methods across all evaluation metrics on all datasets.

1 Introduction

1.1 Background and Significance of Research

With the growing development of websites, a mass amount of data are being generated in real-time. It is not practical to process the information manually. Therefore, many effective methods have been proposed for dealing with online information in different domains in these years, like sentiment analysis methods for sentences [1, 2], opinion target extraction for opinionated sentences [3, 4]. The recommender system is an essential technique for discovering things of interest in online applications. It aims to tackle the problem of information overload in online e-commerce transactions and social network platforms [5]. Almost every service (search engines, social media sites, E-commerce, and news portals) that provides the content to users is equipped with a recommender system [6]. In recommender systems, item represents different kinds of contents consumed by users [7], like a movie, a song, or a book.

There are three types of personalized ranking recommendation approaches, including social recommendation, content-based filtering and collaborative filtering. Collaborative filtering (CF) based methods [8, 9], which take account of user-item interactions, have made great satisfactory success, among different recommended strategies [10, 11, 12, 13]. CF-based algorithms can be divided into two categories: memory-based algorithms and model-based algorithms [14, 15]. User-based collaborative filtering algorithms and item-based collaborative filtering algorithms, which all belong to memory-based algorithms (neighborhood-based algorithms), have been widely employed in practical recommender systems [16, 17] and provide users with explainable recommendation results [18]. The model-based collaborative filtering algorithms, especially latent-factor models based on matrix factorization (MF) [37], have shown effectiveness in recommender systems. However, the sparsity of the user-item interaction matrix usually influences the performance of CF-based recommendation methods. Researchers have alleviated this problem by incorporating additional information into CF-based methods [19, 20, 21].

The recommender system for the dataset, like MovieLens [22], in which users give the rating values scale 1-5, can be considered as a multi-class (explicit feedback) task.

For example, the accurate preference scores of users on different movies are predicted based on multi-class methods. MF-based recommendation methods have been shown to perform well for multi-class feedback in a number of real-world competitions and systems [39]. In many datasets, users' ratings on items are not always available [23] because of the large-scale and extreme diversity of multimedia content [40]. Existing many datasets, only one-class feedback is available [24], e.g., bought products, watched movies, and clicked Web pages. One-class collaborative filtering (OCCF) [102] is an emerging setup in collaborative filtering. Such datasets only have each user's positive feedback, usually called one-class feedback [25] or implicit feedback [26]. It is necessary to study recommendation methods for solving the one-class problem. The one-class collaborative filtering problem is different from that of the five stars rating prediction problem since the former only contains observed items rather than both positive items and negative items in the latter. Figure 1.1 shows the illustration of the representation matrix of the one-class feedback data. The symbol "+" represents the user who has observed the item, such as u_1 has observed i_1, and the corresponding position in the representation matrix is recorded as "1". The symbol "?" represents the user who has no interaction with the item, such as u_1 has no interaction with i_2, and the corresponding position in the representation matrix is recorded as "0".

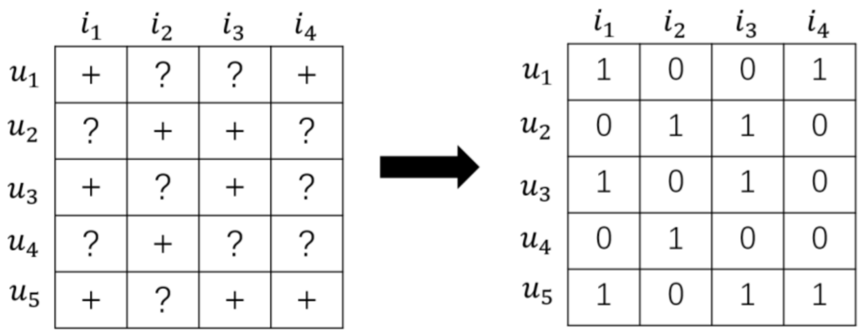


Figure 1.1 Illustration of the representation matrix of the one-class feedback data.

Compared with the traditional collaborative filtering setting where the data has ratings, one-class collaborative filtering is more realistic in many scenarios when no explicit feedbacks are available [42]. The recommendation methods for solving the one-class problem, which is also the implicit feedback problem, can be divided into two branches based on previous MF-based researches: pointwise regression methods [27, 28] and pairwise ranking methods [29, 30]. The pointwise regression methods, which take observed items as absolute preference scores [31] for the one-class problem, learn latent representations of users and items to represent users' preference scores and minimize a pointwise square loss to approximate the absolute scores. The pairwise ranking methods take an observed item and an unobserved item of a user as a triple <user, an observed item, an unobserved item>, assuming that users prefer observed items to unobserved items, and maximize the likelihood of pairwise preferences over observed items and unobserved items. Empirically, the pairwise ranking methods [32, 33], which have been successfully adopted in many scenarios [34, 35], achieve much better performance than pointwise methods [25, 28].

Bayesian personalized ranking (BPR) [32] is one of the most popular pairwise ranking methods. The optimization process of the BPR model assumes equal importance of the huge unobserved items. Actually, it is unsuitable for treating all unobserved items of the user as negative feedback. Some items may just not be seen by users. Moreover, BPR uses stochastic gradient descent (SGD) to learn parameters in the model. Most SGD updates have no effect when the big preference difference between the observed item and the unobserved item causes the vanishing gradient problem [33].

In general, we aim to study the novel recommendation model based on one-class feedback for more accurate recommendations in this thesis. The key points of our research are as follows:

(1) alleviating data sparsity problem with only user-item interactions;

(2) enhancing the performance of BPR by relaxing the problems mentioned above.

1.2 Literature Review

1.2.1 Pointwise methods and pairwise methods

One-class collaborative filtering recommendation methods are proposed for solving problems in real-world scenarios that only positive examples can be observed [41]. Pointwise methods and pairwise methods are two main manners of MF-based algorithms in recent researches.

In pointwise methods, the absolute preference scores are recoreded as one for further studying. Hu et al. [28] proposed the factor model for solving the one-class feedback problem. They treated the data as an indication of positive and negative preferences, which are associated with vastly varying confidence levels. In addition, many popular models like Factored Item Similarity Methods (FISM) [36] and Neural Collaborative Filtering (NCF) [38], using the pointwise method to train the model.

In recent years, pairwise ranking [43] has been widely used for the personalized recommendation from one-class feedback. It focuses on relative preferences rather than absolute ratings [44]. As mentioned before, the BPR model is one of the most popular pairwise methods and has been used as a basic learning model [7] in many studies [45, 46]. BPR adopts two fundamental assumptions: (a) the user prefers observed items over unobserved items. (b) the joint likelihood of pairwise preferences of user u is independent of that of the others. However, two fundamental assumptions exist limitations. Many extensions of BPR pointed out its limitations and proposed improved assumptions for better recommendation performance.

1.2.2 Recommendation methods for the data sparsity problem

- **Recommendation methods with the auxiliary feedback**

One important way to alleviate the data sparsity problem and enhance the performance of the recommender systems is to leverage the additional information. Many previous researchers have tried to improve the performance of BPR by taking into account additional information [47, 48]. Qiu et al. [49] adopted two typical actions, view and like, as auxiliary feedback to enhance the recommendation performance of service systems. They proposed the heterogeneous implicit feedback (BPRH) model that integrated multiple types of auxiliary action and target action into a unified model for more accurate recommendations. Li et al. [50] proposed a paper recommendation method (PRHN) based on the heterogeneous network, which integrates papers, venues, authors, terms, users and their relations. They applied random walk to calculate the recommendation scores of candidate papers to target users and employed BPR as the objective function to discover the user's personalized weights on different meta-paths that are investigated in the network. Zhao et al. [51] assumed that users tend to be interested in items that their friends prefer. They used social network information to distinguish the differences between users' unobserved items and then proposed Social-BPR (SBPR) for better recommendations. Sun et al. [52] proposed a social recommendation framework, which improves the performance of the recommendation results by leveraging friend and group information.

- **Recommendation methods without additional information**

Considering the situation that many real-world datasets do not contain additional information like social information in [51, 53], auxiliary feedback in [49, 54]. In such cases, the recommender systems could only rely on user-item interactions. As such, in existing efforts to enhance pairwise learning, many extensions of BPR do not need any additional information and can provide more accurate recommendations [15, 55]. Pan et al. [56] pointed out the limitations of BPR. They assumed that the group preference on item *i* is more likely stronger than user *u*'s preference on item *j* if user *u* observed item *i* and not observed item *j*. With this assumption, they proposed a novel model called group Bayesian personalized ranking (GBPR), which injected richer interactions among users. Yu et al. [57] proposed a Multiple Pairwise Ranking (MPR) approach to exploit the preference difference among multiple pairs of items by dividing the unobserved items into different parts. For scenarios without additional click data, they divided the unobserved items based on popularity, i.e., items' observed counts by all users. Liu et al. [58] adopted cosine similarity to measure the similarities between users for further constructing users' neighborhoods. Then, they proposed collaborative pairwise learning to rank (CPLR) algorithm, which considers the influence between users on the preferences.

1.2.3 Recommendation methods for the vanishing gradient problem

Recommendation methods based on BPR and its extensions use the uniform sampling assumption, which might have the vanishing gradient problem in the learning process when the item popularity has a tailed distribution. Gantner et al. [59] extended BPR to a probabilistic ranking criterion with the assumption that the unobserved items are sampled from a given distribution. Rendle et al. [33] did not fix the sampling probabilities. They analyzed issues in tailed item distributions and proposed a non-uniform item sampler to overcome the problem. To solve the problem that the convergence of the SGD learning algorithm slows down, Zhang et al. [60] proposed to dynamically choose negative training samples from the ranked list produced by the current prediction model and iteratively update their model. Ding et al. [61] proposed a simple yet effective sampler by leveraging the additional view data.

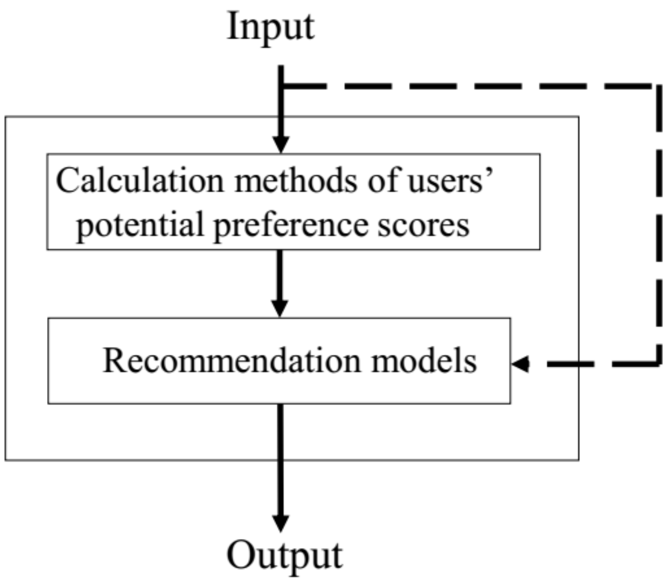


Figure 1.2 The overview of the model-based recommendation methods for one-class collaborative filtering

1.3 Research Contents and Contributions

Recommender systems play an essential role in finding users' potential preferences on products or services in various domains [62], such as social media sites, search engines and e-commerce platforms. In many real-world scenarios, explicit feedback data is not always available [63]. Only one-class feedback data can be observed, e.g., likes in WeChat, watches in YouTube, purchases in Amazon. Such data is also called implicit feedback data, which records the interactions between users and items without any scores. Recommendation methods for dealing with the one-class feedback problem assume that everything that has been selected is positive feedback for the user [33]. According to the above situation, in this thesis, we study recommendation models to give more accurate recommendations for solving the one-class problem.

The overview of using recommendation models for one-class collaborative filtering is shown in Figure 1.2. It mainly contains two parts, including calculation methods of users' potential preference scores and recommendation models.

- **Calculation methods of users' potential preference scores**

Considering that further study needs to distinguish the relative preference of each user's any two unobserved items, we first study calculation methods of users' potential preference scores on their unobserved items, including aspects of collaborative filtering and latent dirichlet allocation.

Motivated by the situation that the user may have interest in items that have been observed by his/her like-minded users and the user may like the similar items, we calculate users' potential preference scores on items by considering items from their like-minded users and similarities between items based on user-item interactions. Each item contains many entities, like the song containing the information of its topic, scenes, mood and language. The user might be interested in the item or the entity. Inspired by this, we calculate the user similarity at the item level and the entity level. We also calculate the user-based and entity-based item similarity for better performance.

We observe that each user has his/her own unique way of interacting with items, such as a user likes to choose a song according to the singer. We assume users have their personalized chosen intentions on different service systems and consider most people's chosen intention about items have continuity and do not change suddenly. We recommend items for users by tracking the chosen intentions from users' historical interactions. Convincing recommendation results can be obtained with the learning results of users' chosen intentions. Therefore, we leverage the topic model to model the generating process of each user's historical interactions for further evaluating each user's potential preference scores on unobserved items; the user's chosen intention is hidden information of the model.

- **Recommendation models**

Then, we study novel extensions of BPR for solving the one-class problem. We relax the assumption of the BPR that the equal importance of the huge unobserved items and propose a novel Prior-based Bayesian Pairwise Ranking (PBPR) model [64]. PBPR considers the pairwise preference between any two unobserved items according to users' potential preference scores on items rather than dividing unobserved items into different groups. We also take into account several strategies to enhance the performance of PBPR, which is denoted as the improving PBPR (PBPR*) model.

For the vanishing gradient problem caused by the tailed distribution, instead of proposing any sampling methods, we propose the Double Bayesian Pairwise Learning (DBPL) model [65]. We consider that, for each user, the unobserved item, which has a higher potential preference score, has a smaller preference difference with the observed item of the user. With this consideration, the vanishing gradient problem caused by the bigger preference difference between an observed item and an unobserved item of the user could be alleviated by further fusing the relatively smaller preference difference between another pair of items.

- **Contributions**

Our proposed recommendation models significantly improve the recommendation performance of the BPR model and can run without any additional information except user-item interactions and can be adjusted for the social information from a specific dataset.

In addition, we calculate potential preference scores as prior information based on users' historical interactions, which can be used to (1) measure fine-grained preference difference between any two unobserved items; (2) alleviate the data sparsity problem by potential preference scores between users and their unobserved items.

1.4 Organizations

This thesis focuses on studying recommendation models for one-class collaborative filtering, which are all considered each users' relative preferences for unobserved items. Therefore, we further study the calculation methods of users' potential preference scores on items on their unobserved items for measuring the fine-grained differences between any two unobserved items of the user. The rest of this thesis is organized as follows.

Section 2: Related Knowledge

In this section, we begin by reviewing related knowledge of this thesis, including (1) the basic concept of the matrix factorization algorithm and the gradient descent optimization algorithm; (2) the fundamental theoretical work in the fundamental recommendation model and its extension models, and (3) the structure and description of the topic model.

Section 3: Calculation Methods of Potential Preference Scores

In this section, we give the problem definition and then describe the calculation methods of users' potential preference scores on their unobserved items in aspects of collaborative filtering and latent dirichlet allocation in detail.

Section 4: Recommendation Models for One-class Collaborative Filtering

In this section, we propose three recommendation models (PBPR, PBPR * and DBPL) for more accurate recommendations. The objective functions and learning processes of the three models are described carefully.

Section 5: Experiment and Analysis

In this section, we conduct a series of experiments and analyze the results. The effectiveness of the proposed recommendation models is demonstrated by experimental results with all evaluation metrics on three real-world datasets.

Section 6: Conclusion and Future Work

In this section, we summarize the main contents of this thesis and give meaningful directions for future work.

2 Related Knowledge

We review the related knowledge of this thesis in this section. Matrix factorization is an essential algorithm for studying recommender systems. We first review the matrix factorization (MF) algorithm and the gradient descent (GD) optimization algorithm. Then, we describe the fundamental theoretical work in BPR and its extension models of BPR, i.e., group Bayesian personalized ranking (GBPR) and collaborative pairwise learning to rank (CPLR), in detail. We finally introduce the latent dirichlet allocation (LDA) topic model in this section, which will be used in the following study.

2.1 Matrix Factorization

Collaborative filtering, which can be divided into memory-based collaborative filtering [66] (user-based collaborative filtering and item-based collaborative filtering) and model-based collaborative filtering [67], is an essential technique for solving the problem of recommender systems. The model-based methods, i.e., latent factor model (LFM)-based methods [68, 69, 70], have attracted more and more attention. In recommendation algorithms, the advantage of the latent factor model is that it can be modeled according to the implicit information in users and items. Therefore, the deeper relationships between users and items can be mined by the model. Matrix factorization is the most successful implementation of the latent factor model. It has been widely applied in model-based collaborative filtering tasks because of its scalability and flexibility [71].

The core assumption of the MF-based recommendation algorithms is to use latent variables to represent users and items. The original relationship between users and items can be obtained by the product of their representation matrics. This hypothesis is established because the actual interaction data between users and items is generated under the influence of a series of hidden variables. Hidden variables represent the common features of users and items, including items' attribute characteristics and users' preference characteristics. However, these factors do not have practical significance and may not have very good interpretability. In addition, there is no definite label for each dimension in the matrix.

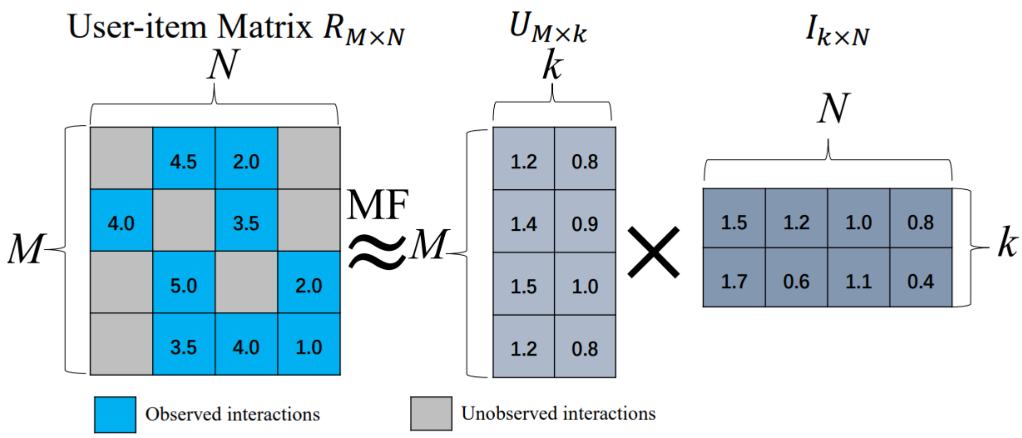


Figure 2.1 Illustration of matrix factorization

For a recommender system on the dataset, interactions between users and items can be represented by the user-item matrix $R_{M\times N}$. In the matrix $R_{M\times N}$, each row represents a user, and each column represents an item. If a user u rated an item i with a score, the value of R_{ui} is equal to the score. The user-item matrix $R_{M\times N}$ is called the scoring matrix. Figure 2.1 shows an example of the user-item scoring matrix. The matrix R is represented by the two low-rank matrices U and I, which, when multiplied, approximately reconstruct R The gray areas indicate that users have not interacted with items. The matrix factorization algorithm decomposes the user-item interaction matrix $R_{M\times N}$ into the inner product of two low-rank matrices, $U_{M\times k}$ and $V_{N\times k}$.

$$R_{M\times N} \approx U_{M\times k} \cdot V_{N\times k}^{T} \tag{2.1}$$

where $U_{M\times k}$ and $V_{N\times k}$ represent the user latent factor matrix and the item latent factor matrix. k represents the number of latent factors and k≪ min (M, N). Therefore, user u's preference score on item i can be calculated by:

$$r_{ui} = U_u \cdot V_i^T \tag{2.2}$$

where U_u denotes the latent factor vector of user u and I_i denotes the latent factor vector of item i. Matrix factorization algorithm has the following advantages:

(1) It is easier to implement by programming and can be trained by the stochastic gradient descent optimization algorithm, which will be introduced in Section 2.2.

(2) The accuracy of MF-based recommendation methods is higher than memory-based collaborative filtering methods.

(3) MF algorithm has very good scalability. It is convenient to consider other factors into the user feature vector and item feature vector.

The disadvantages of the MF algorithm are:

(1) The training process of the MF algorithm is time-consuming.

(2) The recommendation result of the MF algorithm is not interpretable. Each dimension of low-rank matrics of users and items cannot be explained in real life, so it can only be considered as a latent semantic space.

2.2 Gradient Descent and Stochastic Gradient Descent

Optimization problems have a major position in machine learning. The final goal of many machine learning algorithms is to solve optimization problems. Gradient descent (GD) is the simplest and most common calculation method among various optimization algorithms.

The gradient descent optimization algorithm in machine learning is usually used to calculate the model parameters for algorithms [72], i.e., linear regression model, logistic regression model, neural networks model, etc.

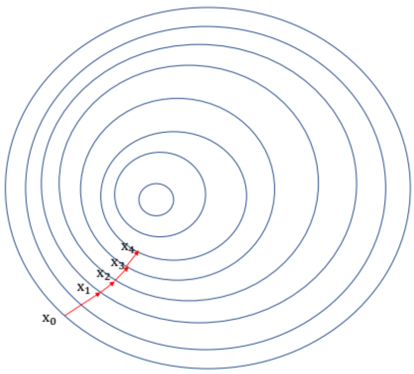


Figure 2.2 Description of the gradient descent optimization algorithm

Figure 2.2 shows the description of the gradient descent method. The principle of the gradient descent optimization algorithm is: the gradient of the objective function J_θ with respect to the parameter θ will be the fastest rising direction of the loss function. For minimizing the loss [73], we only need to advance the parameters by one step in the opposite direction of the gradient to achieve a drop in the objective function. The step size η is also called the learning rate. The parameter update formula is as follows:

$$\theta \leftarrow \theta - \eta \cdot \nabla J_\theta \tag{2.3}$$

where ∇J_θ is the gradient of the parameters.

There are three types of gradient descent: batch gradient descent, stochastic gradient descent and mini-batch gradient descent. Figure 2.3 summarizes the key points of three optimization algorithms.

Batch Gradient Descent	Stochastic Gradient Descent	Mini-Batch Gradient Descent
Parameters are updated after computing the gradient of error with respect to the entire training set	Parameters are updated after computing the gradient of error with respect to a single training example	Parameters are updated after computing the gradient of error with respect to a subset of the training set
It makes smooth updates in the model parameters	It makes very noisy updates in the parameters	Depending upon the batch size, the updates can be made less noisy – greater the batch size less noisy is the update

Figure 2.3 Description of the key points of three optimization algorithms

Stochastic Gradient Descent (SGD) is a simple but very efficient method and has been widely used for solving many machine learning problems [74, 75]. The time complexity of GD is $O(n)$ in each iteration when the number of training examples is n. The computational cost of GD is high when faced with massive data resources. SGD alleviates the computational cost by randomly sampling a single training example uniformly in each iteration. The time complexity is reduced from $O(n)$ to $O(1)$ in each iteration. In this thesis, parameters in all recommendation models are learned by the SGD algorithm.

2.3 Bayesian Personalized Ranking

BPR is one of the most popular pairwise ranking methods that take one-class feedback as relative preferences rather than absolute preferences [55]. BPR adopts two fundamental assumptions: (1) user u prefers the observed items over all the unobserved items. (2) the likelihood of pairwise preference of user u is independent of the others. The likelihood of BPR for all users and all items can be formulated as

$$\mathrm{BRR} = \prod_{u \in U^{tr}} \prod_{i \in I_u^+} \prod_{j \in I^{tr} \backslash I_u^+} Pr(r_{ui} > r_{uj})\,[1 - Pr(r_{ui} < r_{uj})] \tag{2.4}$$

where U^{tr} represents the set of users, I_u^+ represents the observed items of user *u*, $I^{tr} \backslash I_u^+$ represents the unobserved items of user *u*. r_{ui} represents user *u*'s preference on item *i*, which can be calculated by the inner product between the latent factor vector of user *u* ($U_u \in \mathbb{R}^d$) and the latent factor vector of item *i* ($V_i \in \mathbb{R}^d$), i.e., $r_{ui} = U_u^T \cdot V_i$, where *d* represents the number of latent dimensions.

BPR uses $\sigma(x) = \frac{1}{1+e^{-x}}$ to approximate the probability $Pr(\cdot)$ and adopts the log-likelihood to reduce the computational complexity. For each randomly sampled record, it includes a user *u*, an observed item *i* of user *u*, and an unobserved item *j*. Based on the above assumption, it maximizes the preference difference of user *u* towards item *i* and item *j*. The objective function can be written as

$$f(u,i,j) = -\ln \sigma(r_{uij}) + \frac{\beta}{2}(\|U_u\|^2 + \|V_i\|^2 + \|V_j\|^2) \tag{2.5}$$

where $\|U_u\|^2$, $\|V_i\|^2$ and $\|V_j\|^2$ are regularization terms to avoid overfitting, β is the regularization parameter. For user *u*, $r_{uij=} r_{ui} - r_{uj}$ represents the preference difference between item *i* and item *j*.

BPR uses the well-known SGD algorithm to learn the parameters. The parameters could be updated according to the following rule:

$$\theta \leftarrow \theta - \eta(1 - \sigma(r_{ui} - r_{uj}))\frac{\partial(r_{ui} - r_{uj})}{\partial\theta} \tag{2.6}$$

where θ can be U_u, V_i, or V_j, and $\eta > 0$ is the learning rate. Learning parameters in BPR are done by looping over Eq. (2.6).

2.4 Group Bayesian Personalized Ranking

Pan et al. proposed an extension of BPR, which is called group Bayesian personalized ranking (GBPR), in 2013. Motivated by the assumption that, comparing with the probability that user *u*'s preference score on observed item *i* is higher than user *u*'s preference score on unobserved item *j*, the probability that the preference score from a group of users on item *i* than unobserved item *j* is stronger. It can be written as

$$(G, i) > (u, j) \tag{2.7}$$

where G represents the group of users who have observed item i, and user u is in the group $(u \in G)$. The illustration of the above assumption via the toy example is shown in Figure 2.4.

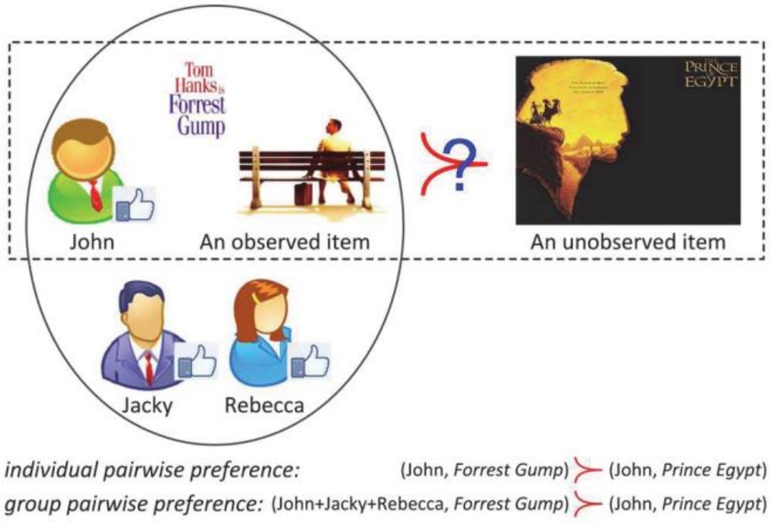


Figure 2.4 Illustration of preference assumption in group Bayesian personalized ranking

They combined the group preference and the individual preference and represented them by

$$(G, i) + (u, i) > (u, j) \text{ or } \widehat{r_{Gui}} > \widehat{r_{uj}} \tag{2.8}$$

where $\widehat{r_{Gui}} = \rho\widehat{r_{Gi}} + (1 - \rho)\widehat{r_{ui}}$ represents the fused preference on item i, and the trade-off parameter ρ is ranges from 0 to 1. The joint likelihood of any two users (u and w) can be approximated via $GBPR(u,w) \approx GBPR(u)GBPR(w)$. The overall likelihood of GBPR for all users and all items can be formulated as

$$\text{GBRR} = \prod_{u \in U^{tr}} \prod_{i \in I_u^+} \prod_{j \in I^{tr} \setminus I_u^+} Pr\left(\widehat{r_{Gui}} > \widehat{r_{uj}}\right)\left[1 - Pr\left(\widehat{r_{Gui}} < \widehat{r_{uj}}\right)\right] \tag{2.9}$$

where U^{tr} represents the set of users, I_u^+ represents the observed items of user u, $I^{tr} \setminus I_u^+$ represents the unobserved items of user u. G represents the user group, user u has observed item i and has not observed item j.

Following BPR, GBPR uses $\sigma(x) = \frac{1}{1+e^{-x}}$ to approximate the probability $Pr(\cdot)$ and adopts the log-likelihood to reduce the computational complexity. For each randomly sampled record, it includes a user u, an observed item i of user u, and an unobserved item j. Based on the above assumption, it maximizes the preference difference between the fused group preference towards item i and user u towards item j. The objective function can be written as

$$f(G,u,i,j) = -\ln \sigma\left(\widehat{r_{Gui:uj}}\right) + \frac{\beta}{2}\left(\sum_{w \in G} \|U_w\|^2 + \|V_i\|^2 + \|V_j\|^2\right) \tag{2.10}$$

where $\widehat{r_{Gui:uj}} = \widehat{r_{Gui}} - \widehat{r_{uj}}$ represents the difference between the fused group preference on item i and user u's preference on item j. $\sum_{w \in G} \|U_w\|^2$, $\|V_i\|^2$ and $\|V_j\|^2$ are regularization terms to avoid overfitting, β is the regularization parameter.

GBPR also uses the well-known SGD algorithm to learn the parameters. The parameters could be updated according to the following rule:

$$\theta \leftarrow \theta - \eta \frac{\partial f(G,u,i,j)}{\partial \theta} \tag{2.11}$$

where θ can be U_u $(u \in G)$, V_i, or V_j, and $\eta > 0$ is the learning rate. Learning parameters in GBPR are done by looping over Eq. (2.11).

2.5 Collaborative Pairwise Learning to Rank

Liu et al. proposed a novel model called collaborative pairwise learning to rank (CPLR) in 2018. They thought three problems in BPR, including (1) treats each user's unobserved items as the same; (2) treats each user's observed items as the same; (3) ignores the influence between any two users and solves them by considering the influence from other users on each user's preferences on observed items and unobserved items.

With the motivation that users preferred in the past develop to prefer in the future, the user has a high probably prefer items which have been observed by his/her like-minded users. Therefore, for each user u, all items can be divided into three sets: P_u, C_u and L_u. P_u represents user u's observed items, C_u represents items observed by user u's like-minded users, and L_u represents the left out items of user u.

$$\begin{cases} P_u = \{i \mid R_{ui} = 1\}, & \text{Positive set} \\ C_u = \{t \mid \exists w \in N_u,\ t \in P_w \ \&\ t \notin P_u\}, & \text{Collaborative set} \\ L_u = I - P_u - C_u, & \text{Left-out set} \end{cases}$$

where N_u represents the set of user u's like-minded users, the similarity between any two users is determined by the cosine similarity, which is defined as follows.

$$sim(u, w) = \frac{|P_u \cap P_w|}{|P_u| \cdot |P_w|} \tag{2.12}$$

CPLR adopts two fundamental assumptions: (1) user u prefers the observed items over the other unobserved items, and (2) user u would prefer the unobserved items, which have been observed by his/her like-minded users, over the other unobserved items. Therefore, the relative preference can be written as follows.

$$f(u, i) > (u, t), \qquad (u, t) > (u, j), \qquad (u, i) > (u, j) \tag{2.13}$$

Based on the assumptions, for each randomly sampled record, it includes a user u, an observed item i of user u ($i \in P_u$), and two unobserved items t ($t \in C_u$) and j ($j \in L_u$). Based on the above assumption, it maximizes the preference differences of user u towards three pairs of items (item i and item t, item t and item j, item i and item j). The objective function can be written as

$$\begin{aligned} f(u, i, t, j) = -(&\alpha \ln \sigma(c_{uit}\,(r_{ui} - r_{ut})) + \beta \ln \sigma\big(c_{utj}\,(r_{ut} - r_{uj})\big) \\ &+ \gamma \ln \sigma\big(c_{uij}\,(r_{ui} - r_{uj})\big)) + \frac{\lambda_\theta}{2}(\|U_u\|^2 + \|V_i\|^2 + \|V_t\|^2 \\ &+ \left\|V_j\right\|^2) \end{aligned} \tag{2.14}$$

where $\|U_u\|^2, \|V_i\|^2, \|V_t\|^2$ and $\left\|V_j\right\|^2$ are regularization terms to avoid overfitting, α, β, γ are control coefficients, and λ_θ is the regularization parameter. For user u, r_{uit}, r_{utj} and r_{uij} represent user u's preference differences between item i and item t, item t and item j, item i and item j, respectively. In Eq. (2.14), c_{uit}, c_{utj} and c_{uij} are three confidence coefficients, which are defined as

$$c_{uit} = \frac{1 + s_{ui}}{1 + s_{ut}} \tag{2.15}$$

$$c_{utj} = 1 + s_{ut} \tag{2.16}$$

$$c_{uij} = 1 + s_{ui} \tag{2.17}$$

where s_{ui} and s_{ui} are support coefficients. The support coefficient s_{ui} is calculated by

$$s_{ui} = \sum_{w \in N_u} sim(u,w) \cdot \delta(i \in P_w) \tag{2.18}$$

where $sim(u,w)$ represents the similarity between u and w, $\delta(*)$ is the indicator function, its value is equal to 1 when $*$ is true otherwise 0.

CPLR uses the SGD optimization algorithm to learn the parameters. The parameters could be updated according to the following rule:

$$\theta \leftarrow \theta - \eta \frac{\partial f(u,i,t,j)}{\partial \theta} \tag{2.19}$$

where θ can be U_u, V_i, V_t, or V_j, and $\eta > 0$ is the learning rate. Learning parameters in CPLR are done by looping over Eq. (2.19).

2.6 Latent Dirichlet Allocation

In 2003, Blei et al. proposed a topic model called LDA, a generative probabilistic graphical model. LDA is an unsupervised learning model and has been used for feature extraction [76], textual classification [77] and emotion classification [78].

LDA is a generative probabilistic graphical model. We describe some rules about the probabilistic graphical model in order to have a good understanding of the LDA model. A probabilistic graphical model is a graph model with probabilistic. In a probabilistic graphical model [79], the circle with the darkened background color indicates the observable variable, and others are implied variables. Plate notation is often used to describe probabilistic graphical models. The dependencies between many variables can be obtained shortly with it. The boxes are "plates" standing for replicates, which are repeated entities. The dependencies between variables are represented by the direction of the arrows between the variables.

The graphical model for LDA is depicted in Figure 2.5. There are three layers to the LDA representation. In Figure 2.5, *z* and *w* represent the topic and word, T, M and N_d represent the number of topics, documents and words, respectively. $\theta \in \mathbb{R}^{M \times T}$ represents a multinomial distribution over topics under documents. $\Phi \in \mathbb{R}^{N_d \times T}$ represents a multinomial distribution over words under topics. α and β represent dirichlet hyper parameters for θ and Φ, respectively.

LDA models the generative process of a document as follows.

1. For each document, sample the topic proportions θ from Dirichlet distribution $Dir(\alpha)$.
2. For each word in the document *m*
   a. Choose a topic *z* randomly according to the sampled topic proportions θ.
   b. Choose a word w randomly from the multinomial distribution Φ of topic z.

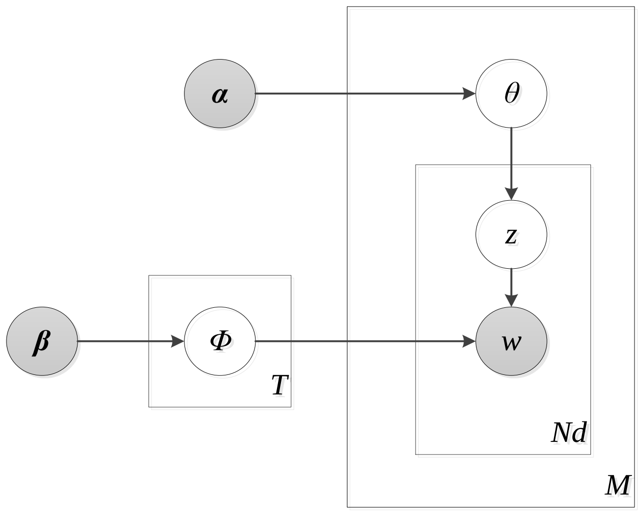


Figure 2.5 A graphical model representation of Latent Dirichlet Allocation

Two approaches are used to estimate parameters in the LDA model. One is variational inference, including mean-field and expectation propagation, and Gibbs sampling [80]. In this thesis, we use Gibbs sampling to estimate parameters in the LDA model. It is a Markov chain Monte Carlo (MCMC) algorithm and can obtain a sequence of observations that are approximated from a specified multivariate probability distribution when direct sampling is difficult to realize.

The probability of topic *z* for the *i*-th word in document d_m, conditioned on the other variables, is given as:

$$P(z_i = z \mid \boldsymbol{z}_{-i}, w) \propto \frac{\{n_z^w\}_{-i} + \beta}{\{n_z\}_{-i} + N_d\beta} \times \frac{\{n_m^z\}_{-i} + \alpha}{\{n_m\}_{-i} + T\alpha} \tag{2.20}$$

where $\{n_z^w\}_{-i}$ represents the number of word *w* in topic *z* in all documents except for the *i*-th word in document d_m, $\{n_z\}_{-i}$ counts the number of words in topic *z* in all documents. $\{n_m^z\}_{-i}$ presents the number of word *w* in topic *z* in document d_m except for the *i*-th word in document d_m, and $\{n_m\}_{-i}$ counts the number of words in document d_m. During the process of Gibbs sampling, Eq. (2.20) is used to update parameters in the model.

2.7 Summary

In this section, we described related knowledge of this thesis.

The algorithm matrix factorization has been widely used to solve problems in recommender systems. It will be used to predict users' preference scores on their unobserved items in this thesis.

The pairwise learning model BPR is the fundamental of our proposed recommendation models; its extensions GBPR and CPLR will be treated as benchmark methods for comparison in Section 5. The well-known optimization algorithm SGD will be used to learn parameters in all recommendation models, including BPR, GBPR. CPLR and the proposed models.

In addition, the topic model LDA plays an important role in calculating users' potential preference scores on their unobserved items. More details will be shown in Section 3.

3 Calculation Methods of Potential Preference Scores

We first give the problem definition. Our proposed recommendation models are designed taking into account the relative preference of each user on any two unobserved items. Therefore, prior information is needed to distinguish users' preference differences between their unobserved items. In this section, we focus on studying the calculation methods of potential preference scores (prior information) between users and their unobserved items, including collaborative filtering-based method and latent dirichlet allocation-based method, based on their historical interactions.

3.1 Problem Definition

We use $U^{tr} = \{u\}_{u=1}^{M}$ and $I^{tr} = \{i\}_{i=1}^{N}$ to denote the sets of users and items and denote a user-item interaction matrix as $R \in \mathbb{R}^{M \times N}$, where M and N represent the number of users and items, respectively. We use R_{ui} to record the interaction between user u and item i. $R_{ui} > 0$ indicates user u has interacted with item i, and $R_{ui} = 0$ indicates user u has no interaction with item i. Our goal is to recommend a ranking list of unobserved items to users.

3.2 Potential Preference Scores based on Collaborative Filtering

3.2.1 Motivation

We find that the historical interactions between users and items provide resources about users' partial preferences. Specifically, we could discover a user's potential preference on his/her unobserved items, according to some explainable connections between observed items and unobserved items. For example, a recommender system contains three users (*A*, *B*, and *C*) and five movies (*a*, *b*, *c*, *d*, and *e*). The interactions between users and items are shown in Figure 3.1. Movie *b* is a nice choice while

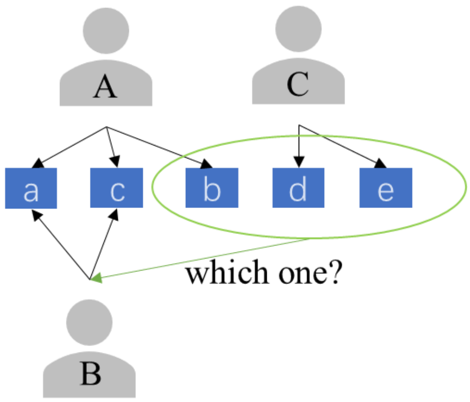


Figure 3.1 Illustration of the example of three users' implicit feedback on five movies

recommending a new movie for user *B*. Considering that user *B* has the same historical interactions (*a* and *c*) with user *A*, user *A* can be considered the like-minded user of user *B*. Thus, user *B* may be interested in the movie, which has been watched by user *A*.

Additionally, users' potential preference scores on items provide resources about potential interactions between users and items, which can alleviate the data sparsity problem and can also be used to evaluate the fine-grained preference difference between any two unobserved items.

3.2.1 Calculation methods of user similarities and item similarities

We employ cosine similarity to measure the similarity between users and the similarity between items. Each item contains many entities, like the movie contains the information of its director, actors and genre. The user might be interested in the item or the entity. Inspired by this, we calculate the user similarity at the item level and the entity level. We also calculate the user-based and entity-based item similarity.

The item-based user similarity (Sim_{user}^{item}) and the user-based item similarity (Sim_{item}^{user}) can be calculated by the interaction matrix *R*. The user similarity between user *u* and user *w* and the item similarity between item *i* and item *j* are defined as follows.

$$Sim_{user}^{item}(u,w) = \frac{\sum_k R_{u_k} \cdot R_{w_k}}{\sqrt{\sum_k R_{u_k}^2}\sqrt{\sum_k R_{w_k}^2}} \tag{3.1}$$

$$Sim_{item}^{user}(i,j) = \frac{\sum_k R_{:,i_k} \cdot R_{:,j_k}}{\sqrt{\sum_k R_{:,i_k}^2}\sqrt{\sum_k R_{:,j_k}^2}} \tag{3.2}$$

The entity-based user similarity is calculated based on the user-entity matrixes ($M_{ue1}, \dots, M_{ueQ}$) of different relations (The movie is directed by *A*. “directed by” is the relation), where Q represents the number of relations. Each value in the user-entity matrix is the count of entities in the observed items. The entity-based user similarity between user *u* and user *w* can be calculated by the average of similarities under different relations as follows.

$$Sim_{user}^{entity}(u,w) = \frac{1}{Q}\sum_{r=1}^{Q} \frac{\sum_q M_{uer_{u_q}} \cdot M_{uer_{w_q}}}{\sqrt{\sum_q {M_{uer_{u_q}}}^2}\sqrt{\sum_q {M_{uer_{w_q}}}^2}} \tag{3.3}$$

Following the previous work, we calculate the item similarity at the entity level based on the item-entity matrixes ($M_{ie1}, \dots, M_{ieQ}$) of different relations, where Q also represents the number of relations. The corresponding value in the item-entity matrix is equal to 1 when the item has a relation with the entity. Firstly, we compute the commuting matrix of each relation *r* as follows:

$$P_r = M_{ier} \cdot M_{ier}^T \tag{3.4}$$

where · represents multiplication. The commuting matrix of all relations can be calculated by the Hadamard product of all matrixes ($P_1, \dots, P_r, \dots, P_Q$), which is defined as

$$P_{all} = P_1 \odot, \dots, \odot P_r \odot, \dots, \odot P_Q \tag{3.5}$$

After obtaining the commuting matrix P_{all}, the entity-based item similarity between item *i* and item *j* is defined as follows.

$$Sim_{item}^{entity}(i,j) = \frac{2 \cdot P_{all_{i,j}}}{(P_{all_{i,i}} + P_{all_{j,j}})} \tag{3.6}$$

3.2.2 Calculation methods of potential preference scores

We calculate the potential preference score matrix S ($S \in \boldsymbol{R}^{M \times N}$) based on user similarities and item similarities. Five users who have the highest user similarities are considered as each user’s like-minded users. The potential preference score matrix (S_u') from each user’s historical interactions is defined as Eq. (3.7). The potential preference score matrix (S''_u) from each user's like-minded users is defined as Eq. (3.8). The final potential preference score matrix (S_u) of each user *u* is defined as Eq. (3.9).

$$S'_u = \frac{R_u \cdot Sim_{item}}{\sum_i R_{ui}} \tag{3.7}$$

$$S''_u = \frac{1}{|G_u|} \sum_{w \in G_u} \left(Sim_{user}(u, w) \cdot S'_w\right) \tag{3.8}$$

$$S_u = \frac{1}{2} S'_u + \frac{1}{2} S''_u \tag{3.9}$$

where G_u represents the group of *u*'s, like-minded users. In S'_u, S''_u and S_u, the corresponding values are fixed as 1 when items are observed by the user. In Table 3.1, we use abbreviations to represent different integration strategies.

Table 3.1 Abbreviations of different integration strategies for calculating potential preference scores

Sim_{user}	Sim_{item}	Abbr	Sim_{user}	Sim_{item}	Abbr
Entity-based	Entity-based	*UEIE*	Entity-based	User-based	*UEIU*
Item-based	Entity-based	*UIIE*	Item-based	User-based	*UIIU*

3.2.3 Performance of CF-based methods

We discuss the performance of potential preference scores calculated based on different integration strategies. We use the real-world dataset MovieLens 1M [81] for experiments, which will be introduced in Section 5. We randomly sampled 20%, 50% and 80% user-item interactions as training data and the rest as test data.

For evaluating the performance of different methods, we use top-*N* evaluation metrics, including top-*N* results of precision, and normalized discounted cumulative gain (NDCG), which will also be described in Section 5. The results are shown in Table 3.2. The numbers in boldface are the best results among all methods, from which we can see that:

- The results based on *UEIU* are the best on the dataset MovieLens (20%), and the results based on *UIIU* perform the best among all methods on the datasets MovieLens (50%) and MovieLens (80%).
- *UEIU* performs well when the sparsity of the user-item interaction matrix is high. The gap between *UEIU* and *UIIU* on the dataset MovieLens (20%) can be ignored considering the simplicity of *UIIU*.

Therefore, we adopt the *UIIU* method to calculate users' potential preference scores (S) for distinguishing relative preference for each user's unobserved items in recommendation models.

Table 3.2 The results of different integration strategies methods

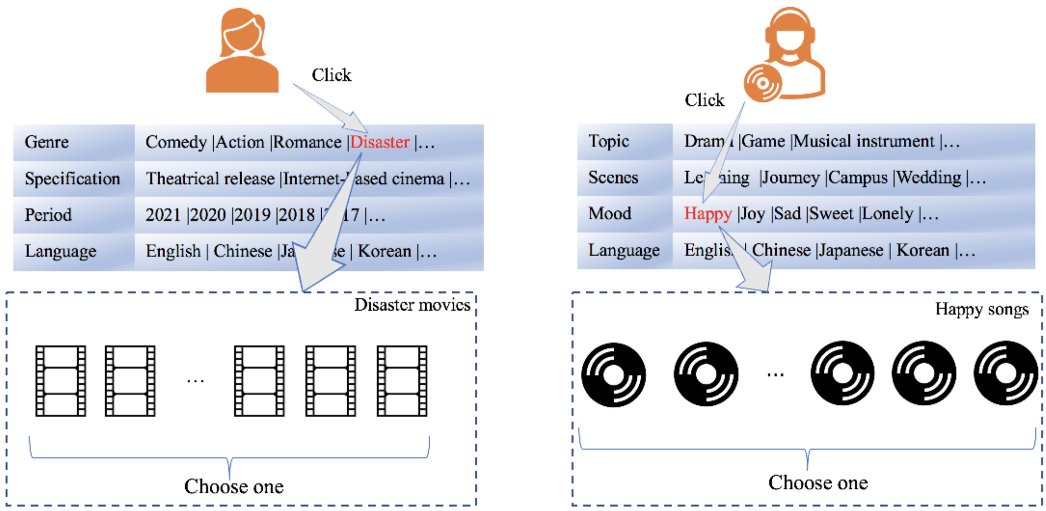
Datasets	Methods	Prec@5	NDCG@5	NDCG@10
MovieLens 1M (20%)	*UEIE*	0.2971	0.3098	0.2835
	UIIE	0.2512	0.2623	0.2419
	UEIU	**0.4074**	**0.4144**	**0.3929**
	UIIU	0.3917	0.3861	0.3739
MovieLens 1M (50%)	*UEIE*	0.3267	0.3393	0.3076
	UIIE	0.2945	0.3082	0.2784
	UEIU	0.4185	0.4286	0.3999
	UIIU	**0.4294**	**0.4379**	**0.4097**
MovieLens 1M (80%)	*UEIE*	0.2303	0.2427	0.2258
	UIIE	0.2149	0.2281	0.2105
	UEIU	0.3143	0.3350	0.3095
	UIIU	**0.3274**	**0.3471**	**0.3220**

3.3 Potential Preference Scores based on LDA

3.3.1 Motivation

We leverage the *UIIU* method to calculate potential preference scores between users and items for further distinguishing each user's preference difference between each pair of unobserved items. It provides an explainable reason for why the user might have an interest in the items with higher potential preference scores. For instance, recommending item 1 to user 2 might be explained by "$item\ 1 \underset{\rightarrow}{\text{watched}}\ user\ 1 \underset{\rightarrow}{\text{like} - \text{minded}}\ user\ 2$". However, for a user, the potential preference scores of the user's unobserved items might be all equal to zero when lacking the information about like-minded users and similarities between items.

We observe that users have their personalized chosen intentions when they choose items on a service system. Learning these intentions can help to find items users might like and give explainable recommendations.



(a) (b)

Figure 3.2 Illustration of examples that (a) the user chooses a movie in movie application, (b) the user chooses a song in music listening application

Examples. Suppose Amy is a regular user of a movie application. She likes to choose movies according to their genres when she does not have a clear goal. She prefers disaster movies to other movies, so she first clicks the disaster option under the "genre". Then, she chooses a movie she has an interest in it. Figure 3.2 illustrates the above process and how Amy chooses a song in the music listening application. Amy likes to choose songs according to her current mood. She clicks the happy option under the “Mood” and chooses a song.

Amy’s intentions are clear in the above examples when choosing a movie or a song, i.e., according to the movie’s genre or the emotion of the music. We assume users have their personalized chosen intentions on different service systems and consider most people’s chosen intention about items have continuity and do not change suddenly. We recommend items for users only by tracking the chosen intentions from users’ historical interactions without using information about like-minded users and similarities between items. Convincing recommendation results can be obtained with the learning results of users’ chosen intentions.

Motivated by these considerations, we propose a novel calculation method to give more accurate recommendations. We treat users’ chosen intentions as the latent information and model the generating process of users’ historical interactions to summarize each user’s personalized chosen intentions, which is explainable. Users’ potential preference scores on their unobserved items could be obtained.

3.3.2 User's chosen intention

As mentioned before, in this paper, we propose a novel calculation method of potential preference scores considering the user's chosen intention, the definition of which is as follow:

Definition (user's chosen intention) The user's chosen intention is the user's personalized behavior when he/she chooses items. For example, a user chooses a movie according to the movies' genre (comedy, action, romance, etc.) when he/she does not have a clear goal in a movie application. Here, "movie genre" is the user's chosen intention.

3.3.3 Calculation method of potential preference scores

Table 3.3 Notations for the LDA model

Symbols	Descriptions
M	the number of users
N_u	the number of user u's historical items
K	the number of chosen intentions
f	chosen intention
u	user
i	item
ϑ	user-chosen intention distribution
$\emptyset$	chosen intention-item distribution
α	dirichlet hyper parameter for ϑ
β	dirichlet hyper parameter for $\emptyset$

For M users, we assume each user's N_u historical items are created from the user's K chosen intentions and leverage LDA to model the generating process of users' historical interactions. LDA samples user-chosen intention distribution ϑ ($\vartheta \in \mathbb{R}^{M_u \times K}$) and chosen intention-item distribution $\emptyset$ ($\emptyset \in \mathbb{R}^{K \times N}$) according to two dirichlet hyper parameters (α and β). Then, the user chooses chosen intention *f*. Based on chosen intention *f*, the user chooses item *i*. The generative process of users' historical items is defined in Algorithm 1.

Algorithm 1 The generative process of users' historical items of LDA modeling

for each user $u \in \{1,2, \dots , M\}$.

  for each chosen intention $f \in \{1,2, \dots , K\}$.

    choose $\vartheta_{u,f} \sim Dir(\alpha)$

for each chosen intention $f \in \{1,2, \dots , K\}$.

  for each item $i \in \{1,2, \dots , N_u\}$.

    choose $\emptyset_{f,i} \sim Dir(\beta)$

for each item of user u.

  choose $f \sim Mul(\vartheta_u)$

  choose $i \sim Mul(\emptyset_f)$

The probability of chosen intention *f* for user *u*'s *n*-th historical item, conditioned on the other variables, is given as:

$$P(f_n = f \mid \boldsymbol{f}_{-n}, i) \propto \frac{\{n_f^i\}_{-n} + \beta}{\{n_f\}_{-n} + N_u\beta} \times \frac{\{n_u^f\}_{-n} + \alpha}{\{n_u\}_{-n} + K\alpha} \tag{3.10}$$

where $\{n_f^i\}_{-n}$ represents the number of item *i* in chosen intention *f* of all users except for the *n*-th historical item of user *u*, and $\{n_f\}_{-n}$ counts the number of historical item *i* in chosen intention *f* of all users. $\{n_u^f\}_{-n}$ presents the number of historical item *i* in chosen intention *f* of user *u* except for the *n*-th historical item of user *u*, and $\{n_u\}_{-n}$ counts the number of historical items of user *u*.

The approximate probability of chosen intention *f* for user *u* is shown in Eq. (3.11).

$$\vartheta_{u,f} = \frac{n_u^f + \alpha}{n_u + K\alpha} \tag{3.11}$$

The approximate probability of item *i* belongs to the chosen intention *f* is shown in Eq. (3.12).

$$\emptyset_{f,i} = \frac{n_f^i + \beta}{n_f + N_u\beta} \tag{3.12}$$

We could calculate user *u*'s potential preference score on item *i* based on two distributions ϑ and $\emptyset$, which is defined as the inner product of ϑ_u and $\emptyset_i$.

3.3.4 Performance of the LDA-based method

We discuss the performance of the LDA-based calculation method inspired by our assumption about users' chosen intentions on items. We provide each user a list of ranking items according to potential preference scores calculated by the inner product of ϑ and $\emptyset$. We first compare the LDA-based recommendation results with existing recommendation methods PopRank, *UIIU* and BPR, which will be introduced in Section 5. Then, we will investigate the impacts of different settings of users' chosen intentions.

We also use three real-world datasets Last-FM, MovieLens 100K and MovieLens 1M, for experiments, which will also be introduced in Section 5.1. We randomly 20% and 50% observed user-item interactions as training data and the rest as test data. For evaluating the performance of different methods, we use top-*N* evaluation metrics, including top-*N* results of precision and normalized discounted cumulative gain (NDCG).

In our experiments, the iteration number is set as 3000 for the LDA model. We set hyper parameters as α=0.05, β=0.01. The number of chosen intentions is fixed as *K*=20 for all three datasets. For BPR, we employ a grid search in {0.02, 0.05, 0.1} to find the best learning rate η for three datasets. The dimension of the latent feature vector and the regularization parameter is fixed as 20 and 0.01, respectively. The value of the iteration number is set to 10,000.

The results of all recommendation methods are shown in Table 3.4 and Table 3.5. The best result on each dataset is displayed in bold, from which we could see that:

(1) LDA achieves more accurate recommendation results than other recommendation methods in terms of Prec@5, NDCG@5 and NDCG@10 in all cases, especially on the dataset Last-FM. This further shows the benefit of performing each user's prediction based on the assumption of users' chosen intentions.

(2) In Table 3.4 and Table 3.5, it is obvious that LDA performs better than *UIIU* significantly on all datasets. The reason may be that the LDA-based calculation method does not suffer from the problem caused by lacking information about like-minded users and similarities between items in the *UIIU* method.

Table 3.4 Performance comparison for PopRank, *UIIU*, BPR and LDA in terms of Prec@5, NDCG@5 and NDCG@10 on datasets Last-FM (20%), MovieLens 100K (20%) and MovieLens 1M (20%)

Datasets	Methods	Prec@5	NDCG@5	NDCG@10
Last-FM (20%)	PopRank	0.1588	0.1314	0.1441
	UIIU	0.1751	0.1794	0.1637
	BPR	0.2874	0.2947	0.2819
	LDA	**0.3453**	**0.3380**	**0.3184**
MovieLens 100K (20%)	PopRank	0.4838	0.4960	0.4702
	UIIU	0.4592	0.4729	0.4402
	BPR	0.4880	0.4975	0.4769
	LDA	**0.5519**	**0.5677**	**0.5295**
MovieLens 1M (20%)	PopRank	0.3611	0.3362	0.3528
	UIIU	0.3917	0.3861	0.3739
	BPR	0.5145	0.5213	0.4999
	LDA	**0.5281**	**0.5196**	**0.5103**

Table 3.5 Performance comparison for PopRank, *UIIU*, BPR and LDA in terms of Prec@5, NDCG@5 and NDCG@10 on datasets Last-FM (50%), MovieLens 100K (50%) and MovieLens 1M (50%)

Datasets	Methods	Prec@5	NDCG@5	NDCG@10
Last-FM (50%)	PopRank	0.1434	0.1538	0.1501
	UIIU	0.2740	0.2870	0.2596
	BPR	0.3346	0.3500	0.3062
	LDA	**0.3573**	**0.3725**	**0.3239**
MovieLens 100K (50%)	PopRank	0.3862	0.4009	0.3713
	UIIU	0.5007	0.5201	0.4771
	BPR	0.5461	0.5613	0.5266
	LDA	**0.5648**	**0.5820**	**0.5394**
MovieLens 1M (50%)	PopRank	0.3301	0.3454	0.3293
	UIIU	0.4294	0.4379	0.4097
	BPR	0.5092	0.5177	0.4912
	LDA	**0.5214**	**0.5317**	**0.5017**

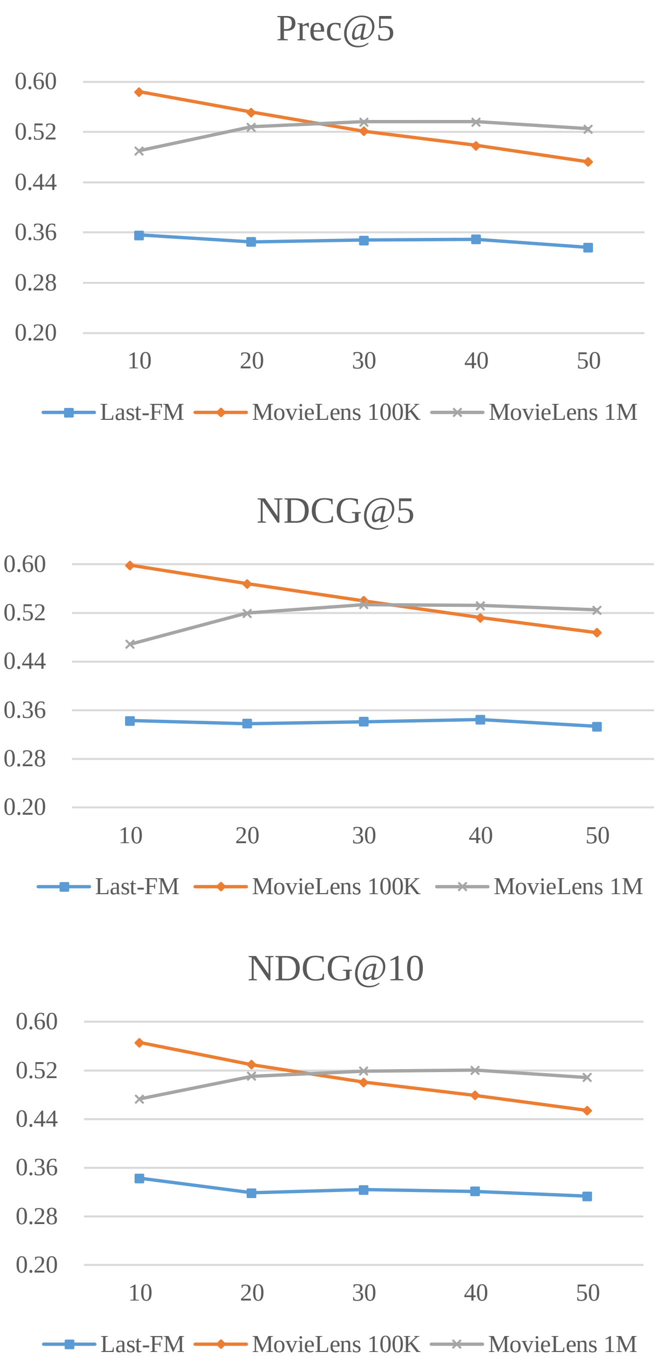


Figure 3.3 Results of different chosen intention numbers in terms of Prec@5, NDCG@5 and NDCG@10 on datasets Last-FM (20%), MovieLens 100K (20%) and MovieLens 1M (20%)

The number of chosen intentions represents the complexity of the LDA model; the descriptions of users and items are limited when the number of chosen intentions is small. And the complexity of the LDA model will be increased with the increase of the chosen intention number. We conduct experiments to study the effects of different chosen intention numbers on the performance of the model on three datasets.

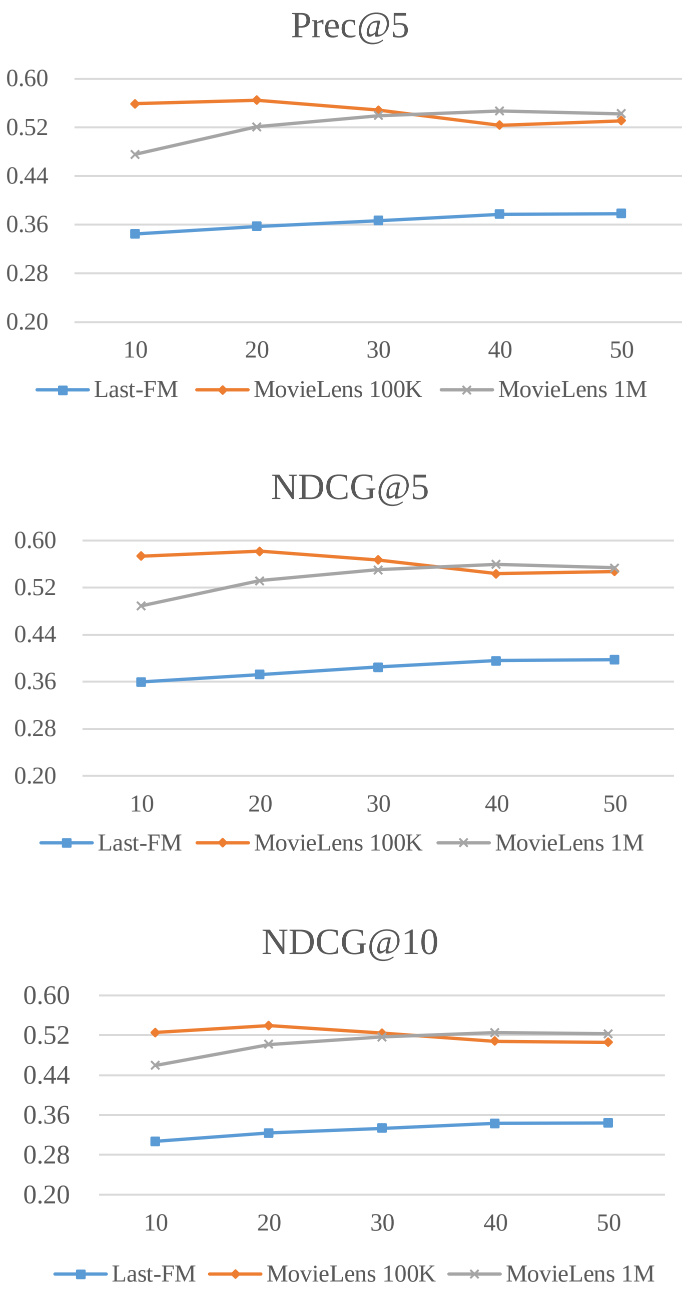


Figure 3.4 Results of different chosen intention numbers in terms of Prec@5, NDCG@5 and NDCG@10 on datasets Last-FM (50%), MovieLens 100K (50%) and MovieLens 1M (50%)

To explore the effects of chosen intentions (K) on recommendation results, we fuse different chosen intentions $K = \{10, 20, 30, 40, 50\}$ into the LDA model. Figure 3.4 illustrates the results across all evaluation metrics on datasets Last-FM, MovieLens 100K and MovieLens 1M.

Figure 3.5 Illustration of the representation matrix of the one-class feedback data with user's potential preference scores

In Figure 3.3 and Figure 3.4, the performance of the LDA-based method behaves differently on three datasets when chosen intentions are adjusted in $\{10, 20, 30, 40, 50\}$. We observe that optimal results on different datasets are obtained with different values.

3.4 Summary

In this section, we studied two calculation methods of users' potential preference scores on their unobserved items based on users' historical interactions for further studying, including the *UIIU*-based method and the LDA-based method.

Overall, the potential preference scores calculated by the *UIIU*-based method are the aggregation of each user's like-minded users and the similarities between items. It suffers from the data sparsity problem. Existing users' like-minded users might not be found when the user-item interaction matrix has high sparsity. More user-item interactions could obtain more accurate recommendations. Different from the *UIIU*-based method, the potential preference scores calculated by the LDA-based method only need each user's own historical interactions.

We calculate users' potential preference scores to distinguish the preference differences between their unobserved items. Actually, we only need the relative preference difference between unobserved items. This process could be realized by the more simple strategy such as distinguishing the potential preference differences between items according to item popularity, i.e., the item that has higher popularity among users may attract more attention than another item. Moreover, the specific potential preference scores could not only be used to measure the fine-grained differences between unobserved items but also be taken into account in the calculation process for more accurate results, such as considering the potential preference scores into the prediction function or using the potential preference scores to adjust the confidence of pairwise preferences.

As we mentioned in Section 1 that CF-based recommendation methods usually suffer from the sparsity of user-item interactions. We consider this problem could be relaxed with the potential preference scores because users' potential preference scores provide more potential interactions between users and items. An example of the representation matrix of the one-class feedback data with the user's potential preference scores is shown in Figure 3.5. The grey blocks in the representation matrix are recorded users' potential preference scores on their unobserved items.

4 Recommendation Models for One-class Collaborative Filtering

The previous section introduced different calculation methods of users' potential preference scores and analyzed their performances. The relative preference of users' unobserved items could be distinguished according to potential preference scores. With the prior information, three extension models of BPR (PBPR, PBPR's improved model PBPR* and DBPL) are studied for more accurate recommendation results. We will describe the objective functions and learning methods of three recommendation models in this section.

4.1 Prior-based Bayesian Pairwise Ranking

Prior-based Bayesian Pairwise Ranking (PBPR) is proposed with the assumption that users' potential preferences on their unobserved items are different. It is also an extension model of BPR and does not need any social information, for datasets containing user-item interactions. Moreover, we measure the fine-grained preference differences between unobserved items of the user according to potential preference scores, rather than dividing unobserved items into different groups and distinguishing the preference difference between unobserved items from different groups.

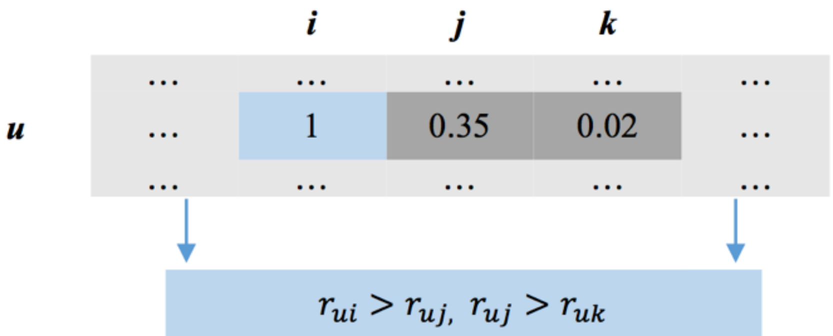


Figure 4.1 Illustration of relative preference assumption based on users' preference scores

Table 4.1 Notations for the proposed PBPR model

Symbols	Descriptions
S	potential preference score matrix
u	a user
i	an observed item of user *u*
j	an unobserved item of user *u*
k	an unobserved item of user *u* and $S_{uj} > S_{uk}$
U^{tr}	the set of users
U_j^{tr}	the set of users who have interest in item *j*
U_u	the latent factor vector of user *u* ($U_u \in \mathbb{R}^d$)
V_i	the latent factor vector of item *i* ($V_i \in \mathbb{R}^d$)
I_u^+	observed items of user *u*
$I^{tr} \backslash I_u^+$	unobserved items of user *u*
w	a user who has observed item *j*
r_{ui}	user *u*'s preference on item *i*
r_{uij}	user *u*'s preference difference between item *i* and item *j*
λ_1^p	a trade-off parameter used to fuse users' (*u* and *w*) preferences on item *j*
λ^p	a control coefficient used to fuse r_{uij} and r_{Nujuk}
β^p	the hyper-parameter to tune the regularization terms
η	the learning rate

4.1.1 Objective function

Besides observed items I_u^+, the remaining items which are unobserved by users could be attributed to two reasons: the user has not seen these items, or the user is not interested in these items [57]. Different from the assumption adopted by the BPR with equal importance of the unobserved items, we consider user's preferences on unobserved items are not equal, and the preference difference between unobserved items could be measured by potential preference scores between users and items. We give an illustration of preference assumption based on prior information in Figure 4.1. For user *u*, we record "1" for his/her observed items and record the user's potential preference scores on unobserved items in the grey area with a value varied from 0 to 1. We consider that item *j* has a higher possibility of attracting the user *u*'s attention than item *k* according to their potential preference scores. $r_{uj} > r_{uk}$ represents the preference difference between item *j* and item *k*. Inspired by this assumption, we can simply extend BPR to our proposed new model called Prior-based Bayesian Pairwise Ranking (PBPR). The tentative likelihood for users and items could be written as follows:

$$\text{PBPR} = \prod_{u \in U^{tr}} \prod_{i \in I_u^+} \prod_{j,k \in I^{tr} \setminus I_u^+} Pr(r_{ui} > r_{uj}, r_{uj} > r_{uk}) \left[1 - Pr(r_{ui} < r_{uj}, r_{uj} < r_{uk})\right] \tag{4.1}$$

where I_u^+ represents observed items of user u. j and k are two random sampled unobserved items of user u, and item j is more likely to be observed by the user than item k. r_{ui} represents user u's preference on item i, which can be calculated by the inner product between the latent factor vector of user u ($U_u \in \mathbb{R}^d$) and the latent factor vector of item i ($V_i \in \mathbb{R}^d$), i.e., $r_{ui} = U_u^T \cdot V_i$, where d represents the number of latent dimensions.

The shortcoming of PBPR in Eq. (4.1) is that its performance largely depends on the accuracy of potential preference scores. We consider that when the individual preference r_{uj} is fused with an individual preference r_{wj} ($w \in U_j^{tr}$, U_j^{tr} represents the set of users who observe item j), the fused preference r_{Nuj} is more likely to be higher than the individual preference r_{uk} compared with the individual preference r_{uj}. Based on the above consideration, the likelihood of our proposed PBPR model could be given by:

$$\text{PBPR} = \prod_{u \in U^{tr}} \prod_{i \in I_u^+} \prod_{j,k \in I^{tr} \setminus I_u^+} Pr(r_{ui} > r_{uj}, r_{Nuj} > r_{uk}) \left[1 - Pr(r_{ui} < r_{uj}, r_{Nuj} < r_{uk})\right] \tag{4.2}$$

where $r_{Nuj} = \lambda_1^p r_{wj} + (1 - \lambda_1^p) r_{uj}$ represents the fused preferences on item j, λ_1^p is a trade-off parameter used to fuse two different users' preferences on item j.

4.1.2 Learning the model

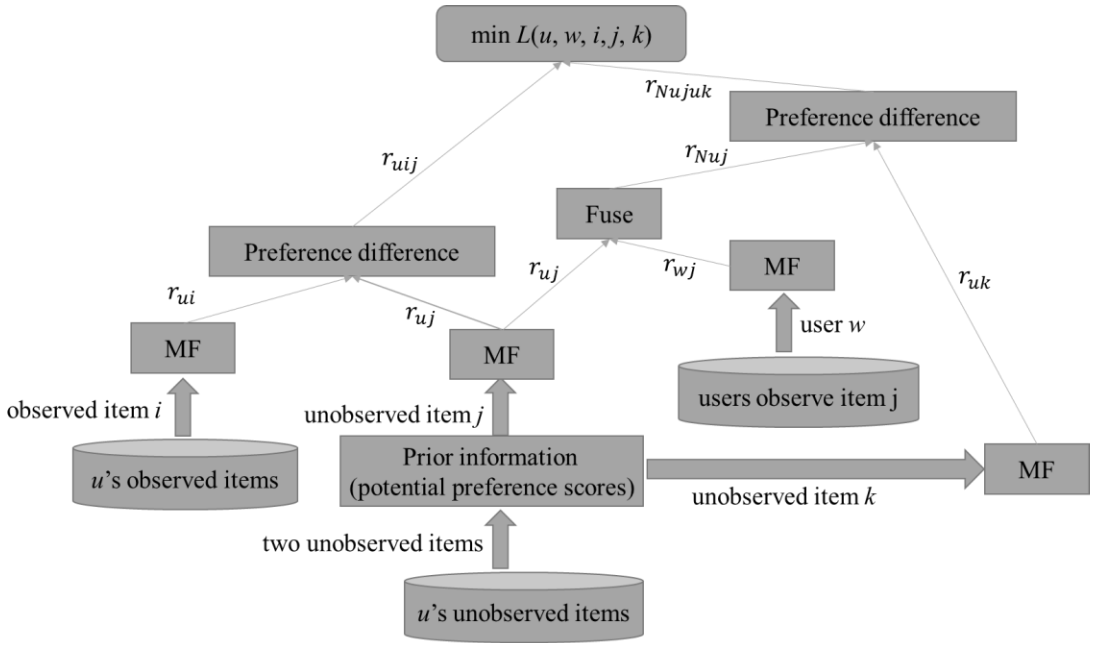


Figure 4.2 Illustration of each randomly sampled record in PBPR

We represent $r_{ui} > r_{uj}$, $r_{N_{uj}} > r_{uk}$ as $\lambda^p(r_{ui} - r_{uj}) + (1-\lambda^p)(r_{N_{uj}} - r_{uk})$, where λ^p is a control coefficient used to fuse their relations. To maximize the posterior probability for PBPR, we employed $\sigma(x) = 1/(1+\exp(-x))$ to approximate the probability $Pr(\cdot)$, following the BPR algorithm, where σ represents the logistic sigmoid function. The log-likelihood is also employed to reduce the calculation complexity of PBPR. For each randomly sampled record, it includes a user u, an observed item i of user u, two unobserved items j, k of user u, and a user w, where $w \in U_j^{tr}$. The illustration of this process is shown in Figure 4.2. For each user u, an observed item i and two unobserved items are selected. The unobserved item with a higher potential preference score is item j, another is item k. User w has positive feedback on item j.The objective function can be written as:

$$L(u,w,i,j,k) = -\ln\sigma\left(\lambda^p\left(r_{ui} - r_{uj}\right) + (1-\lambda^p)\left(r_{N_{uj}} - r_{uk}\right)\right) + \frac{\beta^p}{2}\left(\|U_u\|^2 + \|U_w\|^2 + \|V_i\|^2 + \left\|V_j\right\|^2 + \|V_k\|^2\right) \tag{4.3}$$

where $\|U_u\|^2$, $\|U_w\|^2$, $\|V_i\|^2$, $\left\|V_j\right\|^2$ and $\|V_k\|^2$ are regularization terms to prevent overfitting in the learning process, and β^p is the hyper-parameter to tune the regularization terms. The individual preference score is modeled by matrix factorization.

Following the well-known stochastic gradient descent (SGD) algorithm, each parameter could be updated as follows,

$$\theta \leftarrow \theta - \eta \frac{\partial L(u, w, i, j, k)}{\partial \theta} \tag{4.4}$$

where θ can be U_u, U_w, V_i, V_j, or V_k, and $\eta > 0$ is the learning rate.

We represent the fuse of the preference difference between r_{ui} and r_{uj}, and the preference difference between r_{Nuj} and r_{uk} as $r_{uiuj;\,Nujuk}$. So, we can derive the following gradients for users:

$$\frac{\partial L(u, w, i, j, k)}{\partial U_u} = \frac{\partial L(u, w, i, j, k)}{\partial r_{uiuj;Nujuk}} \times \left(\lambda^p V_i + \left((1-\lambda^p)(1-\lambda_1^p) - \lambda^p\right) V_j - (1-\lambda^p) V_k\right) + \beta^p U_u \tag{4.5}$$

$$\frac{\partial L(u, w, i, j, k)}{\partial U_w} = \frac{\partial L(u, w, i, j, k)}{\partial r_{uiuj;Nujuk}} \times \left(\lambda_1^p (1-\lambda^p) V_j\right) + \beta^p U_w \tag{4.6}$$

and we can get the gradients for items as follows:

$$\frac{\partial L(u, w, i, j, k)}{\partial V_i} = \frac{\partial L(u, w, i, j, k)}{\partial r_{uiuj;Nujuk}} \times (\lambda^p U_u) + \beta^p V_i \tag{4.7}$$

$$\frac{\partial L(u, w, i, j, k)}{\partial V_j} = \frac{\partial L(u, w, i, j, k)}{\partial r_{uiuj;Nujuk}} \times \left(\left((1-\lambda^p)(1-\lambda_1^p) - \lambda^p\right) U_u + \lambda_1^p (1-\lambda^p) U_w\right) + \beta^p V_j \tag{4.8}$$

$$\frac{\partial L(u, w, i, j, k)}{\partial V_k} = \frac{\partial L(u, w, i, j, k)}{\partial r_{uiuj;Nujuk}} \times (-(1-\lambda^p) U_u) + \beta^p V_k \tag{4.9}$$

where $\frac{\partial L(u,w,i,j,k)}{\partial r_{uiuj;Nujuk}} = -\frac{\exp(-r_{uiuj;Nujuk})}{1+\exp(-r_{uiuj;Nujuk})} = -\frac{1}{1+\exp\,(r_{uiuj;Nujuk})}$.

4.1.3 The PBPR algorithm

The steps of PBPR are summarized in Algorithm 2 in detail. We divide PBPR into two steps for notational simplicity: user and item sampling (lines 4-7) and parameter update (lines 8-13).

Algorithm 2 Learning parameters for the PBPR model

Input:

User-item interaction matrix *R*.

Potential preference score matrix *S*.

Parameters η, λ^p, λ_1^p, and β^p.

Output:

The learned model parameters $\{U_u, V_i, u \in U^{tr},\ i \in I^{tr}\}$.

1. Randomly initialize *U* and *V*.
2. **For** $t_1 = 1, \dots, T$
3.   **For** $t_2 = 1, \dots, M$
4.   Randomly pick a user $u \in U^{tr}$.
5.   Randomly pick an item $i \in I_u^+$.
6.   Randomly pick two items $j, k \in I^{tr} \backslash I_u^+$, $S_{uj} > S_{uk}$.
7.   Randomly pick a user $w \in U_j^{tr}$.
8.   Calculate r_{Nuj} and $\frac{\partial L}{\partial r_{uiuj;Nujuk}}$.
9.   Update U_u via Eq. (4.5).
10.   Update U_w via Eq. (4.6).
11.   Update V_i via Eq. (4.7).
12.   Update V_j via Eq. (4.8).
13.   Update V_k via Eq. (4.9).
14.   **End**
15. **End**

In line 6, we randomly pick two unobserved items *j* and *k* of user *u*, and the unobserved item with higher potential preference scores is considered as unobserved item *j*, another is unobserved item *k*. More details in the sampling process will be discussed in Section 5.

4.1.4 Computational complexity

PBPR needs the potential preference scores of each user to distinguish the preference differences between unobserved items. We use a user-based and an item-based collaborative filtering method, i.e., the *UIIU* method, which considers the similarities between each pair of users and the similarities between each pair of items. The computational complexity of computation for potential preference scores is $O(M^2 + N^2 + |G_u|M)$, where M is the number of users, N is the number of items, and $|G_u|$

is the number of each user's like-minded users.

We need to update the value of $5d$ parameters $\{U_u,\ U_w,\ V_i,\ V_j,\ V_k\}$. Therefore, the computational complexity of the learning parameters in the PBPR is $O(MTd)$, where $|U|$ is also the number of users, T is the number of iterations, and d is the number of latent dimensions. In total, the time complexity of the recommended method is $O(M^2 + N^2 + |G_u|M + MTd)$.

4.2 Improving PBPR

We use potential preference scores obtained from the user's chosen intention learning module to improve the performance of PBPR. In addition, we consider several strategies for more accurate recommendation results. More details will be described in the following subsections.

4.2.1 Strategies for PBPR

We observe some shortcomings in the previous work. Firstly, PBPR needs users' potential preference scores on their unobserved items to distinguish the difference between each pair of unobserved items. In PBPR, potential preference scores were calculated by the user-based and item-based collaborative filtering method. This method could provide an explainable reason why users might have an interest in some items. However, many existing users have few interactions with items. It caused some users' potential preference scores on unobserved items to be zero. So, the second strategy for distinguishing differences between each user's unobserved items was designed to assume that users like popular unobserved items over other unobserved items. During the sampling process, PBPR randomly samples two unobserved items for further studying. Actually, the recommendation results may be influenced when the potential preference difference between two unobserved items is very small because even a human possibly hesitates to make a choice when faced with two similar items. We consider the reasons why two items have close potential preference scores:

(1) the user really has similar preferences on them,

(2) the sparsity of data causes users to have similar preference scores on some items.

For enhancing the performance of PBPR, two strategies are proposed to alleviate the problems above, including:

Table 4.2 Notations for the proposed PBPR* model

Symbols	Descriptions
u	a user
i	an observed item of user u
j	an unobserved item of user u
k	an unobserved item of user u and $S_{uj} > S_{uk}$
w	a user who has observed item j
λ_1^*	a trade-off parameter used to fuse users' (u and w) preferences on item j
λ^*	a control coefficient used to fuse r_{uij} and r_{Nujuk}
β^*	the hyper-parameter to tune the regularization terms
η	the learning rate
ϑ	user-chosen intention distribution
$\emptyset$	chosen intention-item distribution

(1) Instead of the user-based and item-based collaborative filtering method, we leverage the inner product of two distributions obtained in the user's chosen intention learning module to measure the differences between unobserved items. We do not worry about the problem caused by lacking information such as like-minded users in the above situation because the calculation method of potential preference scores based on two distributions only relies on users' historical interactions.

(2) Empirically, we adopt a simple strategy to widen the preference gap between user u's two unobserved items (j and k) randomly picked in the sampling process, which is that at least an unobserved item of user u has been observed by at least N_A users.

4.2.2 Objective function

We denote the modified PBPR as PBPR*. Motivated by [58], the objective function of the PBPR* model can be rewritten as:

$$L(u,w,i,j,k) = -\ln\sigma\left(\lambda^* \cdot c_{uij}\left(r_{ui} - r_{uj}\right) + (1-\lambda^*) \cdot c_{ujk}(r_{N_{uj}} - r_{uk})\right) + \frac{\beta^*}{2}(\|U_u\|^2 + \|U_w\|^2 + \|V_i\|^2 + \left\|V_j\right\|^2 + \|V_k\|^2) \quad (4.10)$$

where c_{uij} and c_{ujk} are confidence coefficients, which are used to adjust the confidence of pairwise preferences, β^* is the hyper-parameter to tune the regularization terms. r_{ui} represents user u's preference on item i, which can be calculated by the inner product between the latent factor vector of user u ($U_u \in \mathbb{R}^d$) and the latent factor vector of item i ($V_i \in \mathbb{R}^d$), i.e., $r_{ui} = U_u^T \cdot V_i$, where d represents the number of latent dimensions.

4.2.3 Learning the model

Following [58], we define Eq. (4.11) to simplify the representation.

$$r_{uij} = c_{uij}(r_{ui} - r_{uj}) \tag{4.11}$$

The optimization problem of the objective function in Eq. (4.10) can also be solved by the SGD algorithm. The parameter θ in PBPR* could be updated as follows:

$$\theta \leftarrow \theta - \eta \frac{\partial L(u,w,i,j,k)}{\partial \theta} \tag{4.12}$$

where θ can be U_u, U_w, V_i, V_j or V_k, and $\eta > 0$ is the learning rate. We then derive the following gradients for users:

$$\frac{\partial L(u,w,i,j,k)}{\partial U_u} = \frac{\partial L(u,w,i,j,k)}{\partial r_{uiuj;Nujuk}} \times \left(\lambda^* \cdot c_{uij} V_i + \left((1-\lambda^*)(1-\lambda_1^*) \cdot c_{ujk} - \lambda^* \cdot c_{uij}\right) V_j - (1 - \lambda^*) \cdot c_{ujk} V_k\right) + \beta^* U_u \tag{4.13}$$

$$\frac{\partial L(u,w,i,j,k)}{\partial U_w} = \frac{\partial L(u,w,i,j,k)}{\partial r_{uiuj;Nujuk}} \times \left(\lambda_1^*(1-\lambda^*) \cdot c_{ujk} V_j\right) + \beta^* U_w \tag{4.14}$$

and we can get the gradients for items as follows:

$$\frac{\partial L(u,w,i,j,k)}{\partial V_i} = \frac{\partial L(u,w,i,j,k)}{\partial r_{uiuj;Nujuk}} \times \left(\lambda^* \cdot c_{uij} U_u\right) + \beta^* V_i \tag{4.15}$$

$$\frac{\partial L(u,w,i,j,k)}{\partial V_j} = \frac{\partial L(u,w,i,j,k)}{\partial r_{uiuj;Nujuk}} \times \left(\left((1-\lambda^*)(1-\lambda_1^*) \cdot c_{ujk} - \lambda^* \cdot c_{uij}\right) U_u + \lambda_1^*(1-\lambda^*) \cdot c_{ujk} U_w\right) + \beta^* V_j \tag{4.16}$$

$$\frac{\partial L(u,w,i,j,k)}{\partial V_k} = \frac{\partial L(u,w,i,j,k)}{\partial r_{uiuj;Nujuk}} \times \left(-(1-\lambda^*) \cdot c_{ujk} U_u\right) + \beta^* V_k \tag{4.17}$$

where $\frac{\partial L(u,w,i,j,k)}{\partial r_{uiuj;Nujuk}} = -\frac{\exp(-r_{uiuj;Nujuk})}{1+\exp(-r_{uiuj;Nujuk})} = -\frac{1}{1+\exp\,(r_{uiuj;Nujuk})}$.

Algorithm 3 Learning parameters for the PBPR* model

Input:

User-item interaction matrix R.

User-chosen intention distribution ϑ.

Chosen intention-item distribution $\emptyset$.

Parameters η, λ^*, λ_1^*, and β^*.

Output:

The learned model parameters $\{U_u, V_i, u \in U^{tr},\ i \in I^{tr}\}$.

1. Randomly initialize U and V.
2. **For** $t_1 = 1, \ldots, T$
3.   **For** $t_2 = 1, \ldots, M$
4.   Randomly pick a user $u \in U^{tr}$.
5.   Randomly pick an item $i \in I_u^+$.
6.   Randomly pick two items $j, k \in I^{tr} \backslash I_u^+$.
           $((U_j^{tr} \geq N_A\ or\ U_k^{tr} \geq N_A)$ and $S_{uj} > S_{uk})$
7.   Randomly pick a user $w \in U_k^{tr}$.
8.   Calculate r_{N_k} and $\frac{\partial L(u,w,i,j,k)}{\partial r_{uiuj;Nujuk}}$.
9.   Update U_u via Eq. (4.13).
10.   Update U_w via Eq. (4.14).
11.   Update V_i via Eq. (4.15).
12.   Update V_j via Eq. (4.16).
13.   Update V_k via Eq. (4.17).
14.   **End**
15. **End**

4.2.4 The PBPR* algorithm

The process of PBPR* is summarized in Algorithm 3 in detail. We divide PBPR* into two steps for notational simplicity: user and item sampling (lines 4-7) and parameter update (lines 8-13).

In line 6, s_{uj} and s_{uk} represent user u's potential preference scores on item j and item k, and item j has been observed by at least N_A users, i.e., $U_j^{tr} \geq N_A$. We fix the user u's potential preference score on each observed item as 1 and use the sigmoid function to map the potential preference score of the unobserved item into the range [0, 1), which is given by Eq. (4.18).

$$s_{uj} = \frac{1}{1 + e^{-\vartheta_u \cdot \phi_j}} \tag{4.18}$$

The confidence coefficients c_{uij} is defined as

$$c_{uij} = \frac{1}{1 + e^{-(s_{ui} - s_{uj})}} \tag{4.19}$$

and the confidence coefficients c_{ujk} is defined as

$$c_{ujk} = \frac{1}{1 + e^{-(s_{Nuj} - s_{uk})}} \tag{4.20}$$

where s_{Nuj} is calculated by $s_{Nuj} = \lambda_1^* s_{wj} + (1 - \lambda_1^*) s_{uj}$.

4.2.5 Computational complexity

In the PBPR*-based recommendation method, the potential preference scores of each user on items are calculated by the LDA-based method. The computational complexity of computation for potential preference scores is $O(KN_{all}T)$, where K is the number of chosen intentions, N_{all} is the number of all interactions in the dataset, and T is the number of iterations setting for the LDA model.

The computational complexity of the learning parameters in the PBPR* is $O(MTd)$. Therefore, the total time complexity of the recommended method is $O(KN_{all}T + MTd)$, where M is the number of users. It is obvious that the LDA-based calculation method takes a lot of time for obtaining potential preference scores when the number of all interactions in the dataset is very large.

4.3 Double Bayesian Pairwise Learning

In this section, we propose a recommendation model called Double Bayesian Pairwise Learning (DBPL). For the vanishing gradient problem caused by the tailed distribution, instead of proposing any sampling methods, we treat each user's preference difference between the observed item and the unobserved item differently according to his/her potential preference on unobserved items and use a relatively smaller preference difference to reduce the current preference difference for further alleviating the problem. DBPL can also run without any additional information except user-item interactions and can be adjusted by the social information from a specific dataset.

Table 4.3 Notations for the proposed DBPL model

Symbols	Descriptions
S	potential preference score matrix
u	a user
i	an observed item of user *u*
m	an observed item of user *u*
j	an unobserved item of user *u*
k	an unobserved item of user *u* and $S_{uj} > S_{uk}$
w	a user who has observed item *k*
λ_1^d	a trade-off parameter used to fuse users' (*u* and *w*) preferences on item *k*
λ^d	a control coefficient used to fuse r_{uiuj} and r_{umNk}
β^d	the hyper-parameter to tune the regularization terms
η	the learning rate

4.3.1 Objective function

In the learning process of BPR, as it can be seen in Eq. (4.21), each gradient step has a multiplicative scalar, which is called the *gradient magnitude* [33] of a sampled case (*u*, *i*, *j*).

$$\Delta_{u,i,j} := \left(1 - \sigma\left(r_{ui} - r_{uj}\right)\right) \tag{4.21}$$

It is clear in Eq. (4.21) that the gradient magnitude $\Delta_{u,i,j}$ is close 0 if the preference difference of user *u* towards the observed item *i* and the unobserved item *j* is very large; nothing can be learned from the sampled case (*u*, *i*, *j*) because its gradient vanishes, i.e., the parameters are not changed by Eq. (4.21).

However, the above situation does not mean that the loss is inappropriate, but that the potential positive items have not been seen by the algorithm of BPR. Therefore, we propose the DBPL model for dealing with this problem.

We relax the assumption adopted by BPR that the equal importance for the unobserved items and use the potential preference matrix *S* to measure preference differences of users' unobserved items. For user *u*, we consider the preference difference r_{uij} between an observed item *i* and an unobserved item *j* is smaller than the preference difference r_{umk} between an observed item *m* and an unobserved item *k*, i.e., $r_{uij} < r_{umk}$ when user *u*'s potential preference score on item *j* is larger than user *u*'s potential preference score on item *k*, i.e., $S_{uj} > S_{uk}$. The preference difference of a pairwise preference (*u*, *m*, *k*) can be reduced by fusing with the pairwise preference (*u*, *i*, *j*). The comparison of preference differences of different pairs of items can be expressed by

$$r_{uij} < \lambda^d r_{uij} + (1 - \lambda^d) r_{umk} < r_{umk} \tag{4.22}$$

where λ^d represents the control coefficient used to fuse two preferences. Based on the above consideration, the tentative likelihood for users and items could be shown as:

$$\mathrm{DBRL} = \prod_{u \in U^{tr}} \prod_{i,m \in I_u^+} \prod_{j,k \in I^{tr} \setminus I_u^+} Pr\left(r_{ui} > r_{uj}, r_{um} > r_{uk}\right) \left[1 - Pr\left(r_{ui} < r_{uj}, r_{um} < r_{uk}\right)\right] \tag{4.23}$$

where I_u^+ represents the set of observed items of user *u*, $I^{tr} \setminus I_u^+$ represents the set of unobserved items of user *u*, item *i* and item m are observed items of user u, item j, and item *k* are unobserved items of user *u*. r_{ui} can be calculated by the inner product between the latent factor vector of user *u* ($U_u \in \mathbb{R}^d$) and the latent factor vector of item *i* ($V_i \in \mathbb{R}^d$), i.e., $r_{ui} = U_u^T \cdot V_i$, where *d* represents the number of latent dimensions.

It is obvious that the performance of our proposed strategy is influenced by the accuracy of potential preference scores. The comparison between unobserved items based on the potential preference matrix might not match the real situation. For increasing the *gradient magnitude* of each case, we consider that the preference score of user *w* who has observed item *k* is higher than user *u*, who does not observe item *k*, i.e., $r_{uk} < r_{wk}$.

The *gradient magnitude* of the case (*u*, *m*, *k*) could be increased by fusing user *u*'s potential preference on item *k* with another user *w*'s preference on item *k*, where user *w* has positive feedback on item *k* ($w \in U_k^{tr}$). Therefore, the comparison of preference differences of different pairs of items can be written as follows

$$\lambda^d r_{uij} + (1 - \lambda^d) r_{umN_k} < \lambda^d r_{uij} + (1 - \lambda^d) r_{umk} < r_{umk} \tag{4.24}$$

where $r_{umN_k} = r_{um} - r_{N_k}$. r_{N_k} represents users' fused preference on item *k,* which is calculated by $r_{N_k} = \lambda_1^d r_{uk} + (1 - \lambda_1^d) r_{wk}$. We can further obtain that

$$\Delta_{u,m,k} < \Delta_{N_u} < \Delta'_{N_u} \tag{4.25}$$

where the quantity Δ_{N_u} represents the fused *gradient magnitude* of the case (u, i, m, j, k), which is calculated by $\Delta_{N_u} := (1 - \sigma(\lambda^d r_{uij} + (1 - \lambda^d) r_{umk}))$. Δ'_{N_u} represents the fused *gradient magnitude* of the case (u, w, i, j, m, k), which is calculated by $\Delta_{N_u} = (1 - \sigma(\lambda^d r_{uij} + (1 - \lambda^d) r_{umN_k}))$. Following BPR, we adopt the matrix factorization technique to predict the preference of user u on item j. The likelihood of DBPL can be given by

$$\text{DBRL} = \prod_{u \in U^{tr}} \prod_{i,m \in I_u^+} \prod_{j,k \in I^{tr} \setminus I_u^+} Pr(r_{ui} > r_{uj}, r_{um} > r_{N_k}) \left[1 - Pr(r_{ui} < r_{uj}, r_{um} < r_{N_k})\right] \tag{4.26}$$

To maximize the posterior probability, DBPL also uses $\sigma(x) = \frac{1}{1+e^{-x}}$ to approximate the probability $Pr(\cdot)$ and adopts the log-likelihood to reduce the computational complexity following BPR. For each randomly sampled record, it includes a user u, two observed item i, m of user u, two unobserved item j, k and a user w, where $w \in U_k^{tr}$. The illustration of this process is shown in Figure 4.3. For each user, two observed items and two unobserved items are selected. The unobserved item with a higher potential preference score is item j, another item k. User w has positive feedback on item k. The objective function can be written as

$$L(u, w, i, j, m, k) = -ln\sigma(\lambda^d (r_{ui} - r_{uj}) + (1 - \lambda^d)(r_{um} - r_{N_k})) + \frac{\beta^d}{2}(\|U_u\|^2 + \|U_w\|^2 + \|V_i\|^2 + \|V_j\|^2 + \|V_m\|^2 + \|V_k\|^2) \tag{4.27}$$

where $\|U_u\|^2, \|U_w\|^2, \|V_i\|^2, \|V_j\|^2, \|V_m\|^2$ and $\|V_k\|^2$ are regularization terms to avoid overfitting. β^d is the regularization parameter for regularization terms.

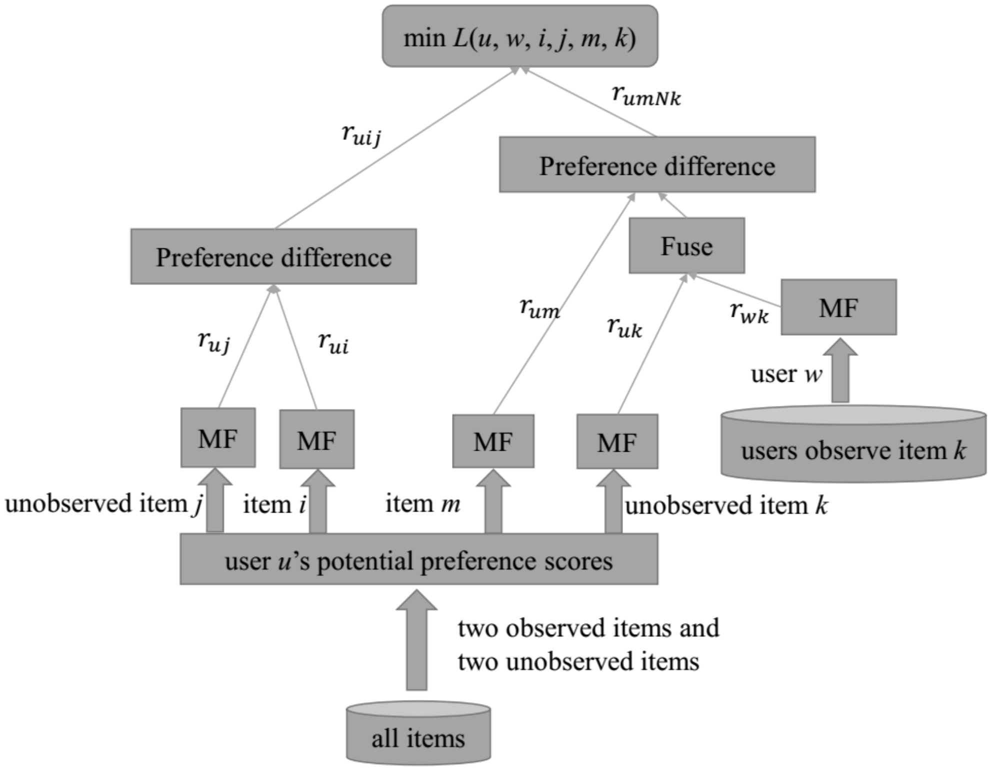


Figure 4.3 Illustration of each randomly sampled record in DBPL

4.3.2 Learning the model

The optimization problem of the objective function in Eq. (4.27) can also be solved by the SGD algorithm. The parameter θ in DBPL could be updated as follows

$$\theta \leftarrow \theta - \eta \frac{\partial L(u,w,i,j,m,k)}{\partial \theta} \tag{4.28}$$

where θ can be U_u, U_w, V_i, V_j, V_m or V_k, and $\eta > 0$ is the learning rate. We then derive the gradients of the user-specific parameters,

$$\begin{aligned}\frac{\partial L(u,w,i,j,m,k)}{\partial U_u} &= \frac{\partial L(u,w,i,j,m,k)}{\partial r_{uiuj;umNk}} \\ &\times \left(\lambda^d V_i + (1-\lambda^d) V_m - \lambda^d V_j - (1-\lambda^d)\lambda_1^d V_k\right) + \beta^d U_u\end{aligned} \tag{4.29}$$

$$\frac{\partial L(u,w,i,j,m,k)}{\partial U_w} = \frac{\partial L(u,w,i,j,m,k)}{\partial r_{uiuj;umNk}} \times (-(1-\lambda^d)(1-\lambda_1^d)V_k) + \beta^d U_w \tag{4.30}$$

and the gradients of the item-specific parameters,

$$\frac{\partial L(u,w,i,j,m,k)}{\partial V_i} = \frac{\partial L(u,w,i,j,m,k)}{\partial r_{uiuj;umNk}} \times (\lambda^d U_u) + \beta^d V_i \tag{4.31}$$

$$\frac{\partial L(u,w,i,j,m,k)}{\partial V_j} = \frac{\partial L(u,w,i,j,m,k)}{\partial r_{uiuj;umNk}} \times (-\lambda^d U_u) + \beta^d V_j \tag{4.32}$$

$$\frac{\partial L(u,w,i,j,m,k)}{\partial V_m} = \frac{\partial L(u,w,i,j,m,k)}{\partial r_{uiuj;umNk}} \times ((1-\lambda^d) U_u) + \beta^d V_m \tag{4.33}$$

$$\begin{aligned}\frac{\partial L(u,w,i,j,m,k)}{\partial V_k} &= \frac{\partial L(u,w,i,j,m,k)}{\partial r_{uiuj;umNk}} \\ &\times (-(1-\lambda^d)\lambda_1^d U_u - (1-\lambda^d)(1-\lambda_1^d) U_w) + \beta^d V_k\end{aligned} \tag{4.34}$$

where $\frac{\partial L(u,w,i,j,m,k)}{\partial r_{uiuj;umNk}} = -\frac{1}{1+\exp\,(r_{uiuj;umNk})}$.

4.3.3 The DBPL algorithm

The steps of DBPL are summarized in Algorithm 4 in detail. We divide DBPL into two steps for notational simplicity: user and item sampling (lines 4-7) and parameter update (lines 8-14).

4.3.4 Computational complexity

In the DBPL-based recommendation method, we also use the user-based and item-based collaborative filtering (*UIIU*) method to calculate the potential preference scores of each user for further distinguishing the preference differences between any two unobserved items. The computational complexity of computation for potential preference scores is $O(M^2 + N^2 + |G_u|M)$, where M is the number of users, N is the number of items, and $|G_u|$ is the number of each user's like-minded users, which is the same as that of PBPR.

We need to update the value of 6*d* parameters $\{U_u,\ U_w,\ V_i,\ V_j,\ V_m,\ V_k\}$. Therefore, the computational complexity of the learning parameters in the DBPL is $O(MTd)$, where M is the number of users, *T* is the number of iterations, and *d* is the number of latent dimensions. In total, the time complexity of the recommended method is $O(M^2 + N^2 + |G_u|M + MTd)$.

Algorithm 4 Learning parameters for the DBPL model

Input:

User-item interaction matrix R.

Potential preference matrix S.

Parameters η, λ^d, λ_1^d, and β^d.

Output:

The learned model parameters $\{U_u, V_i, u \in U^{tr},\ i \in I^{tr}\}$.

1. Randomly initialize U and V.
2. **For** $t_1 = 1, \dots, T$
3.   **For** $t_2 = 1, \dots, M$
4.   Randomly pick a user $u \in U^{tr}$.
5.   Randomly pick two items $i, m \in I_u^+$.
6.   Randomly pick two items $j, k \in I^{tr} \backslash I_u^+$, and $S_{uj} > S_{uk}$.
7.   Randomly pick a user $w \in U_k^{tr}$.
8.   Calculate r_{N_k} and $\frac{\partial L(u,w,i,j,m,k)}{\partial r_{uiuj;umNk}}$.
9.   Update U_u via Eq. (4.29).
10.   Update U_w via Eq. (4.30).
11.   Update V_i via Eq. (4.31).
12.   Update V_j via Eq. (4.32).
13.   Update V_m via Eq. (4.33).
14.   Update V_k via Eq. (4.34).
15.   **End**
16. **End**

4.4 Summary

In this section, we studied three recommendation models to solve the one-class collaborative filtering problem for more accurate recommendation results, including PBPR. PBPR* and DBPL. Three recommendation models are all extensions of BPR and need users' potential preference scores on items. We summarize the calculation methods which are used to provide potential preference scores for them in Table 4.4.

Table 4.4 Calculation methods of users'potential preference scores for different recommendation models

Recommendation models	The *UIIU*-based Method	The LDA-based Method
PBPR	✓	
PBPR*		✓
DBPL	✓	

PBPR relaxes the assumption in BPR and assumes each user's preference differences between any two unobserved items. We distinguished the preference differences by users' potential preference scores on items. Then, we proposed PBPR* to enhance the performance of PBPR by several strategies. We will conduct experiments to evaluate the effectiveness of these strategies. DBPL also considers users' potential preference scores to distinguish the preference differences between users' unobserved items. In addition, we assumed that the preference difference between the observed item and the unobserved item could be reduced by fusing a relatively smaller preference difference between another pair of items for further alleviating the vanishing gradient problem in the learning process of DBPL.

Our proposed recommendation models have the following advantages:

(1) The proposed recommendation models can be realized based on only sparse interactions between users and items without any additional social information.

(2) The proposed recommendation models only need relative potential preference differences between each user's unobserved items rather than the specific preference values. The relative potential preference differences have been measured by potential preference scores calculated by the methods in Section 3. It can also be realized by other methods like item popularity, i.e., for any two unobserved items of the user, he/she may prefer a more popular item than another one.

(3) We fused the preference score from another user on the specific item in all recommendation models. It can alleviate the error given by the potential preference scores and relax the strict relative potential preferences between users' unobserved items.

5 Experiments and Analysis

In this section, we demonstrate the effectiveness of the proposed recommendation models by a series of experiments on three real-world datasets.

5.1 Dataset

Three real-world datasets are employed as experimental data, including Last-FM[1], MovieLens 100K[2] and MovieLens 1M.

➢ **Last-FM**

Last-FM is the music listening dataset collected from Last. FM online music systems. It has rich tags. Last.FM tags have been widely used in many studies [82, 83], including the research on recommender systems [84, 85]. The dataset has 92,834 user-item listening records given by 1,892 users on 17,632 items.

➢ **MovieLens**

MovieLens is a system for movie recommendation. It was developed by the GroupLens research group [86]. We use MovieLens 100K and MovieLens 1M for experiments. MovieLens 100K dataset is collected from the MovieLens website. It contains 100,000 ratings between 943 users and 1,682 items. MovieLens 1M dataset, which is also collected from the MovieLens website, contains 951,612 ratings assigned by 6040 users over 3,952 items.

For studying the one-class feedback problem on datasets MovieLens 100K and MovieLens 1M, we do not preprocess datasets according to their rating values scale 1-5, such as keeping the ratings larger than 3 as the observed feedback. We consider all observed user-item pairs as positive feedbacks in all experiments [55]. The description of the experimental datasets is presented in Table 5.1.

For all three datasets, we randomly sample 20% and 50% of user-item interactions on each dataset as training data, respectively, and the rest as test data.

[1] https://grouplens.org/datasets/hetrec-2011/.
[2] https://grouplens.org/datasets/movielens/.

Table 5.1 Description of the experimental datasets

Dataset	#Users	#Items	#Interactions	Sparsity
Last-FM	1,892	17,632	92,834	99.72%
MovieLens 100K	943	1,682	100,000	93.70%
MovieLens 1M	6,040	3,952	951,612	96.01%

5.2 Experimental Design

We will conduct the following experiments to evaluate the performance of our proposed recommendation models.

➢ **Performance of recommendation models**

To evaluate the performance of recommendation models, we compare our proposed models with benchmark methods, including PopRank, *UIIU*, LDA, BPR, GBPR and CPLR, by convincing evaluation metrics.

➢ **Impacts of parameters in PBPR**

In PBPR, the trade-off parameter λ_1^p is used to adjust the preference scores from two users. We will investigate the performance of PBPR with different parameter settings on three real-world datasets.

➢ **Impacts of components in PBPR***

Different from PBPR, PBPR* adopts the LDA-based method to calculate potential preference scores. In addition, there are two main components in PBPR*. We will test and verify the effects of two components in PBPR* on the recommendation results, including the parameter N_A and the confidence coefficients.

➢ **Impacts of parameters in DBPL**

In DBPL, we use the trade-off parameters to a) adjust the preference scores from two users and b) consider the potential preference scores into the prediction function. We will conduct experiments to study the effects of different parameter settings on the recommendation results.

➢ **Discussion**

We will describe more details of the sampling process in PBPR and DBPL and then discuss the explainability in the existing recommendation methods and the explainability of potential preference scores in this thesis.

5.3 Performance of Recommendation Models

5.3.1 Benchmark methods

We compare our proposed recommendation models PBPR, PBPR* and DBPL with several recommendation methods, including

- PopRank [54]: PopRank recommends each user a ranking list of items according to the item's popularity in the training data. It is not a personalized recommendation approach usually used for solving the user cold-start problem.
- *UIIU*: *UIIU* represents the user-based and item-based recommendation method, which we propose in Section 3.2. It can also be used to predict users' preferences via aggregation of the item-item similarity and the like-minded users' preferences.
- LDA: LDA represents the LDA-based recommendation method, which we propose in Section 3.3. It can also be used to predict users' preferences via the inner product of two distributions.
- BPR: As introduced in Section 2.3, the Bayesian Personalized Ranking method is a state-of-the-art pairwise learning method based on matrix factorization. It recommends a personalized ranking list of items for each user only based on the user's historical interactions.
- GBPR: Group preference-based Bayesian Personalized Ranking method is an extension of the BPR model, which relaxes individual and independence assumptions in the BPR model. This work is able to accommodate richer interactions among users.
- CPLR: Collaborative Pairwise Learning to Rank is a generalized BPR. It uses the influence between users on the preferences for observed items and unobserved items to relax the assumption of BPR.

5.3.2 Experimental setting

We set the iteration number T=10,000. To make the comparison fair, we initialize all pairwise ranking models (BPR, GBPR, CPLR, PBPR, PBPR* and DBPL) with the same random distribution. We set the same dimensionality and the same hyper-parameter of the regularization terms for all recommendation models. The dimensionality is empirically set to $k = 20$, and the hyper-parameter is set to $\beta^p = \beta^* = \beta^d = 0.01$. The learning rate η is chosen from {0.02, 0.05, 0.1}.

For the GBPR model, we fixed the user group size as $|G| = 3$, the parameters ρ=1 and ρ=0.6 are set for the datasets MovieLens 100K and MovieLens 1M, respectively, considering the settings in [56]. The parameter ρ=0.8 is set for the dataset Last-FM. For the CPLR model, the size of the neighborhood is set to 50, and the control coefficients are set to $\alpha = \beta = \gamma = 0.1$.

For the PBPR model, the control coefficient is set as $\lambda^p = 0.7$ by default, the control coefficient for the dataset Last-FM is set as $\lambda_1^p = 0.8$. The trade-off parameter λ_1^p is set as 0.9 for the datasets MovieLens 100K and MovieLens 1M, and λ_1^p is set as 0.3 for Last-FM. For the PBPR* model, the control coefficient $\lambda^* = 0.7$ for the datasets MovieLens 100K and MovieLens 1M, and the control coefficient $\lambda^* = 0.8$ for the dataset Last-FM. The trade-off parameter λ_1^* is fixed as 0.9 for the datasets MovieLens 100K and MovieLens 1M, and 0.3 for the dataset Last-FM. The parameter N_A is set as 1, 0, 4 for the datasets Last-FM (20%), MovieLens 100K (20%) and MovieLens 1M (20%), and N_A is set as 3, 1, 2 for the datasets Last-FM (50%), MovieLens 100K (50%) and MovieLens 1M (50%), respectively.

For the DBPL model, we set the control coefficient λ^d= 0.7 and the trade-off parameter λ_1^d= 0.2 on all datasets. We add potential preference scores to the prediction function of DBPL and denote the recommendation results with the novel prediction function as DBPL+. For the DBPL+ model, the user *u*'s preference score on item *i* is defined as $r'_{ui} = r_{ui} + \alpha^d \cdot \sqrt{S_{ui}}$, where α^d is a trade-off parameter for the potential preference scores. In experiments, we set α^d = 0.8 for the datasets Last-FM, MovieLens 100K and MovieLens 1M.

5.3.3 Evaluation metrics

We report the average performance of users in the test data. Users pay attention to a few top-ranked items [87], so we use top-*N* evaluation metrics [12, 88], including precision at rank *N* (Prec@*N*) and normalized discounted cumulative gain at rank *N* (NDCG@*N*), to evaluate the performance of our methods.

The metric Prec@*N* [89] for user *u* is defined as follows:

$$Prec@N = \frac{1}{N}\sum_{i=1}^{N}\delta(I_i \in I_u^+) \tag{5.1}$$

where I_i represents the i-th item in the ranking list, $\delta(\cdot)$ represents the indicator function. Its value is equal to 1 when $I_i \in I_u^+$, otherwise 0.

The normalized discounted cumulative gain (NDCG) [90], which takes into account the position of correctly recommended items, is a standard measure of ranking quality. It is defined as

$$NDCG@N = \frac{1}{Z_N}\sum_{j}^{N}\frac{2^{r(j)}-1}{\log(j+1)} \tag{5.2}$$

where j represents the j-th position in the ranking list, and $r(j)$ presents the relevance of the item in position j. Z_N represents the ideal value for the discounted cumulative gain (DCG, $\mathrm{DCG} = \sum_j^N \frac{2^{r(j)}-1}{\log(j+1)}$). Here, we set N=5 for the metric Prec@N, and we set N=5, N=10 for the metric NDCG@N.

5.3.4 Results and analysis

To evaluate the performance of recommendation models, we compared our proposed models with benchmark methods by convincing evaluation metrics.

Our experiments are performed on a Windows 10 with 3.6GHz Intel Core i3, 8 GB. The results of all recommendation methods on the datasets Last-FM, MovieLens 100K and MovieLens 1M are presented in Table 5.2, Table 5.3, Table 5.4, respectively. The numbers in boldface are the best results among all methods. From the three tables, we make the following observations:

(1) From Table 5.2, Table 5.3 and Table 5.4, it is obvious that our proposed recommendation models significantly improve BPR in terms of Prec@5, NDCG@5 and NDCG@10 on all datasets, which demonstrates the effectiveness of our assumptions for enhancing the performance of models. DBPL^{+} performs the best in most cases according to evaluation metrics in Section 5.3.3. The experimental results of PBPR^{*} outperform PBPR on all datasets, which shows the effectiveness of strategies proposed to enhance the performance of PBPR in Section 4.2.1. The results of DBPL^{+} outperform DBPL in most datasets, demonstrating the effect of considering potential preference scores into the prediction functions.

Table 5.2 Recommendation performance of PopRank, *UIIU*, LDA, BPR, GBPR, CPLR, PBPR, DBPL and DBPL+ on the dataset Last-FM in terms of Prec@5, NDCG@5 and NDCG@10

Dataset	Method	Prec@5	NDCG@5	NDCG@10
Last-FM (20%)	PopRank	0.1588	0.1314	0.1441
	UIIU	0.1751	0.1794	0.1637
	LDA	0.3453	0.3380	0.3184
	BPR	0.2874	0.2947	0.2819
	GBPR	0.3131	0.3200	0.3027
	CPLR	0.3169	0.3070	0.2929
	PBPR	0.3226	0.3327	0.3053
	PBPR*	0.3483	0.3554	0.3296
	DBPL	0.3572	0.3605	0.3355
	DBPL+	**0.3687**	**0.3701**	**0.3468**
Last-FM (50%)	PopRank	0.1434	0.1538	0.1501
	UIIU	0.2740	0.2870	0.2596
	LDA	0.3573	0.3725	0.3239
	BPR	0.3346	0.3500	0.3062
	GBPR	0.3425	0.3543	0.3135
	CPLR	0.3907	0.4156	0.3581
	PBPR	0.3653	0.3852	0.3362
	PBPR*	0.3875	0.4034	0.3553
	DBPL	0.3906	0.4152	0.3547
	DBPL+	**0.4014**	**0.4270**	**0.3623**

(2) Note that the sparsity of the user-item interaction matrix has an influence on the recommendation results. Experimental results on the dataset Last-FM, which has the highest sparsity among all datasets, are the smallest compared to the results on other datasets. From the three tables, it is significant that the results of the *UIIU* method have a larger gap with other methods on the Last-FM than other datasets, implying the performance of the *UIIU* method also suffers from the data sparsity problem. It is obvious in Table 5.3 and Table 5.4 that the overall results of the dataset MovieLens 100K are slightly better than MovieLens 1M.

Table 5.3 Recommendation performance of PopRank, *UIIU*, LDA, BPR, GBPR, CPLR, PBPR, DBPL and DBPL+ on the dataset MovieLens 100K in terms of Prec@5, NDCG@5 and NDCG@10

Dataset	Method	Prec@5	NDCG@5	NDCG@10
MovieLens 100K (20%)	PopRank	0.4838	0.4960	0.4702
	UIIU	0.4592	0.4729	0.4402
	LDA	0.5519	0.5677	0.5295
	BPR	0.4880	0.4975	0.4769
	GBPR	0.5355	0.5443	0.5223
	CPLR	0.5843	0.5977	0.5719
	PBPR	0.5642	0.5758	0.5478
	PBPR*	0.5720	0.5816	0.5567
	DBPL	0.5840	0.5840	0.5568
	DBPL+	**0.6059**	**0.6160**	**0.5860**
MovieLens 100K (50%)	PopRank	0.3862	0.4009	0.3713
	UIIU	0.5007	0.5201	0.4771
	LDA	0.5648	0.5820	0.5394
	BPR	0.5461	0.5613	0.5266
	GBPR	0.5828	0.6006	0.5553
	CPLR	0.5949	0.6123	0.5668
	PBPR	0.5922	0.6098	0.5626
	PBPR*	0.5934	0.6116	0.5678
	DBPL	0.6066	0.6202	**0.5716**
	DBPL+	**0.6070**	**0.6210**	0.5707

(3) Compared with the training data with 20% user-item pairs, the results of the training data with 50% user-item pairs increase in terms of Prec@5, NDCG@5 and NDCG@10 because more user-item pairs could provide more information about users' potential preferences. It is obvious recommendation results of PBPR* are better than that of PBPR on all evaluation metrics on three datasets, which shows the effectiveness of conducting strategies for PBPR.

Table 5.4 Recommendation performance of PopRank, *UIIU*, LDA, BPR, GBPR, CPLR, PBPR, DBPL and DBPL+ on the dataset MovieLens 1M in terms of Prec@5, NDCG@5 and NDCG@10

Dataset	Method	Prec@5	NDCG@5	NDCG@10
MovieLens 1M (20%)	PopRank	0.3611	0.3362	0.3528
	UIIU	0.3917	0.3861	0.3739
	LDA	0.5281	0.5196	0.5103
	BPR	0.5145	0.5213	0.4999
	GBPR	0.5497	0.5552	0.5342
	CPLR	0.5504	0.5502	0.5343
	PBPR	0.5550	0.5575	0.5407
	PBPR*	0.5680	0.5733	0.5525
	DBPL	0.5658	0.5713	0.5504
	DBPL+	**0.5731**	**0.5769**	**0.5591**
MovieLens 1M (50%)	PopRank	0.3301	0.3454	0.3293
	UIIU	0.4294	0.4379	0.4097
	LDA	0.5214	0.5317	0.5017
	BPR	0.5092	0.5177	0.4912
	GBPR	0.5307	0.5424	0.5110
	CPLR	0.5382	0.5567	0.5153
	PBPR	0.5527	0.5653	0.5292
	PBPR*	0.5566	0.5693	0.5322
	DBPL	0.5588	0.5729	0.5361
	DBPL+	**0.5619**	**0.5769**	**0.5361**

(4) PopRank performs comparably poorly than other benchmark methods because it does not conduct personalized recommendations. It is worthwhile to note that LDA achieves better performance than *UIIU* and BPR, especially on the dataset Last-FM. This further shows the benefit of performing each user's prediction based on only his/her own historical items. The LDA-based method does not need to calculate the information such as the like-minded users, which may suffer from the sparse data. GBPR and CPLR are all extensions of BPR. They achieve better performance than BPR across all evaluation metrics, indicating the effectiveness of novel assumptions adopted by two models.

(5) The gaps between the *UIIU* method and recommendation models are reduced with the increase of the training data. It is obvious that the *UIIU* method could provide well results when the training user-item pairs are enough. However, data with high density is difficult to obtain in real-life applications. The results of the *UIIU* method could be integrated with the recommendation model to enhance its performance and provide more accurate recommendations.

5.4 Impacts of Parameters in PBPR

5.4.1 Experimental setting

In PBPR, λ_1^p is an important parameter used to fuse preferences from two users. We discuss the effects of different parameter settings on the performance of the PBPR-based recommendation method.

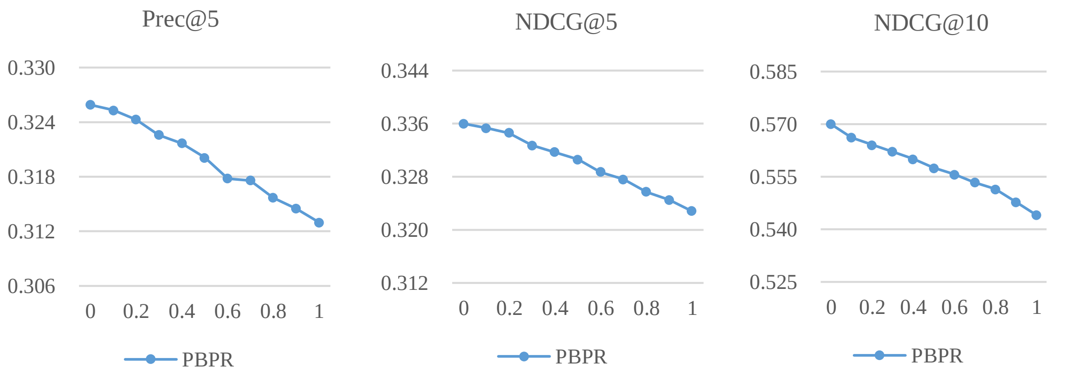
The parameter λ_1^p is set to $\lambda_1^p = \{0, 0.1, \dots, 1\}$. The dimensionality is empirically set to $k = 20$, and the hyper-parameter is set to $\beta^p = 0.01$. The learning rate η is chosen from $\{0.05,\ 0.1\}$. The control coefficient is set as $\lambda^p = 0.7$ for the datasets MovieLens 100K and MovieLens 1M, and the control coefficient for the dataset Last-FM is set as $\lambda_1^p = 0.8$.

5.4.2 Results and analysis

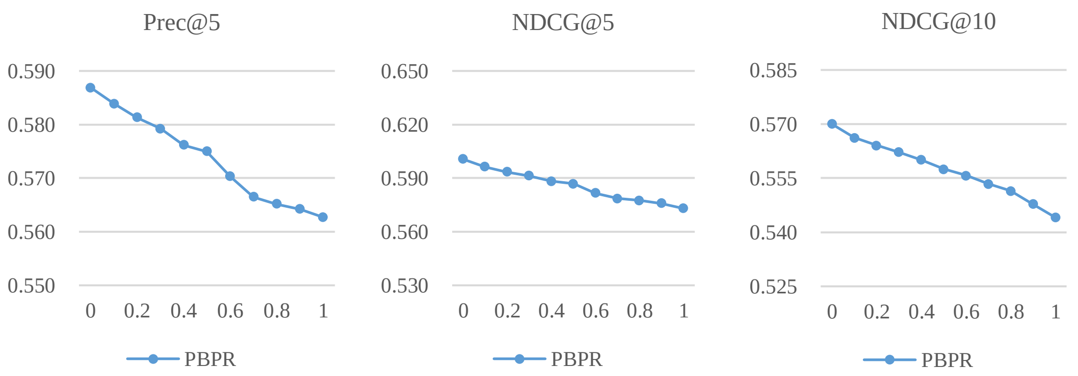
Figure 5.1 shows the experimental results of PBPR with different trade-off parameter values on the datasets Last-FM (20%), MovieLens 100K (20%) and MovieLens 1M (20%), from which we could see that:

(1) The experimental results of the datasets Last-FM (20%) and MovieLens 100K (20%) have a downward trend with the increasing of the value for the trade-off parameter λ_1^p, i.e., considering another user's preference has an influence on the results. PBPR performs best at $\lambda_1^p = 0$ on two datasets. Note that when $\lambda_1^p = 0$, the preference from another user does not take into account.

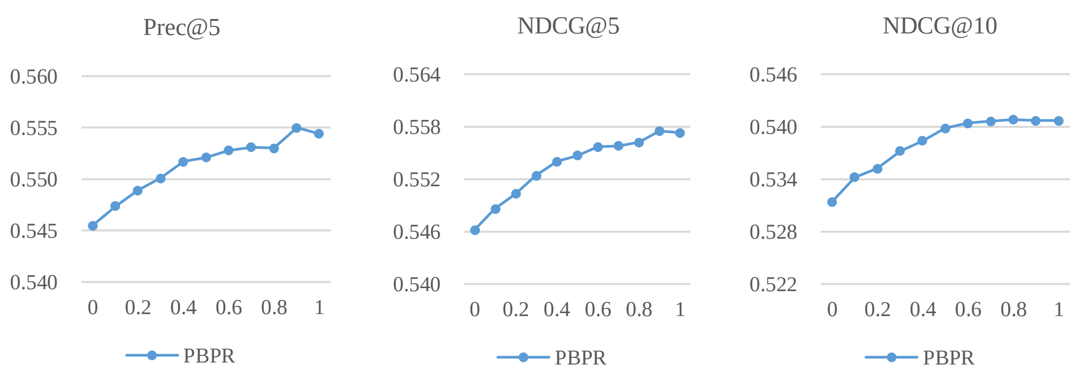
(2) Contrary to the above, the results on the dataset MovieLens 1M (20%) increase when the parameter λ_1^p ranges from 0 to 0.9. From Figure 5.1 c), fusing the preference from another user improves the performance of PBPR.



a) Last-FM (20%)



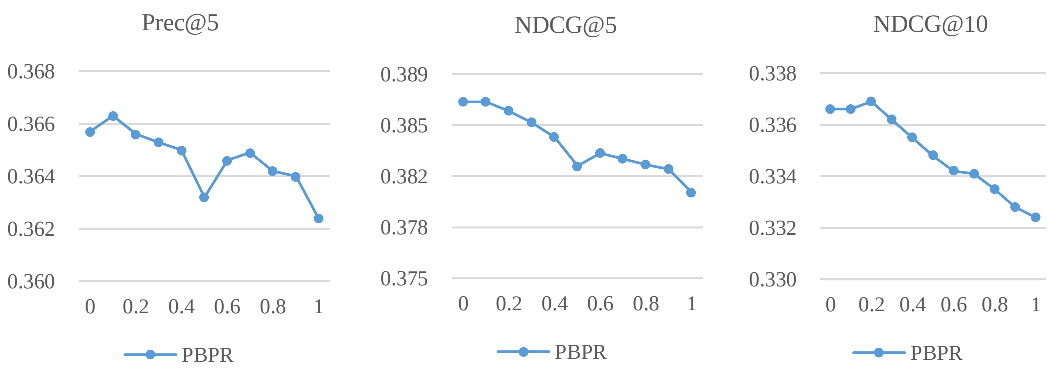
b) MovieLens 100K (20%)



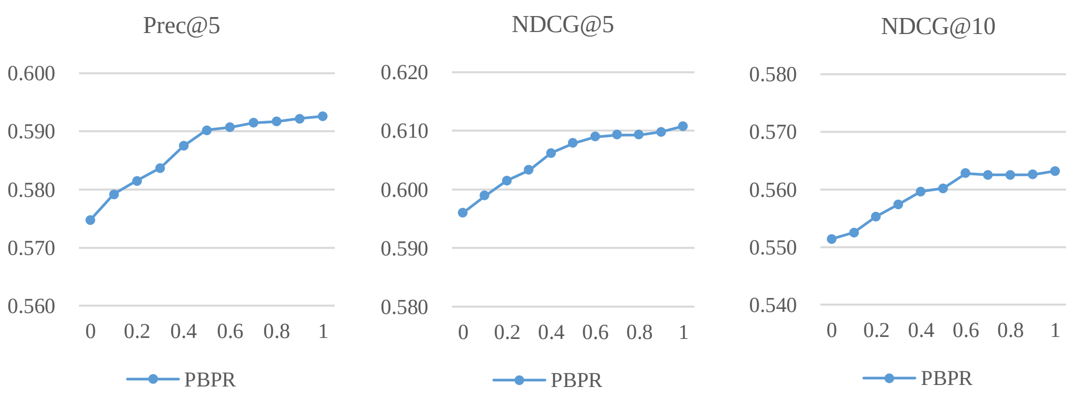
c) MovieLens 1M (20%)

Figure 5.1 The influence of the trade-off parameter on the performance of PBPR on Last-FM (20%), MovieLens 100K (20%) and MovieLens 1M (20%) in terms of Prec@5, NDCG@5 and NDCG@10

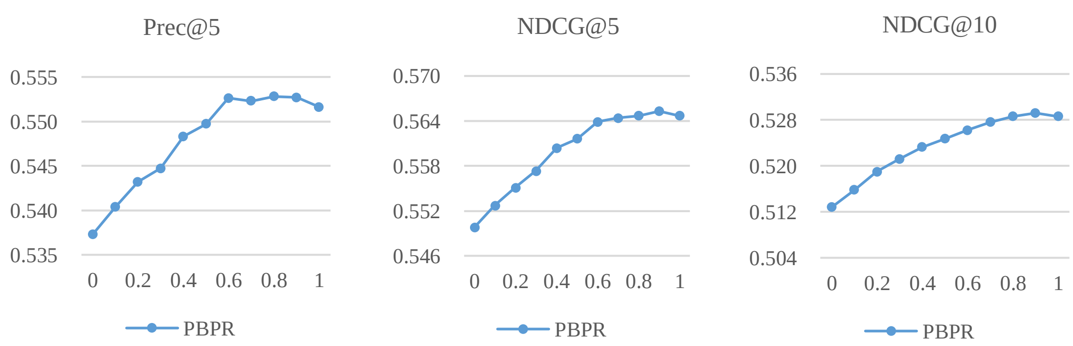
We also conduct experiments for the trade-off parameter λ_1^p on three datasets Last-FM (50%), MovieLens 100K (50%), and MovieLens 1M (50%)), for further studying its influence on the recommendation results.



a) Last-FM (50%)



b) MovieLens 100K (50%)



c) MovieLens 1M (50%)

Figure 5.2 The influence of the trade-off parameter on the performance of PBPR on Last-FM (50%), MovieLens 100K (50%) and MovieLens 1M (50%) in terms of Prec@5, NDCG@5 and NDCG@10

The experimental results are illustrated in Figure 5.2, from which we make the following observations:

(1) It is significant that the results of the dataset Last-FM (50%) have a downward trend with the increasing of the value for the trade-off parameter λ_1^p, and the

Table 5.5 Description of recommendation models on different components

	Component one	Component two
PBPR*-1	×	×
PBPR*-2	✓	×
PBPR*-3	×	✓
PBPR*	✓	✓

results on the datasets MovieLens 100K (50%) and MovieLens 1M (50%) have a growing trend with the increasing of the parameter λ_1^p.

(2) Different from the resulting trend on the dataset MovieLens 100K (20%), the experimental results on the dataset MovieLens 100K (50%) increase when the trade-off parameter ranges from 0 to 1. The reason for this difference might be the performance of the potential preference scores.

5.5 Impacts of Components in PBPR*

5.5.1 Experimental setting

We adopt several strategies to enhance the performance of PBPR for obtaining better recommendation results. From Table 5.2, Table 5.3 and Table 5.4, we can see that the PBPR*-based method outperforms the PBPR-based method on all evaluation metrics, which shows the effectiveness of conducting strategies in Section 4.2. The novel model is named PBPR*. Compared with PBPR, PBPR* mainly has two novel components, including

Component one: for user u's two unobserved items, at least an unobserved item of user u has been observed by at least N_A users;

Component two: the confidence of pairwise preferences is adjusted by confidence coefficients.

To have a deep understanding of the effect of these components, we conduct experiments to test and verify the performance of different PBPR*-based methods (PBPR*-1, PBPR*-2, PBPR*-3 and PBPR*), which are listed in Table 5.5, on all datasets. The dimensionality is also set to $k = 20$, and the hyper-parameter is set to $\beta^* = 0.01$. The learning rate η is chosen from {0.02, 0.05, 0.1}. The control

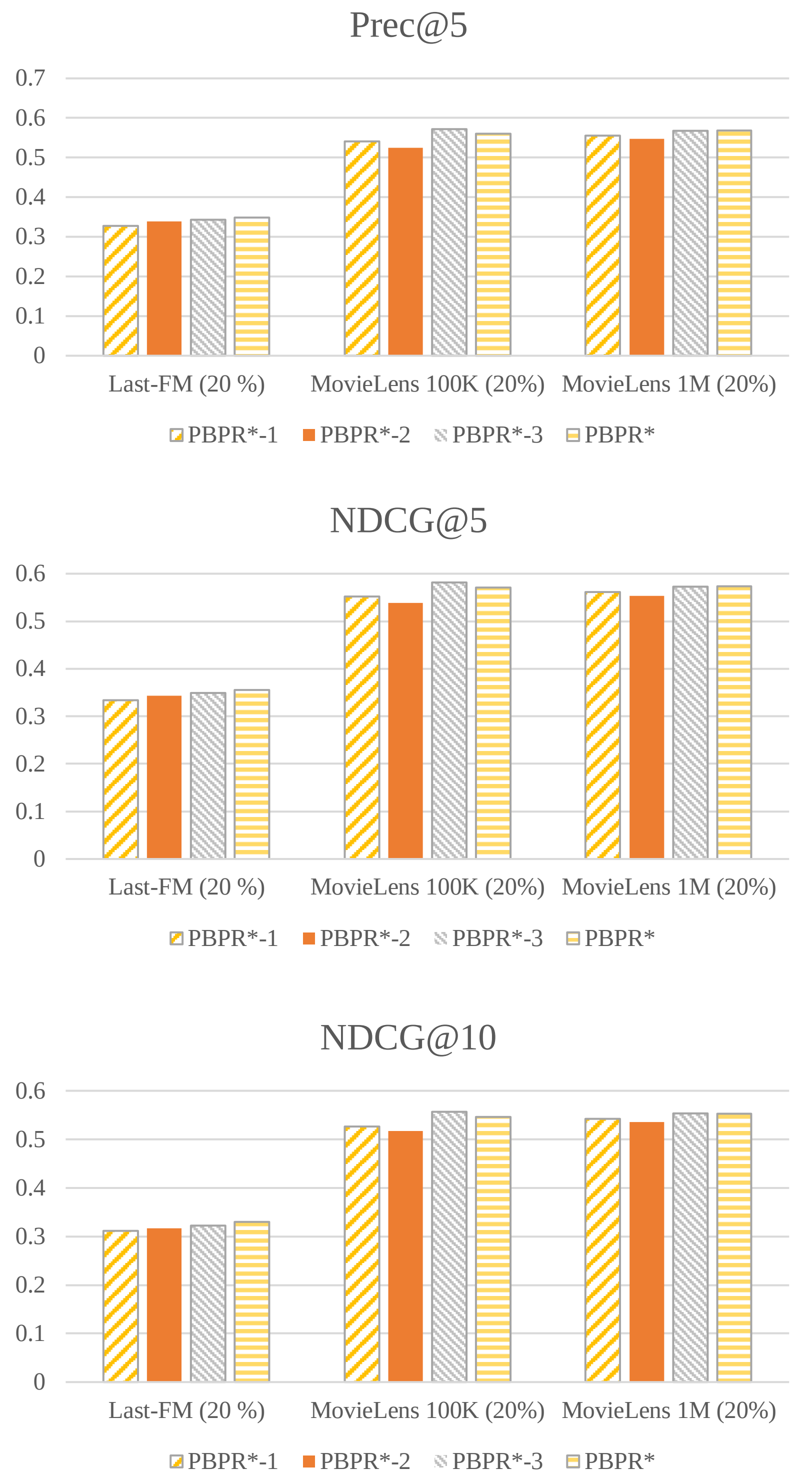


Figure 5.3 The influence of different components on the performance of PBPR* in terms of Prec@5, NDCG@5 and NDCG@10 on Last-FM (20%), MovieLens 100K (20%) and MovieLens 1M (20%)

coefficient is set as $\lambda^* = 0.7$ for the datasets MovieLens 100K and MovieLens 1M, and the control coefficient for the dataset Last-FM is set as $\lambda_1^* = 0.8$.

For the component one, the parameter N_A is set as 1, 1, 4 for the datasets Last-FM (20%), MovieLens 100K (20%) and MovieLens 1M (20%), and N_A is set as 3, 1, 2 for the datasets Last-FM (50%), MovieLens 100K (50%) and MovieLens 1M (50%), respectively.

5.5.2 Results and analysis

The experimental results of different PBPR*-based recommendation methods on all real-world datasets are shown in Figure 5.3 and Figure 5.4. From Figure 5.3 and Figure 5.4, we can see that the experimental results of PBPR* are better than PBPR*-1 in terms of Prec@5, NDCG@5 and NDCG@10 on all datasets. That verifies the effectiveness of components in PBPR*. We use PBPR*-2 and PBPR*-3 to test and verify the effects of the two components, respectively.

To widen the preference gap of each pair of unobserved items in the sampling process, we propose a strategy (component one) in Section 4.2 that at least an unobserved item of user u has been observed by at least N_A users. This strategy is easy to implement but requires to be adjusted on each dataset. Comparing the experimental results of PBPR*-2 with that of PBPR*-1, we can observe that component one does not improve the results on the datasets MovieLens 100K (20%) and MovieLens 1M (20%). However, it could improve the recommendation results of the model on the dataset Last-FM, especially on the dataset Last-FM (50%). We will study the more complicated strategy for good performance on different datasets.

From Figure 5.3 and Figure 5.4, it is obvious that component two improves the results on all datasets, especially on the datasets MovieLens 100K (20%) and MovieLens 1M (20%). That demonstrates the effectiveness of considering confidence coefficients, which are calculated based on the potential preference scores, for further distinguishing each user's preference differences.

Besides, by jointly comparing tables (Table 5.2, Table 5.3 and Table 5.4) and figures (Figure 5.3 and Figure 5.4), we observe that the results of PBPR*-1 are not better than PBPR in some cases. Actually, the potential preference scores calculated by the LDA-based method significantly outperform those calculated by the *UIIU* method. It is clear that a novel integration method is needed to combine the potential preference scores with recommendation models.

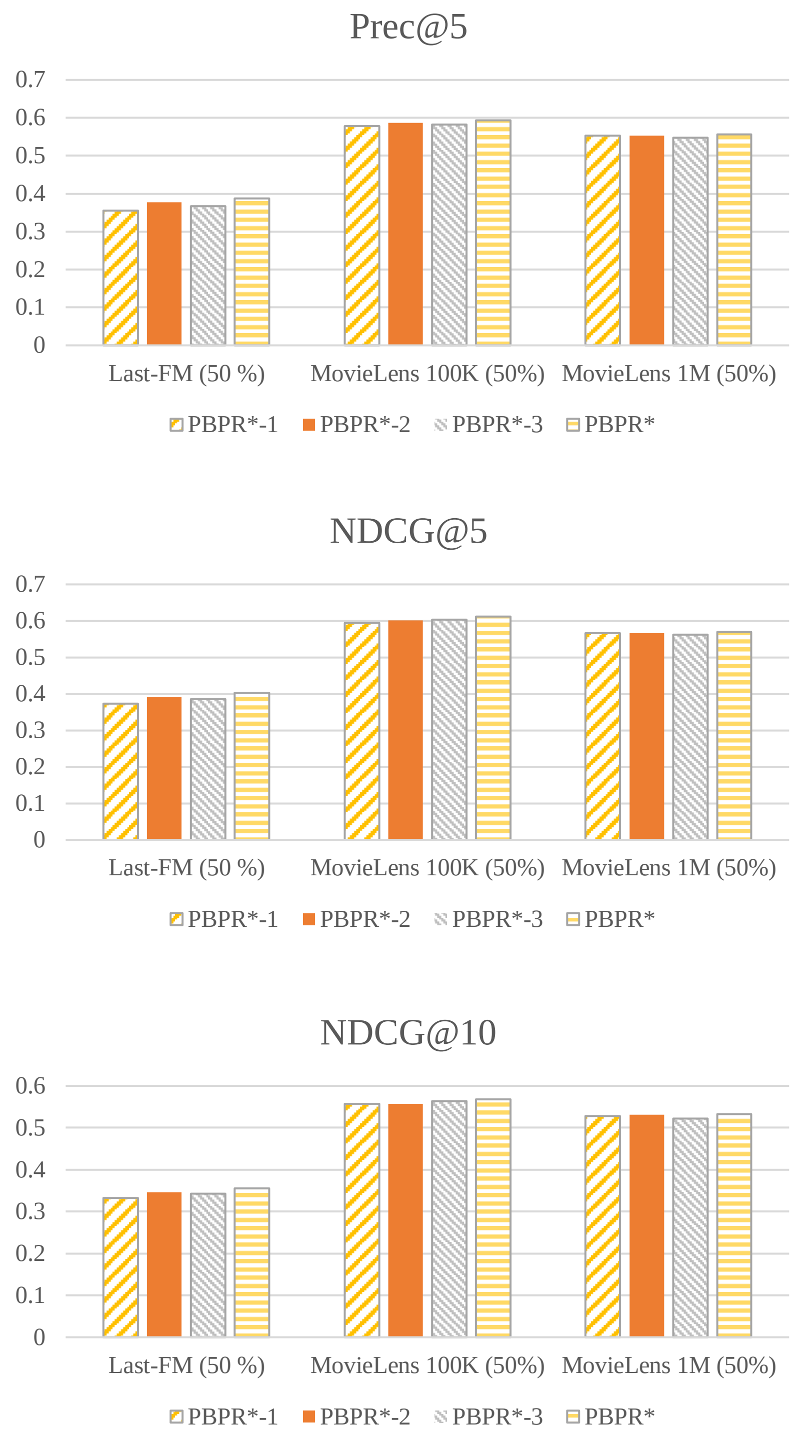


Figure 5.4 The influence of different components on the performance of PBPR* in terms of Prec@5, NDCG@5 and NDCG@10 on Last-FM (50%), MovieLens 100K (50%) and MovieLens 1M (50%)

5.6 Impacts of Parameters in DBPL

5.6.1 Experimental setting

As mentioned before, λ_1^d is an important parameter for DBPL and DBPL+, and α is the trade-off parameter used for fusing potential preference scores into the prediction function. We discuss the effects of different parameter settings on the performance of the DBPL-based recommendation method.

To have a deep understanding of the effect of taking into account another user's preference for DBPL and DBPL+ in terms of Prec@5, NDCG@5 and NDCG@10 on the datasets Last-FM (20%), MovieLens 100K (20%) and MovieLens 1M (20%), we conduct experiments with the parameter is chosen from $\lambda_1^d \in \{0.1, 0.2, \ldots, 1\}$.

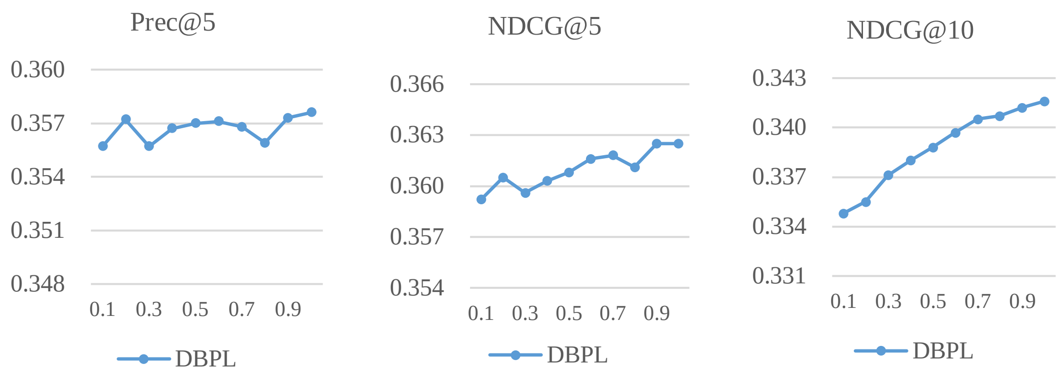
We study the influence of the trade-off parameter α^d on the performance of DBPL+ on the datasets Last-FM (50%), MovieLens 100K (50%) and MovieLens 1M (50%). The parameter α^d is set to $\alpha^d = \{0.1, 0.2, \ldots, 1\}$. We define the preference score of user u on item i as $r'_{ui} = r_{ui} + \alpha^d \cdot \sqrt{S_{ui}}$. In this equation, the difference between potential preferences could be reduced by a square root operation. We use another definition equation $r''_{ui} = r_{ui} + \alpha^d \cdot S_{ui}$, which is denoted as Comparison_DBPL+ (C_DBPL+), for comparing the effectiveness of r'_{ui} on recommendation results.

5.6.2 Results and analysis

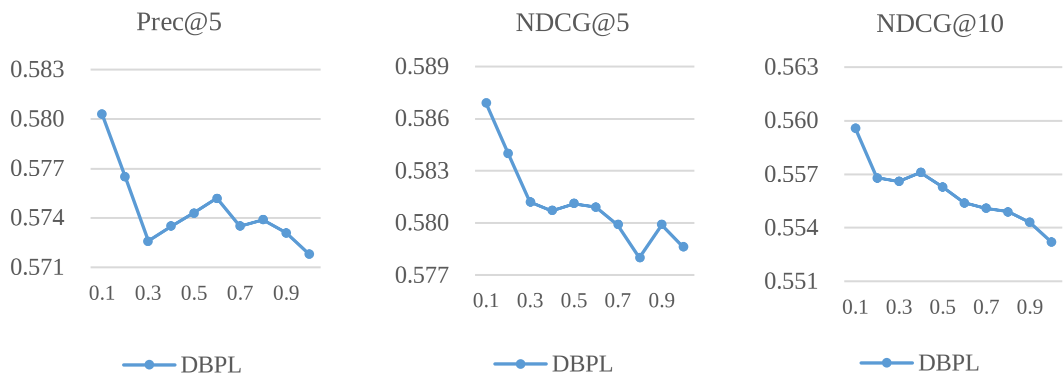
Experimental results of the DBPL-based recommendation method with different values of the parameter λ_1^d are shown in Figure 5.5, from which we could see that:

(1) Overall, the results of the datasets MovieLens 100K (20%) and MovieLens 1M (20%) have a downward trend with the increasing of the value for the parameter λ_1^d, i.e., considering less preference from another user affect results. It can be inferred from the experimental results in Figure 1 that preference differences of unobserved items could not be combined with the potential preference matrix well on the datasets MovieLens 100K (20%) and MovieLens 1M (20%).

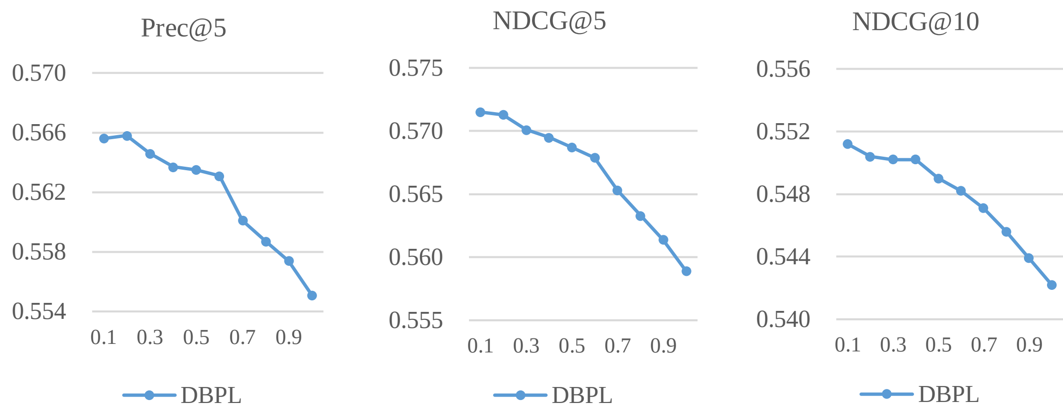
(2) Contrary to the above, the results on the dataset Last-FM (20%) increase when the parameter λ_1^d ranges from 0.1 to 1. DBPL performs best at $\lambda_1^d = 1$. Another user's preference is not taken into account when $\lambda_1^d = 1$.



a) Last-FM (20%)



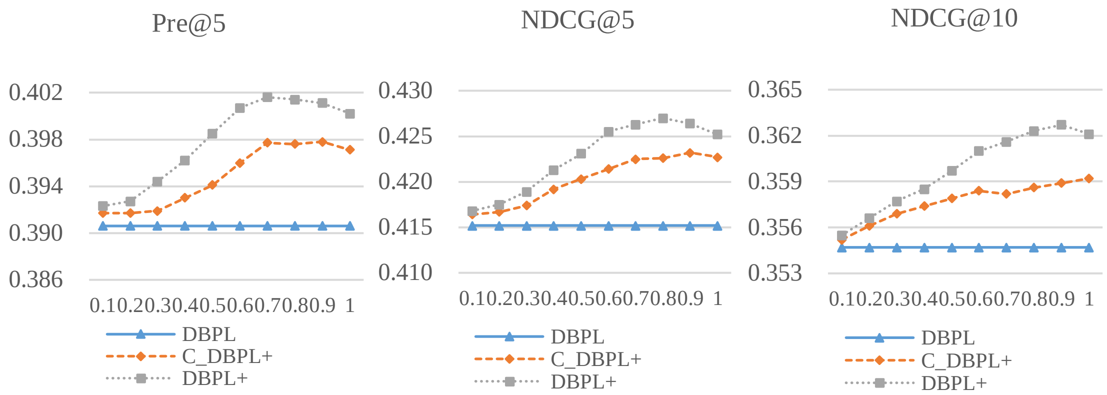
b) MovieLens 100K (20%)



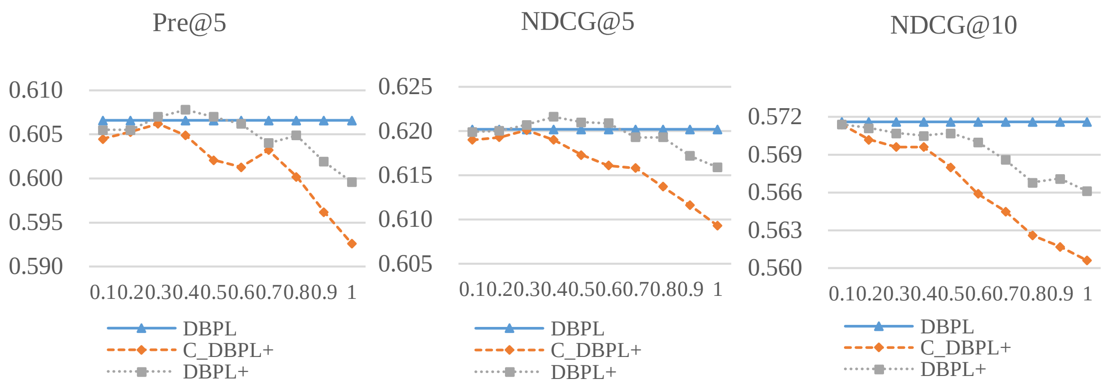
c) MovieLens 1M (20%)

Figure 5.5 The influence of the trade-off parameter on the performance of DBPL on Last-FM (20%), MovieLens 100K (20%) and MovieLens 1M (20%) in terms of Prec@5, NDCG@5 and NDCG@10

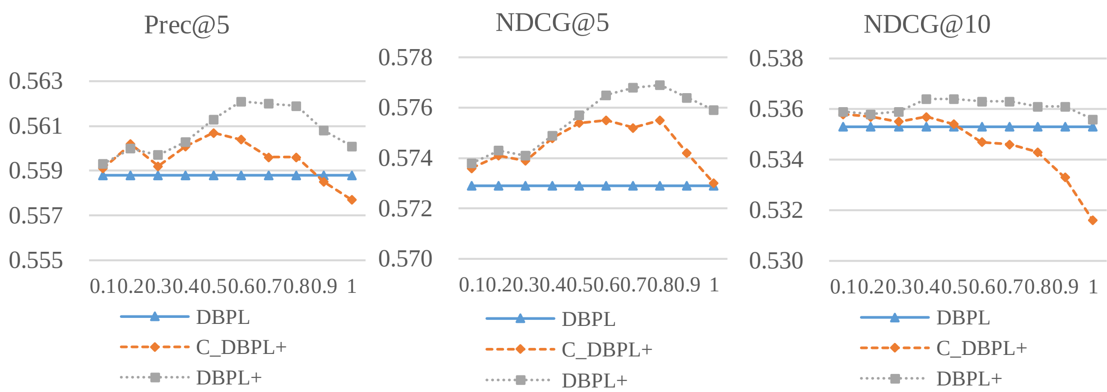
Figure 5.6 shows the results of DBPL, C_DBPL+ and DBPL+ with different settings of the parameter α^d on the datasets Last-FM (50%), MovieLens 100K (50%) and MovieLens 1M (50%).



a) Last-FM (50%)



b) MovieLens 100K (50%)



c) MovieLens 1M (50%)

Figure 5.6 The influence of the parameter α^d on the performance of C_DBPL+ and DBPL+ on Last-FM (50%), MovieLens 100K (50%) and MovieLens 1M (50%) in terms of Prec@5, NDCG@5 and NDCG@10

The results of DBPL+ are better than C_DBPL+ across all evaluation metrics on three datasets, demonstrating that considering the square root of potential preference scores into the prediction function can improve the recommendation results. As can be concluded from Figure 5.6, the potential preference scores behave differently on varied datasets. They can enhance results with some settings of the parameter α^d in most cases except the result of NDCG@10 on the dataset MovieLens 100K (50%). This is possibly caused by different performances of potential preference scores calculated based on user-item interactions. The *UIIU* method performs the best on the dataset MovieLens 100K (50%) compared with other datasets, so the current prediction function may not improve final results well. The novel prediction function should be studied in the future to achieve a balance between the performance of potential preference scores and recommendation results, making the model adaptive to different performances of potential preference scores.

5.7 Discussion

5.7.1 More details in the sampling process

We leverage the *UIIU* method to calculate potential preference scores between users and items for further distinguishing each user's preference difference between each pair of unobserved items. However, for a user, the potential preference scores of the user's unobserved items might be all equal to zero when lacking the information about like-minded users and similarities between items.

We empirically adopt the second strategy for picking two unobserved items in Algorithm 2 and Algorithm 4. We pick the unobserved item observed by more than two users as the unobserved item *j* and randomly pick another unobserved item as the unobserved item *k*. This sampling strategy has been used for solving the above sampling problem on the dataset Last-FM.

5.7.2 Explainability of potential preference scores

Explainability becomes critically important for recommender systems to provide convincing results [18]. Explainability helps to improve the transparency, persuasiveness, effectiveness, trustworthiness, and satisfaction of recommendation systems [91]. Explainable recommender systems [92] aim to reveal why a user might like the item, and it helps improve users' satisfaction or acceptance of recommendation results [93]. The explanations for recommender systems are usually based on phrase sentiment [94], aspect [95], and social networks [96].

In this thesis, we adopt two calculation methods to calculate users' potential preference scores on their unobserved items, which are (1) the user-based and item-based collaborative filtering method and (2) the LDA-based users' chosen intentions method. Both of them provide an explainable reason for why the user might have an interest in the items with higher potential preference scores. Therefore, in PBPR, PBPR* and DBPL, the potential preference scores provide an explainable reason for why the user *u* might prefer the unobserved item *j* over the unobserved item *k*.

5.8 Summary

In this section, we conducted a series of experiments to test and verify the performance of our proposed recommendation models.

We first compared our proposed recommendation models with benchmark methods on three real-world datasets. And then, we investigated the effects of a) the trade-off parameters in PBPR and DBPL; and b) the components in PBPR* on recommendation results. Finally, we discussed more details in the study.

Experimental results of our proposed recommendation models are better than the previous model on three real-world datasets, demonstrating the effectiveness of the assumptions in Section 4. We also evaluated the effects of different parameter settings on the performance of recommendation models. The improvement work could be considered in the future, such as studying the novel integration method for potential preference scores and the recommendation model.

6 Conclusion and Future Work

With the rapid spread of multimedia Web applications, a large number of contents are being generated online in real-time. Users will spend much effort in finding interesting things from the massive data. Therefore, recommender systems are proposed to deal with information overload and meet users' personalized interests. In many real-world applications, only user-item interactions (one-class feedback) can be observed. In this thesis, we focused on studying recommendation models for one-class collaborative filtering. We will summarize the whole thesis and then give our future work in this section.

6.1 Conclusion

The recommendation methods have been studied for personalized ranking with one-class feedback in recent years. Pairwise ranking methods have been widely used for dealing with the one-class problem with the assumption that users prefer their observed items over unobserved items. BPR is a well-performed pairwise ranking model. We studied novel recommendation models, which are extensions of BPR, for obtaining more accurate recommendations. The concrete summaries and contributions of this thesis are summarized as follows:

(1) We considered the users' preference differences on their unobserved items by distinguishing the fine-grained preference difference between any two unobserved items of the user, rather than dividing the users' unobserved items into different groups and studying the preference differences between unobserved items from different groups. Therefore, we studied two calculation methods to calculate users' potential preference scores on their unobserved items. The potential preference scores, which is calculated only with users' historical interaction, could be used to (a) measure the relative preference of users on their unobserved items; (b) relax the data sparsity problem by providing potential preference scores between users and items; (c) integrate with the individual preference for better results; (d) adjust the confidence of pairwise preferences.

(2) We proposed three recommendation models for one-class collaborative filtering, including PBPR, PBPR* and DBPL. They are all extensions of the popular pairwise learning model BPR and relax the limitation in BPR that the equal importance of the unobserved items by considering the relative preference of users on any two unobserved items. In addition, motivated by the assumption that the user's preference difference between the observed item and the unobserved item can be reduced by fusing a relatively smaller preference difference between another pair of items, DBPL is proposed by taking two pairwise preferences into the previous model for further alleviating the vanishing gradient problem in the learning process. Three recommendation models can be realized based on only user-item interactions without any additional social information and can be adjusted for the social information from a specific dataset.

(3) Recommendation methods based on PBPR, PBPR* or DBPL only need information about historical interactions, which is easy to obtain. They could be considered as generic recommendation models and improved by considering more auxiliary information for specific recommended scenarios, such as measuring users' relative preference differences on microblogs according to the topic information [97]. Empirical results show that PBPR, PBPR* and DBPL significantly improve the previous model performance with all evaluation metrics in three real-world datasets indicating the effectiveness of assumptions adopted by the proposed models. The DBPL-based recommendation method performs best among all recommendation methods in all cases. It is obvious from the experimental results that recommendation methods based on our proposed models can provide more accurate recommendations.

6.2 Future Work

For future work, we are interested in (1) considering side information from other domains (affective computing, data mining), such as using sentiment analysis techniques [98] to analyze why users like items (user A prefers action movies) or study users' preferences by considering the auxiliary feedback via the hybrid emotion recognition system [99]; this helps to construct explainable recommender systems; (2) exploring the multi-task learning tasks, which conduct the calculation of potential preference scores and recommendation model learning together.

We observe several meaningful directions. Recently, hash collaborative filtering has become a popular and efficient recommendation technique. It can reduce the storage requirement and make similarity calculations efficient by learning the binary representations of users and items [100]. In addition, recommender systems must be designed to interact with human beings. Researches about the issues and advantages of bridging different fields for the development of recommender systems shaping cognitive architectures should be received sufficient attention for further providing recommendations to agents in order to interact with the environment [101]. Therefore, we will also focus on (1) investigating hash collaborative filtering; and (2) studying more bridging cognitive models for the recommendation in the future.

Bibliography

[1] J. Deng, F. Ren, Multi-label Emotion Detection via Emotion-Specified Feature Extraction and Emotion Correlation Learning, IEEE Transactions on Affective Computing. DOI 10.1109/TAFFC.2020.3034215 (2020).

[2] E. Cambria, Affective computing and sentiment analysis, IEEE intelligent systems. 31(2) (2016) 102-107.

[3] F. Ren, Q. Zhang, An emotion expression extraction method for Chinese microblog sentences, IEEE Access. 8(1) (2020) 69244-69255.

[4] C. Wu, F. Wu, S. Wu, Z. Yuan, Y. Huang, A hybrid unsupervised method for aspect term and opinion target extraction, Knowledge-Based Systems. 148 (2018) 66-73.

[5] A. Da'u, N. Salim, I. Rabiu, A. Osman, Weighted aspect-based opinion mining using deep learning for recommender system, Expert Systems with Applications. 140 (2020) 112871.

[6] X. Wang, X. He, Y. Cao, M. Liu, T, Chua, Kgat: Knowledge graph attention network for recommendation, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 950-958.

[7] J. Chen, H. Zhang, X. He, L. Nie, W. Liu, T. Chua, Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention, in: Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval, 2017, pp. 335-344.

[8] Zhang Y, Shi Z, Zuo W, et al. Joint Personalized Markov Chains with social network embedding for cold-start recommendation, Neurocomputing, 2020, 386: 208-220.

[9] Y. Shi, M. Larson, A. Hanjalic, Collaborative filtering beyond the user-item matrix: A survey of the state of the art and future challenges, ACM Computing Surveys (CSUR) 47(1) (2014) 1-45.

[10] F. Zhang, N. J. Yuan, D. Lian, X. Xie, W. Ma, Collaborative knowledge base embedding for recommender system, in: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. 2016, pp. 353-362.

[11] Q. Liu, H. Wu, Y. Ye, H. Zhao, C. Liu, D. Du, Patent Litigation Prediction: A Convolutional Tensor Factorization Approach, in: Proceeding of the 27th International Joint Conferences on Artifical Intelligence, 2018, pp. 5052-5059.

[12] G. Li, Z. Zhang, L. Wang, Q. Chen, J. Pan, One-class collaborative filtering based on rating prediction and ranking prediction, Knowledge-Based Systems. (124) 2017 46-54.

[13] Y. Shi, A. Karatzoglou, L. Baltrunas, M. Larson, N. Oliver, A. Hanjalic, CLiMF: learning to maximize reciprocal rank with collaborative less-is-more filtering, in: Proceedings of the sixth ACM Conference on Recommender systems, 2012, pp. 139-146.

[14] S. Kant, T. Mahara, Merging user and item based collaborative filtering to alleviate data sparsity, International Journal of System Assurance Engineering and Management, 9(1) (2018) 173-179.

[15] Y. Hu, F. Xiong, S. Pan, X. Xiong, L. Wang, H. Chen, Bayesian personalized ranking based on multiple-layer neighborhoods, Information Sciences, 542 (2021) 156-176.

[16] Z. Zhang, M. Dong, K. Ota, Y. Kudo, Alleviating new user cold-start in user-based collaborative filtering via bipartite network, IEEE Transactions on Computational Social Systems, 7(3) (2020) 672-685.

[17] W. Yue, Z. Wang, W. Liu, B. Tian, S. Lauria, X. Liu, An optimally weighted user-and item-based collaborative filtering approach to predicting baseline data for Friedreich's Ataxia patients, Neurocomputing, 419 (2021) 287-294.

[18] W. Ma, M. Zhang, Y. Cao, W. Jin, C. Wang, Jointly learning explainable rules for recommendation with knowledge graph, in: Proceedings of the nineteenth World Wide Web Conference. 2019, pp. 1210-1221.

[19] 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, 2010, pp. 135-142.

[20] Y. Sun, N. J. Yuan, X. Xie, K. McDonald, R. Zhang, Collaborative intent prediction with real-time contextual data, ACM Transactions on Information Systems (TOIS) 35(4) (2017) 1-33.

[21] H. Wang, F. Zhang, M. Hou, X. Xie, M. Guo, Q. Liu, Shine: Signed heterogeneous information network embedding for sentiment link prediction, in: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, 2018, pp. 592-600.

[22] Y. Hu, F. Xiong, D. Lu, H. Chen, Movie collaborative filtering with multiplex implicit feedbacks, Neurocomputing 398 (2020) 485-494.

[23] 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.

[24] Q. Zhang, F. Ren, Potential Preference Based Bayesian Personalized Ranking for One-Class Collaborative Filtering, in: Proceedings of the fifth International Conference on Natural Language Processing and Knowledge Engineering (NLP-KE'20), 2020, pp. 11.

[25] R. Pan, Y. Zhou, B. Cao, N. N. Liu, R. M. Lukose, M. Scholz, Q. Yang, One-class collaborative filtering, in: Proceedings of the 8th IEEE International Conference on Data Mining, 2008, pp. 502–511.

[26] D. Nichols, Implicit rating and filtering, in: Proceedings of the 5th DELOS Workshop on Filtering and Collaborative Filtering, 1998, pp. 31-36.

[27] Z. Cheng, X. Chang, L. Zhu, R. Kanjirathinkal, M. Kankanhalli, MMALFM: Explainable recommendation by leveraging reviews and images, ACM Transactions on Information Systems (TOIS), 2019, 37(2): 16.

[28] 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.

[29] Y. Guo, Z. Cheng, L. Nie, Y. Wang, J. Ma, M. Kankanhalli, Attentive long short-term preference modeling for personalized product search, ACM Transactions on Information Systems (TOIS) 37(2) (2019) 19.

[30] W. Cai, J. Zheng, W. Pan, J. Lin, L. Li, L. Chen, X. Peng, Z. Ming, Neighborhood-enhanced transfer learning for one-class collaborative filtering, Neurocomputing 341 (2019) 80-87.

[31] W. Pan, M. Liu, Z. Ming, Transfer learning for heterogeneous one-class collaborative filtering, IEEE Intelligent Systems 31(4) (2016) 43-49.

[32] S. Rendle, C. Freudenthaler, Z. Gantner, L. Schmidt-Thieme, BPR: Bayesian personalized ranking from implicit feedback, in: *UAI*, 2009, pp. 452-461.

[33] S. Rendle, C. Freudenthaler, Improving pairwise learning for item recommendation from implicit feedback, in: Proceedings of the 7th ACM international conference on Web search and data mining, 2014, pp. 273-282.

[34] S. Yang, B. Long, A. Smola, H. Zha, Z. Zheng, Collaborative competitive filtering: learning recommender using context of user choice, in: Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2011, pp. 295–304.

[35] S. Rendle, L. Schmidt-Thieme, Pairwise interaction tensor factorization for personalized tag recommendation, in: Proceedings of the 3rd ACM International Conference on Web Search and Data Mining, 2010, pp. 81–90.

[36] S. Kabbur, X. Ning, G. Karypis, Fism: factored item similarity models for top-n recommender systems, in: Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, 2013, pp. 659-667.

[37] S. Purushotham, Y. Liu, C. C. J. Kuo, Collaborative topic regression with social matrix factorization for recommendation systems, in: Proceedings of the 29th ACM International Conference on Machine Learning, Edinburgh, Scotland, UK, ACM (2012), pp. 1255-1265.

[38] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, Tat-Seng Chua, Neural collaborative filtering, in: Proceedings of the 26th International Conference on World Wide Web, 2017, pp. 173–182.

[39] G. Adomavicius, A. Tuzhilin, Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions, IEEE transactions on knowledge and data engineering. 17(6) (2005) 734-749.

[40] X. Geng, H. Zhang, J. Bian, T. Chua, Learning image and user features for recommendation in social networks, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 4274-4282.

[41] Y. Li, J. Hu, C. Zhai, Y. Chen, Improving one-class collaborative filtering by incorporating rich user information, in: Proceedings of the 19th ACM international conference on Information and knowledge management. 2010, pp. 959-968.

[42] J. Chen, D. Lian, K. Zheng, Improving One-Class Collaborative Filtering via Ranking-Based Implicit Regularizer, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2019, pp. 37-44.

[43] H. Ying, L. Chen, Y. Xiong, J. Wu, Collaborative deep ranking: A hybrid pair-wise recommendation algorithm with implicit feedback, in: Proceedings of the Pacific-asia conference on knowledge discovery and data mining, 2016, pp. 555-567.

[44] J. Feng, Z. Xia, X. Feng, J. Peng, RBPR: A hybrid model for the new user cold start problem in recommender systems, Knowledge-Based Systems 214 (2021) 106732.

[45] X. Wang, X. He, M. Wang, F. Feng, T. Chua, Neural graph collaborative filtering, in: Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval, 2019, pp. 165-174.

[46] W. Yuan, H. Wang, X. Yu, N. Liu, Z. Li, Attention-based context-aware sequential recommendation model, Information Sciences 510 (2020) 122-134.

[47] W. Li, X. Zhou, S. Shimizu, M. Xin, J. Jiang, H. Gao, Q. Jin, Personalization recommendation algorithm based on trust correlation degree and matrix factorization, IEEE Access, 2019, 7: 45451-45459.

[48] J. Ding, G. Yu, X. He, Y. Quan, Y. Li, T. S. Chua, D. Jin, J. Yu, Improving Implicit Recommender Systems with View Data, in: Proceeding of the Twenty-Seventh International Joint Conference on Artificial Intelligence, 2018, pp.3343-3349.

[49] H. Qiu, Y. Liu, G. Guo, Z. Sun, J. Zhang, H. T. Nguyen, BPRH: Bayesian personalized ranking for heterogeneous implicit feedback, Information Sciences 453 (2018) 80-98.

[50] Y. Li, R. Wang, G. Nan, D. Li, M. Li, A personalized paper recommendation method considering diverse user preferences, Decision Support Systems (2021) 113546.

[51] 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, 2014, pp. 261-270.

[52] J. Sun, R. Ying, Y. Jiang, J. He, Z. Ding, Leveraging friend and group information to improve social recommender system, Electronic Commerce Research 20(1) (2020) 147-172.

[53] L. A. G. Camacho, S. N. Alves-Souza, Social network data to alleviate cold-start in recommender system: A systematic review, Information Processing & Management 54(4) (2018) 529-544.

[54] X. Zhou, G. Guo, Z. Sun, Y. Liu, Multi-facet user preference learning for fine-grained item recommendation, Neurocomputing. 385 (2020) 258-268.

[55] W. Zhou, J. Li, Y. Zhou, M. H. Memon, Bayesian pairwise learning to rank via one-class collaborative filtering, Neurocomputing 367 (2019) 176-187.

[56] 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.

[57] R. Yu, Y. Zhang, Y. Ye, L. Wu, C. Wu, Q. Liu, E. Chen, Multiple pairwise ranking with implicit feedback, in: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, 2018, pp. 1727-1730.

[58] H. Liu, Z. Wu, X. Zhang, CPLR: Collaborative pairwise learning to rank for personalized recommendation, Knowledge-Based Systems, 148 (2018) 31-40.

[59] Z. Gantner, L. Drumond, C. Freudenthaler, L. Schmidt-Thieme, Personalized ranking for non-uniformly sampled items, in: Proceedings of KDD Cup 2011, 2012, pp. 231-247.

[60] W. Zhang, T. Chen, J. Wang, Y. Yu, Optimizing top-n collaborative filtering via dynamic negative item sampling, in: Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval, 2013, pp. 785-788.

[61] J. Ding, G. Yu, X. He, Y. Li, D. Jin, Sampler design for bayesian personalized ranking by leveraging view data, IEEE Transactions on Knowledge and Data Engineering. 2019.

[62] Z. Jiang, H. Liu, B. Fu, Z. Wu, T. Zhang, Recommendation in heterogeneous information networks based on generalized random walk model and bayesian personalized ranking, in: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, 2018, pp. 288-296.

[63] B. Wu, Y. Ye, BSPR: Basket-sensitive personalized ranking for product recommendation, Information Sciences 541 (2020) 185-206.

[64] Q. Zhang, F. Ren, Prior-based bayesian pairwise ranking for one-class collaborative filtering,

Neurocomputing 440 (2021) 365-374.

[65] Q. Zhang, F. Ren, Double bayesian pairwise learning for one-class collaborative filtering, Knowledge-Based Systems 229 (2021) 107339.

[66] X. Ning, C. Desrosiers, G. Karypis, A comprehensive survey of neighborhood-based recommendation methods, Recommender systems handbook, Springer, Boston MA (2015) 37-76.

[67] Y. Koren, R, Bell, Advances in collaborative filtering, Recommender systems handbook, Springer, Boston MA (2015) 77-118.

[68] Shen Y, Jin R. Learning personal+ social latent factor model for social recommendation, in: Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, 2012, pp. 1303-1311.

[69] Y, Koren, R, Bell, C, Volinsky, Matrix factorization techniques for recommender systems, Computer 42(8) (2009) 30-37.

[70] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, arXiv preprint arXiv:1301.3781, (2013).

[71] A. Mnih, R.R. Salakhutdinov, Probabilistic matrix factorization, in: Proceedings of the twenty-second Advances in neural information processing systems. 2008, pp. 1257-1264.

[72] O. Chapelle, M. Wu, Gradient descent optimization of smoothed information retrieval metrics, Information retrieval 13(3) (2010) 216-235.

[73] S Ruder, An overview of gradient descent optimization algorithms, arXiv preprint arXiv:1609.04747, (2016).

[74] Y. Lei, Y. Ying, Fine-grained analysis of stability and generalization for stochastic gradient descent, in: Proceeding of the International Conference on Machine Learning, 2020, pp. 5809-5819.

[75] A. Jentzen, B. Kuckuck, A. Neufeld, P. Wurstemberger, Strong error analysis for stochastic gradient descent optimization algorithms, IMA Journal of Numerical Analysis 41(1) (2021) 455-492.

[76] Y. A. Ghassabeh, F. Rudzicz, H. A. Moghaddam, Fast incremental LDA feature extraction, Pattern Recognition 48(6) (2015) 1999-2012.

[77] M. Pavlinek, V. Podgorelec, Text classification method based on self-training and LDA topic models, Expert Systems with Applications 80 (2017) 83-93.

[78] Z. Liu, Q. Xie, M. Wu, W. Cao, Y. Mei, J. Mao, Speech emotion recognition based on an improved brain emotion learning model, Neurocomputing 309 (2018) 145-156.

[79] C. M. Bishop, Pattern Recognition and Machine Learning Errata, IEEE Transactions on Information Theory, 9(4) (2006) 257-261.

[80] M. Steyvers, T. Griffiths, Probabilistic topic models, IEEE Signal Processing Magazine, 27(6) (2010) 55-65.

[81] E. Palumbo, G. Rizzo, R. Troncy, Entity2rec: Learning user-item relatedness from knowledge graphs for top-n item recommendation, in: Proceedings of the eleventh ACM conference on recommender systems, 2017, pp. 32-36.

[82] E. Çano, M. Morisio, Music mood dataset creation based on last. fm tags, in: Proceedings of the 2017 International Conference on Artificial Intelligence and Applications, 2017, pp. 15-26.

[83] Liang L, Tang R. An Improved Collaborative Filtering Algorithm Based on Node2vec, in: Proceedings of the 2018 2nd International Conference on Computer Science and Artificial Intelligence, 2018, pp. 218-222.

[84] T. A. Dang, E. Viennet, Collaborative filtering in social networks: A community-based approach, in: Proceedings of the 2013 International Conference on Computing, Management and Telecommunications (ComManTel), 2013, pp.128-133.

[85] W. Zhang, T. Chen, J. Wang, Y. Yu, Optimizing top-n collaborative filtering via dynamic negative item sampling, in: Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval, 2013, pp. 785-788.

[86] S. Forouzandeh, K. Berahmand, M. Rostami, Presentation of a recommender system with ensemble learning and graph embedding: a case on MovieLens, Multimedia Tools and Applications 80(5) (2021) 7805-7832.

[87] L. Chen, P. Pu, Users' eye gaze pattern in organization-based recommender interfaces, in: Proceedings of the 16th international conference on intelligent user interfaces, 2011, pp. 311-314.

[88] A. Gazdar, L. Hidri, A new similarity measure for collaborative filtering based recommender systems, Knowledge-Based Systems. 188 (2020) 105058.

[89] Y. Wu, F. Ren, Exploiting opinion distribution for topic recommendation in Twitter, IEEJ Transactions on Electrical and Electronic Engineering. 10(5) (2015) 567-575.

[90] S. Balakrishnan, S. Chopra, Collaborative ranking, in: Proceedings of the fifth ACM international conference on Web search and data mining, 2012, pp. 143-152.

[91] Y. Zhang, X. Chen, Explainable recommendation: A survey and new perspectives, arXiv preprint arXiv:1804.11192 (2018).

[92] N. Tintarev, J. Masthoff, A survey of explanations in recommender systems in: Proceedings of the IEEE 23rd international conference on data engineering workshop, 2007, pp. 801-810.

[93] H. Wang, F. Zhang, J. Wang, M. Zhao, W. Li, X. Xie, M. Guo, Ripplenet: Propagating user preferences on the knowledge graph for recommender systems, in: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, 2018, pp. 417-426.

[94] Y. Zhang, G. Lai, M. Zhang, Y. Zhang, Y. Liu, S. Ma, Explicit factor models for explainable recommendation based on phrase-level sentiment analysis, in: Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval, 2014, pp. 83-92.

[95] K. Bauman, B. Liu, A. Tuzhilin, Aspect based recommendations: Recommending items with the most valuable aspects based on user reviews, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2017, pp. 717-725.

[96] B. Wang, M. Ester, J. Bu, D. Cai, Who also likes it? generating the most persuasive social explanations in recommender systems, in: AAAI, 2014, pp.173-179.

[97] Q. Zhang, S. Zhang, F. Ren, X. Zhang, Aspect-Based Opinion and Affective Meaning in Microblogging Comments Via Topic Model, JOURNAL OF CHINESE INFORMATION PROCESSING 33(6) (2019) 116-123,140.

[98] F. Ren, Affective Information Processing and Recognizing Human Emotion, Electronic Notes in Theoretical Computer Science 225(2009) (2009) 39-50.

[99] Y. Gu, X. Zhang, Z. Liu, F. Ren, WiFE: WiFi and Vision based Intelligent Facial-Gesture Emotion Recognition, arXiv preprint arXiv:2004.09889 (2020).

[100] Y. Li, S. Wang, Q. Pan, H. Peng, T. Yang, E. Cambria, Learning binary codes with neural collaborative filtering for efficient recommendation systems, Knowledge-Based Systems. 172 (2019) 64-75.

[101] C. Angulo, Z. Falomir, D. Anguita, N. Agell, E. Cambria, Bridging Cognitive Models and Recommender Systems, Cognitive Computation. (12) (2020) 426-427.

[102] R. Pan, Y. Zhou, B. Cao, N. N. Liu, R. Lukose, M. Scholz, Q. Yang, One-class collaborative filtering, in: Proceedings of the 2008 Eighth IEEE International Conference on Data Mining, 2008, pp. 502-511.