Knowledge-Based Systems 239 (2022) 107828

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

SPR: Similarity pairwise ranking for personalized recommendation

Junrui Liu, Zhen Yang, Tong Li *, Di Wu, Ruiyi Wang

Faculty of Information Technology, Beijing University of Technology, 100124, China

ARTICLE INFO

Article history:
Received 12 February 2021
Received in revised form 22 November 2021
Accepted 25 November 2021
Available online 11 December 2021

Keywords:
Recommender system
Pairwise method
Similar item pair
Matrix factorization

ABSTRACT

Bayesian personalized ranking (BPR) has been proposed as an effective method to model pairwise learning, and it is widely used in many personalized recommender systems. However, the effectiveness of BPR can be seriously affected by an imbalanced data distribution because it tends to rank popular items ahead of personalized items. As a result, the personalized needs of users cannot be well met. In this paper, we propose a novel personalized recommendation method called similarity pairwise ranking (SPR) to rank users' favorite items first. SPR eliminates the differences in the scores between popular and personalized items based on their similarity by using a new penalty. In such a way, the SPR-enhanced recommendation will render meaningful and personalized results that better meet the individual needs of users, and it overcomes the negative impact of imbalanced datasets. We design a model to illustrate the improvement of SPR: similarity pairwise ranking matrix factorization (SPRMF). Experimental results obtained using six datasets indicate the superiority in recommendation quality of SPRMF over the recent state-of-the-art methods.

© 2021 Elsevier B.V. All rights reserved.

1. Introduction

The rapid creation and dissemination of information continuously exacerbates the problem of information explosion. Recommender systems have been widely used in various areas as an effective means to tackle this problem. At the same time, recommender systems are also the primary tool to meet the individual needs of users. As an example, the majority of viewing hours on Netflix are generated through the videos that are recommended by recommender systems [1], Some e-commerce websites use a recommender system to increase the conversion rates of users by studying users' online shopping behaviors [2–4]. Meanwhile, there is a system that is designed to help scholars find suitable papers [5].

Among recommendation strategies, collaborative filtering algorithms use the wisdom and behavior of the public to achieve good performance, and thus they have attracted the attention of many researchers [6,7]. Bayesian personalized ranking (BPR) is one such collaborative filtering method, and it is seminal in modeling pairwise learning from the Bayesian perspective [8]. BPR tries to learn from the local ranking relationships to the global ranking of items. Unlike the pointwise method, which maintains consistency with the original information, the pairwise method is designed to capture the existence of partial order among items in the original data. BPR for learning has been recently applied to point-of-interest recommendation [9], visual personalized ranking [10], and graph convolutional networks for collaborative filtering [11,12].

However, datasets derived from the real world are often imbalanced. It has been pointed out that the effectiveness of BPR can be seriously affected by an imbalanced data distribution [13–15]. BPR conducts ineffective learning in training because popular items have more increments than personalized items. It tends to rank popular items ahead of personalized items, and as a result, the personalized needs of users cannot be well met. Many studies have attempted to overcome this problem. The previous pairwise methods can be divided into three categories. First, group-based methods [7,16] solve the problem by introducing group preferences. The source of a group preference is uncertain, and these methods do not punish the scores of popular items. Second, the absolute rating is used to learn the relative relations and absolute information at the same time [17,18]. However, it is inappropriate to use negative samples to represent the missing information in recommender systems [19]. Third, there are also some methods that sample the original dataset into a uniform dataset, but this approach also loses the information on popularity [15,20,21].

The present paper proposes a personalized recommendation method called similarity pairwise ranking (SPR) to overcome the aforementioned problem and rank the users' favorite items first. SPR eliminates the differences in the scores between popular and personalized items based on their similarity by using a new

* Corresponding author.
E-mail addresses: liujunrui@emails.bjut.edu.cn (J. Liu), yangzhen@bjut.edu.cn (Z. Yang), litong@bjut.edu.cn (T. Li), wuxiaodou@emails.bjut.edu.cn (D. Wu), wangruiyi@emails.bjut.edu.cn (R. Wang).

https://doi.org/10.1016/j.knosys.2021.107828
0950-7051/© 2021 Elsevier B.V. All rights reserved.

penalty. As a fundamental feature of items, the item similarity is used in many recommender systems [22]. The KNN method uses historical records to calculate the similarities between items [6]. At the same time, graph regularization and homophily regularization are added to constrain the representations of items [23,24]. Content-based methods also generate recommendation lists by capturing the similarities in the content [25]. The top-ranked items and the items that interact with a user have higher scores in terms of content similarity. Item similarity exists in a wide range of items, including both popular items and personalized items. Additionally, different users can have different views. The same ratings of items illustrate the similarity between different items, i.e., items that satisfy the taste of users are similar to a certain degree. SPR emphasizes the similarity between popular items and personalized items from the perspective of the user. The differences in scores between popular and personalized items is eliminated by using a penalty based on their similarity. The impact of popularity on scores is reduced by our similarity constraints. SPR makes recommendation results by considering user preferences. Personalized items have a large chance to get ahead of popularity items. Thus, the recommendation results are more in line with the individual needs of users.

Our contributions to the literature are summarized as follows:

- We propose a personalized recommendation method called similarity pairwise ranking, which considers the similarities of items. SPR can overcome the influence of an imbalanced data distribution, and the SPR-enhanced recommendation will render meaningful and personalized results that better meet the individual needs of users.
- In SPR, similar item pairs are used to constrain the differences in scores between pairs of items. The same ratings of items illustrate the similarities between different items, i.e., items that satisfy the users' tastes are similar to a certain degree. We utilize a sample method based on ratings to obtain similar item pairs.
- We design a model to illustrate the improvement of SPR: similarity pairwise ranking matrix factorization (SPRMF). Experimental results obtained using six datasets indicate the superiority in the recommendation quality of SPRMF over recent state-of-the-art methods.

The remainder of this paper is organized as follows. Section 2 introduces the preliminaries of the proposed model. In Section 3, we describe our proposed SPR method in detail. We also design a model based on matrix factorization to illustrate the improvements brought by SPR. Section 4 presents the experiments and discusses the experimental results. In Section 5, we review some previous related studies. Finally, we conclude the paper with some remarks and future research directions.

2. Preliminaries

In this section, we first formulate the recommendation problem. Then, the pairwise loss function used in this paper is introduced.

2.1. Recommendation problem

Given a recommendation problem, assume that a user set $\mathcal{U} = \{u^1, u^2, \ldots, u^m\}$ and an item set $\mathcal{I} = \{i^1, i^2, \ldots, i^n\}$ contain m users and n items, respectively. Let $R \in \mathbb{R}^{m*n}$ denote the rating matrix, where r_{ui} is the rating of user u on item i, and we mark *unk* if it is unknown. We list some commonly used notations in Table 1. Most of the existing solutions for recommendations involve constructing the interaction matrix P. Those solutions construct the user–item interaction matrix $P \in \mathbb{R}^{m*n}$ from R with implicit feedback, as follows:

$$P_{ui} = \begin{cases} 0 & if\ r_{ui}\ unk, \\ 1 & otherwise. \end{cases} \tag{1}$$

Recommender systems are commonly formulated to estimate the score of each unobserved entry in P, which are then used for ranking the items. Given a user u history behavior $\mathcal{I}_u = \{i|\forall i \in \mathcal{I}, r_{ui}\ is\ not\ unk\}$, our goal is to provide u a personalized ranking list of items from the remaining item set $\mathcal{I}\backslash\mathcal{I}_u = \{i|\forall i \in \mathcal{I}, r_{ui}\ is\ unk\}$.

Table 1
Notations and explanations used in the paper.

Notation	Explanation
m	Number of users
n	Number of items
$\mathcal{U}$	User set, $\mathcal{U} = \{u^1, u^2, \ldots u^m\}$
$\mathcal{I}$	Item set, $\mathcal{I} = \{i^1, i^2, \ldots i^n\}$
u	A user in $\mathcal{U}$
i, j, q	An item in $\mathcal{I}$
D	Training dataset
r_{ui}	Rating of user u on item i, $r_{ui} \in \{1, \ldots, mr\}$
R	Rating matrix, $R \in \mathbb{R}^{m*n}$
mr	Max rating in R
p_{ui}	Interaction of user u on item i, $p_{ui} \in \{0, 1\}$
P	Interaction matrix, $P \in \mathbb{R}^{m*n}$
unk	Unknown
$\mathcal{I}_u$	User history behavior, $u \in \mathcal{U}, \mathcal{I}_u = \{i\|\forall i \in \mathcal{I}, r_{ui}\ is\ not\ unk\}$.
$\mathcal{I}_u^1$	User history behavior subset, $u \in \mathcal{U}, \mathcal{I}_u = \{i\|\forall i \in \mathcal{I}, r_{ui} = 1\}$.
$\mathcal{I}\backslash\mathcal{I}_u$	Remaining item set, $u \in \mathcal{U}, \mathcal{I}\backslash\mathcal{I}_u = \{i\|\forall i \in \mathcal{I}, r_{ui}\ is\ unk\}$.
k	Latent dimension
X	User matrix, $X \in \mathbb{R}^{m*k}$
X_u	Vector representation of user u, uth line in X
Y	Item matrix, $Y \in \mathbb{R}^{n*k}$
Y_i	Vector representation of item i, ith line in Y

2.2. Pairwise loss function

In pairwise methods, the partial order is described by many pairs of items. A pair of items $i >_u j$ indicates that user u prefers item i to item j. For one user, given a positive sample item i and a negative sample (or "unknown data") item j, the predicted score of item i should be higher than that of item j [8]. The objective function is defined as follows:

$$\begin{aligned} \mathcal{L}_{pair} &= max \sum_{u,i,j \in D} (\Theta \mid i >_u j) \\ &= max \sum_{u,i,j \in D} \ln \sigma\left(\hat{p}_{ui} - \hat{p}_{uj}\right) - \lambda_\Theta \| \Theta \|^2, \end{aligned} \tag{2}$$

In Eq. (2), σ is the sigmoid function $\sigma(x) = 1/(1 + \exp(-x))$, $\hat{p}_{ui}$ is the predicted score of user u on item i from a model, and λ_Θ is the hyperparameter of the regularization.

2.3. The effect of imbalanced data on BPR

The distribution of imbalanced data can seriously affect the effectiveness of BPR [13]. BPR tends to rank popular items ahead of personalized items. We denote $\hat{p}_{uij} = \hat{p}_{ui} - \hat{p}_{uj}$. By using stochastic gradient descent (SGD), each pair of items $i >_u j$ is updated:

$$\Delta\Theta = \left[1 - \sigma\left(\hat{p}_{uij}\right)\right] \frac{\partial \hat{p}_{uij}}{\Theta} - \lambda\Theta. \tag{3}$$

From Eq. (3), it can be seen that the gradient of the parameters is related to $\hat{p}_{uij}$. In the learning process, each gradient of popular items decreases as the difference between positive and negative

sample scores increases. When encountering imbalanced data, popular items obtain many small gradients. Hence, the score of popular items increases to a very large value after several training iterations, even if the gradient is constantly shrinking. If a user obtains an item list from the BPR, popular items could have a greater chance of ranking ahead of personalized items. This outcome the effect of imbalanced data on BPR.

3. Similarity pairwise ranking

In this section, we propose a novel personalized recommendation method called similarity pairwise ranking (SPR). SPR considers item–item relations and emphasizes the similarity between two positive samples from the perspective of users. This approach matches the emphasis of BPR on the relationships between items from the perspective of users. We first discuss the objective function of SPR, which includes a new penalty. Then, a sampling approach is introduced to show how to use ratings to obtain a similar item pair. Third, we solve the objective function, and present the generalized algorithm. Finally, we apply SPR to matrix factorization, and we propose a novel model called pairwise ranking matrix factorization (SPRMF) to optimize the matrix factorization (MF) model for ranking.

3.1. Objective function

The similarity between items is one of the primary ways to solve the recommendation problem, and it can be found in many designs [6,23–25]. The top-ranked items and the items that interact with a user have higher scores in content similarity. Item similarity exists in a wide range of items, including both popular items and personalized items. Therefore, we propose that the item similarity can improve the performance of recommender systems. We narrowed the score between similar item pairs in such a way that the impact of imbalanced datasets is avoided. Thus, the recommendation results are more in line with the individual needs of users.

Following the key idea of BPR, we define the concept of item similarity:

Similar item pair: *Given a user $u \in \mathcal{U}$, where $\mathcal{I}_u = \{i|\forall i \in \mathcal{I}, r_{ui}$ is not unk$\}$ represents an item set that is a subset of $\mathcal{I}$ wherein each item has interacted with u. $\mathcal{I}\backslash\mathcal{I}_u = \{i|\forall i \in \mathcal{I}, r_{ui}$ is unk$\}$ represents an item set wherein each item has not interacted with u and belongs to item set $\mathcal{I}$. A similar item pair $\langle i, q|u, j\rangle$ is defined as u, and prefers items i and q to item j; i and q are similar for user u. More specifically, $i, q \in \mathcal{I}_u$ and $j \in \mathcal{I}\backslash\mathcal{I}_u$*

BPR enhances the differences in the scores between positive samples and negative samples. Using similar item pairs, we propose a new penalty $\hat{p}_{uiq} = \hat{p}_{ui} - \hat{p}_{uq}$ to narrow the score difference between two positive samples to ensure that two items have similar values. Therefore, the model must carefully learn the item information that meets the individual needs of users, not the popularity of the items. Following Eq. (2), the objective function of similarity pairwise ranking is defined as follows:

$$\begin{aligned}\mathcal{L}_{SPR} = min \sum_{u,i,j,q\in D} & -(1-\alpha)\ln\sigma\left(\hat{p}_{uij}\right) \\ & + \alpha\ln\sigma\left(\hat{p}_{uiq}\right) + \lambda_\Theta \| \Theta \|^2,\end{aligned} \tag{4}$$

where α is a hyperparameter, λ_Θ is a regularization coefficient, and σ denotes the sigmoid function $\sigma(x) = 1/(1+\exp(-x))$.

In the training process, the training frequency of a popular item is much higher than that of a personalized item. This circumstance makes the scores of popular items increase as the training time increases. The score gap between popular items and personalized items makes it difficult to recommend personalized items. SPR calculates the similarity between two items to reduce the differences in the scores between items. The relationship between popular items and personalized items is implicitly described by capturing the similarity between two items in SPR. We narrowed the score between popular items and personalized items by their similarity in such a way that the impact of imbalanced datasets was avoided. An advantage of the SPR method is that the SPR-based recommender system can learn more useful features and generate more personalized recommendations when the difference among item scores is small.

3.2. Sampling similar item pairs

In the previous subsection, we defined a key concept called a similar item pair. A question that we must answer is how to sample a similar item pair from a dataset. The same ratings of items illustrate the similarities between different items, i.e., items that satisfy users' tastes are similar to a certain degree. We utilize a sample method based on the ratings to obtain similar item pairs.

A large number of existing datasets contain user ratings on items. However, in the recommendation task, the rating matrix is often converted into an interactive matrix by Eq. (1), and the recommendation method is used for learning and prediction. Through careful consideration of users, we find that the ratings represent the personalized taste of the user, and that similar items have the same rating. Users think about the difference between the current item and previous items when they give a rating.

Formally, in the rating matrix R, the range of each rating $r_{ui} \in R$ is from 1 to mr. For each user, we first extract the set of items $\mathcal{I}_u$ that the user has interacted with from the rating matrix. Second, $\mathcal{I}_u$ is split into mr subsets ${\mathcal{I}_u}^1, \ldots, {\mathcal{I}_u}^{mr}$ according to rating value, and thus, ${\mathcal{I}_u}^1 = \{i|\forall i \in \mathcal{I}, r_{ui} = 1\}$. Then, for each known rating $r_{ui} \in R$ for item i, we randomly select an item q from ${\mathcal{I}_u}^{r_{ui}}$ and an item j from $\mathcal{I}\backslash\mathcal{I}_u$. Finally, we obtain a similar item pair $\langle i, q|u, j\rangle$.

3.3. SPR learning algorithm

We follow the widely used stochastic gradient descent (SGD) algorithm to optimize the objective function in Eq. (4). To simplify the representation, we define

$$z_1 = z_{uij} = 1 - \sigma\left(\hat{p}_{ui} - \hat{p}_{uj}\right), \tag{5}$$

$$z_2 = z_{uiq} = 1 - \sigma\left(\hat{p}_{ui} - \hat{p}_{uq}\right). \tag{6}$$

The negative gradient of parameters in the objective function is calculated as follows:

$$\Delta\Theta = (1-\alpha)\, z_1 \frac{\partial \hat{p}_{uij}}{\partial\Theta} - \alpha z_2 \frac{\partial \hat{p}_{uiq}}{\partial\Theta} - \lambda\Theta. \tag{7}$$

With the above gradients, the model parameters are updated as follows:

$$\begin{aligned}\Theta &= \Theta + lr * \Delta\Theta \\ &= \Theta + lr * \left((1-\alpha)\, z_1 \frac{\partial \hat{p}_{uij}}{\partial\Theta} - \alpha z_2 \frac{\partial \hat{p}_{uiq}}{\partial\Theta} - \lambda\Theta\right),\end{aligned} \tag{8}$$

where lr is the learning rate.

The pseudocode of the SPR learning algorithm is shown in Algorithm 1. It consists of an initialization step and an iterative training process. The initialization step involves randomly setting the model parameters and dividing the user's explicit feedback data $\mathcal{I}_u$ across several subsets ${\mathcal{I}_u}^{r_{ui}}$ according to the rating.

Algorithm 1 Similarity Pairwise Ranking Learning Algorithm

Input: Explicit feedback dataset $\{(u, i, R_{ui})\}$, sampling ratio ng, number of iterations t, learning rate lr, regularization coefficient λ, hyperparameter α.
Output: The model parameters Θ.

```
randomly initialize the parameters Θ.
for each user, draw subset I_u from dataset.
for each subset I_u, draw subset {I_u^1, ..., I_u^mr} by using rating.
for t = 1, ..., t do
  for each (u, i, r_ui) in dataset do
    for n = 1, ..., ng do
      sampling similar item pair ⟨i, q|u, j⟩
      update Θ via Eqs. (7) and (8).
    end for
  end for
end for
Return Θ
```

3.4. Apply similarity pairwise ranking to matrix factorization

SPR is a model-independent top-level method. This method can be combined with models to constrain their training process. In this way, the SPR-enhanced recommendation will render meaningful and personalized results that better meet the individual needs of users, and it overcomes the negative impact of imbalanced datasets. In this subsection, we apply SPR to matrix factorization, and we propose a novel model called pairwise ranking matrix factorization (SPRMF) to optimize the MF model for ranking.

MF assumes that the interactive information matrix P can be reconstructed from two small matrices $X \in \mathbb{P}^{m*k}$ and $Y \in \mathbb{R}^{n*k}$. It learns a latent space to represent users and items. Each line in X is marked as X_u, which represents the embedding representation of a user u in the low-dimensional space; at the same time, each line in Y is marked as Y_i, which represents the embedding of item i in the low-dimensional space. MF assumes the following equation to predict the preference of a user u toward item i:

$$\hat{P}_{ui} = f(u, i|X, Y) = X_u * {Y_i}^T = \sum_{k=1}^{K} X_{uk} * Y_{ik}. \tag{9}$$

To apply the SPR method, we replace the objective function of MF with the SPR objective function in Eq. (4). Specifically, Eq. (9) is used to predict the score $\hat{P}_{ui}$. By replacing the notation Θ in Eq. (4) with MF parameters, the objective function of the SPRMF can be written as follows:

$$\begin{aligned} \mathcal{L}_{SPRMF} = min \sum_{u,i,q,j \in D} & -(1-\alpha) \ln \sigma\left(\hat{p}_{uij}\right) \\ & +\alpha \ln \sigma\left(\hat{p}_{uiq}\right) + \lambda\left(\|X\|^2 + \|Y\|^2\right), \end{aligned} \tag{10}$$

where α is a hyperparameter, σ denotes the sigmoid function $\sigma(x) = 1/1 + \exp(-x)$, and $\hat{p}_{uiq} = \hat{p}_{ui} - \hat{p}_{uq}$.

Matrix factorization can learn the potential characteristics between users and items, and it predicts the interaction information between users and items. After using SPR as the loss function, during prediction, the partial order information between items can be learned while overcoming the impact of data imbalance. When using SPRMF for recommendation, the top-ranked items must have good interactivity with the user, and they have already met the interest preferences shown in the user's historical behavior.

The goal of SPRMF is to find the best X and Y to fit the training data and rank items by scores that are predicted from Eq. (9).

Table 2
The statistics of datasets.

Dataset	#user	#item	#rating	Density (%)
ML-100K	943	1682	100000	4.19
ML-1M	6040	3952	1000209	6.30
ML-10M	69878	10677	10000054	1.34
ML-SL	610	9724	100836	1.70
R3	5050	1000	174497	3.46
R4	2867	10825	145034	0.47

According to Eq. (8), the negative gradient of each parameter obtained by using SGD is as follows:

$$\Delta X_u = (1-\alpha) z_1 \left(Y_i - Y_j\right) - \alpha z_2 \left(Y_i - Y_q\right) - \lambda X_u, \tag{11}$$

$$\Delta Y_i = [(1-\alpha) z_1 - \alpha z_2] X_u - \lambda Y_i, \tag{12}$$

$$\Delta Y_j = -(1-\alpha) z_1 * X_u - \lambda Y_j, \tag{13}$$

$$\Delta Y_q = \alpha z_2 * X_u - \lambda Y_q, \tag{14}$$

where z_1 and z_2 are defined as in Eqs. (5) and (6), respectively. With the above gradients, the model parameters are updated according to Algorithm 1.

We assume that the score p_{ui} is greater than p_{uq} when i is a popular item and q is a personalized item. The parameters are updated using Eqs. (12) and (14). The popular item will receive a differentiated score penalty item to prevent it from further increasing. Personalized items will receive a supplementary gradient to promote the increase in personalized item scores.

4. Experiments

In this section, we describe our experimental setting, and we present the results of comparisons to different types of baseline methods.

4.1. Datasets

We used six datasets for the experiment: four MovieLens datasets,[1] i.e., MovieLens 100K (ML-100K) dataset, MovieLens 1M (ML-1M) dataset, MovieLens 10M (ML-10M), MovieLens Least Small (ML-LS); and two Yahoo! datasets,[2] i.e., Yahoo!R3 (R3) and Yahoo!R4 (R4). For Yahoo! datasets, we filtered out some users with ratings below 20 to make them more consistent with MovieLens datasets. The details of all of the datasets are shown in Table 2.

The characteristics of the datasets are presented in Table 2. Among them, ML-1M has the highest number of ratings and the largest sparsity, which indicates that users have sufficient activities to measure the performance of a model. ML-100K and R3 are used to verify the performance of the model on the user side and item side. According to the number of ratings each item has, we divide the items into 5 groups. We then count the number of items in each group, and we calculate the proportion of each group's total score in the dataset. The details of six datasets are shown in Fig. 1. The first three datasets satisfy the long-tailed distribution. The remaining three datasets have their own characteristics. These six datasets provide a complete and comprehensive analysis of our method.

[1] http://grouplens.org/datasets/movielens/
[2] http://webscope.sandbox.yahoo.com/catalog.php?datatype=r

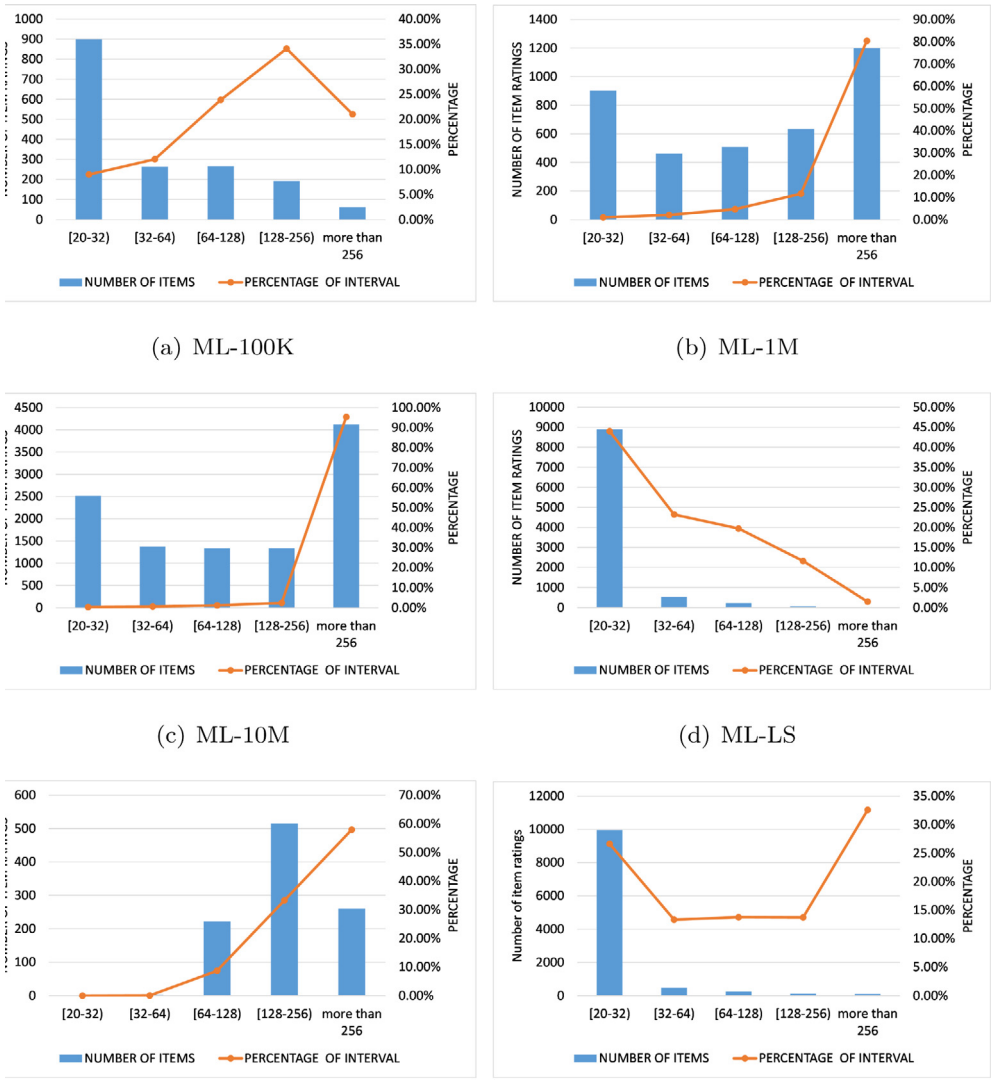


Fig. 1. Comparison of item historical ratings in six different datasets.

4.2. Evaluation methodology

We evaluate the performance of the learning algorithm by separating the data into a training part used to train the model, and a test part used to evaluate it. For each user, 50% of the ratings are randomly selected for training, and the remaining 50% are selected for testing. For each dataset, we repeat the above process 5 times independently, and obtain 5 splits of data. We adopt one widely used ranking metric for the evaluation: normalized discounted cumulative gain NDCG@K (5, 10).

NDCG is a ranking-based measure that assigns higher scores to hits with top ranks. π_u is a permutation of items for user u, and π_u^* is the permutation that generates the maximum of DCG@K.

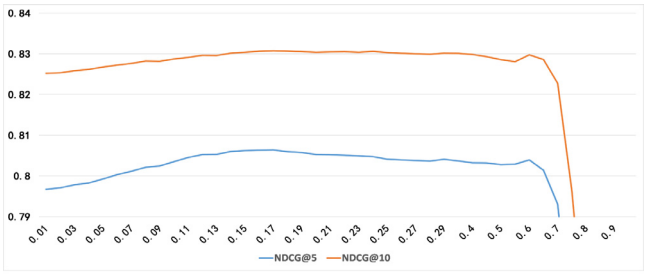
$$DCG@K(u, \pi_u) = \sum_{k=1}^{K} \frac{2^{r_{u,\pi_u(k)}} - 1}{\log_2(k+1)}, \tag{15}$$

$$NDCG@K = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \frac{DCG@K(u, \pi_u)}{DCG@K(u, \pi_u^*)}. \tag{16}$$

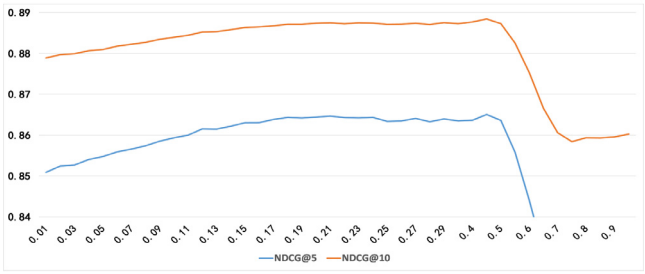
4.3. Baselines for comparison

We compare SPRMF with some of the most advanced models.

- Prod2Vec [4]: Product to vector is a neural language-based algorithm that is specifically tailored for delivering effective product recommendations, which can show popular products and products predicted based on co-occurrence.
- BPMF [26]: Bayesian probabilistic matrix factorization is a fully Bayesian treatment of the probabilistic matrix factorization (PMF) model in which the model capacity is controlled automatically by integrating all model parameters and hyperparameters.
- SVD++ [27]: Singular value decomposition plus plus is a combined model that sums the predictions of latent factor models and neighborhood models.
- BPRMF [8]: Bayesian personalized ranking matrix factorization is a method that was initially proposed to model implicit feedback, and we introduced it in Section 2.2.
- LCR [28]: Local collaborative ranking is a model that assumes that the user preference matrix is locally low-rank, where the locality is defined by a neighborhood with respect to a given metric on pairs of (row, column) indices.
- APPL [17]: Alternating pointwise-pairwise learning is a joint method that combines implicit feedback and explicit feedback in the loss function.



(a) Effect of parameters on ML-100K



(b) Effect of parameters on R4

Fig. 2. Effect of parameters on SPR.

- CoFiSet [29]: Collaborative filtering by learning pairwise preferences over item sets is a new and relaxed assumption of pairwise preferences over item sets, which defines a user's preference on a set of items (item set) instead of on a single item only.
- RBPR [18]: Rating Bayesian personalized ranking is a CF ranking model that combines a rating-oriented approach of probabilistic matrix factorization and a pairwise ranking-oriented approach of Bayesian personalized ranking (BPR).

4.4. Parameter analysis

This subsection starts with a discussion of the effect of the hyperparameter on the performance of SPRMF. Then we discuss the impact of the sampling ratio and latent dimension.

4.4.1. Effect of the hyperparameter

The value of the hyperparameter α is important for the performance on the recommendation task. Hence, we study the effect of α by varying its value. The effect of α on the performance of the model is shown in Fig. 2. The number of iterations t is set to 20, and the latent feature dimension is set to 10. We use a Gaussian distribution $\mathcal{N}(0, 0.1)$ to randomly generate all of the parameters. We explored learning rates of {0.001, 0.002, 0.005, 0.007, 0.01 }, and regularization coefficients of {0.01, 0.02, 0.05, 0.07, 0.1}. Finally, we found that the model has the best performance when the learning rate is fixed to $lr = 0.007$ and regularization parameter $\lambda = 0.05$. During the experiment in Fig. 2, α ranges from 0.01 to 0.30, and the step size is set to 0.01. The step size is set to 0.05 when α ranges from 0.30 to 1. We truncated the data, and the results below a certain value were discarded.

The performance of SPRMF converges (for NDCG@5 and NDCG10) when α approaches 0.17 and 0.45 on ML-100K and R4, respectively. Since the datasets have different data distributions, the hyperparameters must be adjusted to account for the effect of similarity constraints. The higher the average proportion of popular items, the higher the optimal hyperparameter value.

The experimental results show that hyperparameters can significantly affect the performance of SPRMF. We also find that the penalty proposed in this article based on similar item pairs is effective. The significant change in performance on NDCG means that the penalty can help the recommender system rank the items that users are interested in first.

4.4.2. The impact of the sampling ratio

We conducted extensive experiments to show how the sampling ratio influences the model performance. Fig. 3 shows the NDCG obtained by SPRMF w.r.t the number of negative samples per positive instance. As seen, sampling more than two negative instances is helpful. Different data distributions place different requirements on sampling ratios. On ML-100K and ML-LS, the best NDCG@5 and NDCG@10 are obtained when the sampling ratio is set to 2. On ML-10M and R4, the best NDCG@10 is obtained when the sampling ratio is set to 4. On ML-1M and R3, the performance rapidly increases when the sampling ratio increases from 1 to 4. Then, as the sampling ratio increases from 4 to 10, the performance slowly increases. The impact of the sampling ratio on the performance is related to the degree of data imbalance. The higher the proportion of popular items, the higher the sampling ratio needed to achieve the best performance.

Overall, the optimal sampling ratio is approximately 2 to 4. Sampling more negative instances not only requires more time to train the model, but it also sometimes degrades the performance.

4.4.3. The impact of the latent dimension

Fig. 4 shows the performance of SPR with different latent dimensions d. On the whole, the impact of the embedding dimension on the model performance is not as great as that of the sampling ratio. The best NDCG@5 and NDCG@10 are obtained when the latent dimension is set to 40 on ML-100K. The curves on ML-100K and R4 are similar because they both have a vertex, and the performance starts to rise slowly after this vertex. The average number of user interactions is greater than the average number of item interactions. Thus, SPR can mine the relationship between the personalized needs of the users when the user behaviors are rich, and it can promote the recommender system to achieve better performance. On other datasets, the performance basically converges when setting the embedding dimension to 10.

Overall, the optimal latent dimension is between 10 and 40. Moreover, a larger latent dimension d means higher complexity.

4.5. Sampler methods comparison

We verify the time performance and NDCG performance of the sample method used in this article. There are two methods that we have compared:

- random: This method uniformly samples items that the user has interacted [16,18].
- APPLE+: We follow the APPLE method [14] and set the weight of the item as $p(q|i) = exp(r_{uq} - r_{ui})/(1 + exp(r_{uq} - r_{ui}))^2$. Here we sample similar item pairs while the APPL method samples negative samples.

We used an Intel Xeon E5-2620 CPU clocked at 2.4 GHz with 64 GB of RAM. Table 3 shows the running time in seconds of different sampling methods. We can see that the random method is the fastest of those methods. Our method is a variant of the uniform random sampling method and has the same complexity as the uniform random sampling method. The additional time overhead of our method comes from group items by ratings. APPLE+ is more complicated than other methods. Table 4 shows the NDCG performances. Our method is better than the other methods. Especially in Ml-100K, ML-10M, R3, and R4, our method

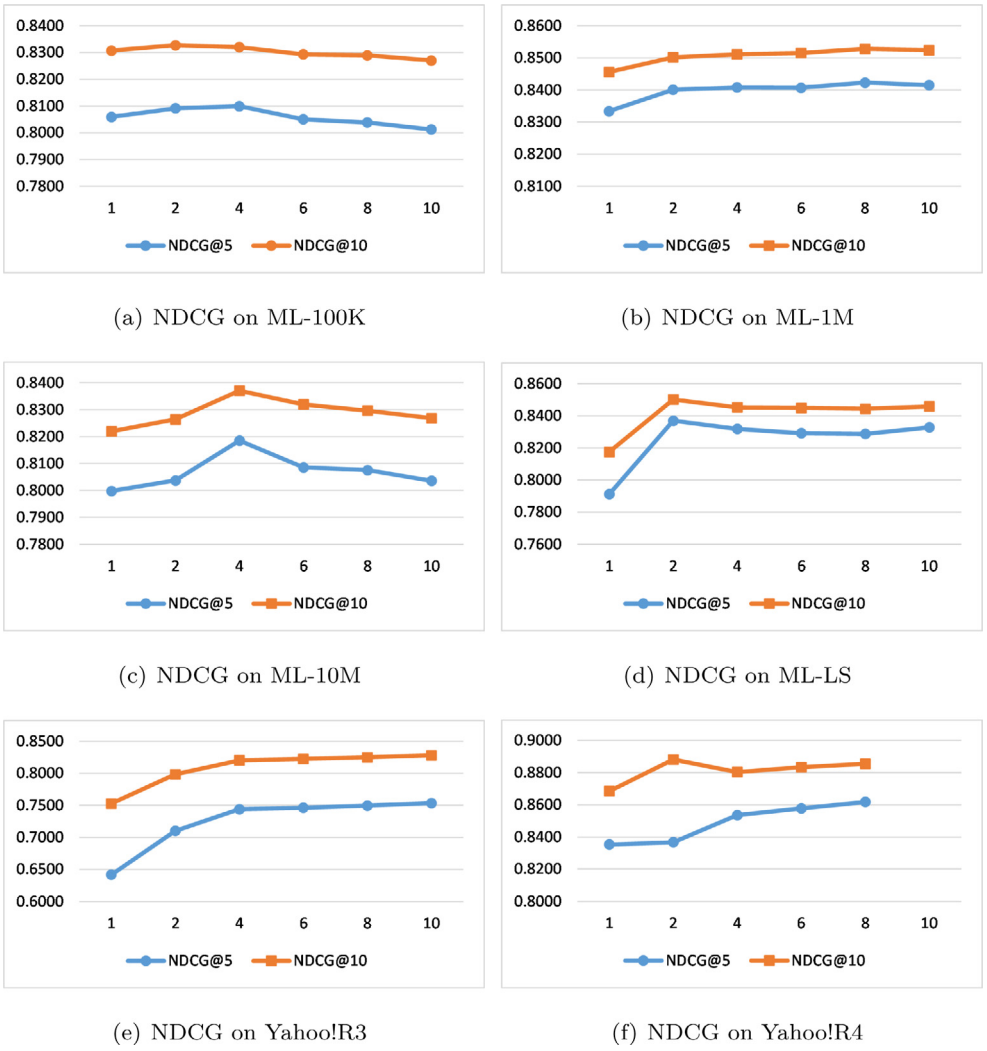


Fig. 3. SPRMF Performance w.r.t different sample ratios of training data on six datasets.

Table 3
Running time in seconds of different sampling methods.

Method	Ml-100K	ML-1M	ML-10M	ML-LS	R3	R34
random	4.00	47.96	12459.56	4.07	23.30	10.41
APPLE+	19.06	316.15	15614.47	43.38	34.80	29.76
our	6.14	58.01	12856.97	6.56	42.95	18.18

outperforms APPLE+ by 2% on average. APPLE+ divides items into several groups by $r_{ui} - r_{uq}$. The probability of an item depends on the rating. The higher the rating is, the greater the probability that an item will appear. This approach makes all items similar to high-rating items. However, the similarity between some low-rating and high-rating items is inappropriate, which also causes the method to perform poorly on some datasets. Overall, our method achieves the best NDCG performance with a slightly higher time overhead than the random sampling method.

4.6. Model performance comparison

For the baseline method, we cite the best results in existing work or in open source frameworks [17,30]. For SPRMF, we use Optuna and the discrete uniform search method to search for the best result. The recommendation performance of SPRMF and other baselines are shown in Table 5. Our model achieved the best performance on all six datasets. In particular, SPRMF outperforms BRPMF by 14.67%, 10.74%, 8.99%, 7.07%, 15.64%, and 3.35% on ML100K, ML-1M, ML-10M, ML-LS, R3, and R4, respectively. SPRMF outperforms CoFiSet by 3.97%, 0.86%, 1.69%, 0.33%, 1.85%, and 2.11% on the datasets. Upon reviewing some statistical information on these datasets, we found a phenomenon where a small number of items have a large number of ratings, and their proportion is extremely high. The more obvious this phenomenon is, the better the performance of SPR. The obtained NDCG@10 is similar to the obtained NDCG@5 on all six datasets.

Note that the improvements of SPRMF on ML-1M and ML-LS are not obvious. By reviewing Fig. 1, we can determine that the distributions of these two datasets are different from those of the other four datasets. In ML-1M and ML-LS, the number of average ratings for items is small, and they lack more information about forming similar pairs of items, which is a major challenge for our model. Because SPR associates two items through similarity, a sparse item relationship makes two items associated with SPR not necessarily appropriate. SPR penalizes similar items i and q, which constitute a similar item pair. When the number of global scores of i is insufficient, the sampling method cannot work well on other datasets.

Overall, SPRMF is optimized for data distributions with long tails, and it can achieve better results than existing models in imbalanced data distributions. Moreover, SPR encourages personalized items to have the same predicted scores as popular

Fig. 4. SPRMF Performance w.r.t different latent dimension of training data on six datasets.

Table 4
Comparison of NDCG obtained on different datasets of different sampling methods.

	Method	Ml-100K	ML-1M	ML-10M	ML-LS	R3	R4
NDCG@5	random	0.8001	0.8397	0.8078	0.8337	0.7407	0.8516
	APPLE+	0.7858	0.8399	0.7974	0.8252	0.7457	0.8627
	our	0.8192	0.8483	0.8185	0.8387	0.7533	0.8702
NDCG@10	random	0.8250	0.8508	0.8327	0.8480	0.8185	0.8789
	APPLE	0.8186	0.8505	0.8218	0.8426	0.8221	0.8861
	our	0.8327	0.8550	0.8332	0.8508	0.8280	0.8917

items, which allows better-personalized recommendations to be provided when faced with data with long-tailed distributions.

4.7. Time complexity analysis

In terms of time complexity, both the original BPR and SPR have $\mathcal{O}(N)$ complexity, where N is the sample pair obtained by sampling. However, we can determine by comparing the objective functions that SPR has two terms while BPR has only one term. Therefore, SPR has a more significant coefficient than BPR. When compared with other methods, such as APPL and RBPR, the coefficients are the same. To further verify the proposed model's time efficiency, we compare its running time with some representative baselines. We record the running time across the models in the same computing environment, and we set the hyperparameters according to the original work. We used an Intel Xeon E5-2620 CPU clocked at 2.4 GHz with 64 GB of RAM.

Table 6 shows the training and prediction time for each model. The training time is the time for training one epoch of the data, and the prediction time is the time required to complete the prediction for the whole testing set. We can observe that the training time of BPR is the largest. The reason is that BPR is implemented in Python,[3] and our core computing code is implemented in C++. Languages that are biased toward the bottom are complicated to implement, but they have good time efficiency. In the test stage, the performance of the two is similar because both use the dot product between the calculated vectors.

[3] https://github.com/gamboviol/bpr

Table 5
Comparison of NDCG obtained on different datasets.

		ML-100K	ML-1M	ML-10M	ML-LS	R3	R4
NDCG@5	Prod2Vec	0.6321	0.6521	0.6369	0.6599	0.6417	0.8181
	BPMF	0.7272	0.7850	0.7021	0.7403	0.6512	0.8551
	SVD++	0.7214	0.7771	0.7184	0.7454	0.6612	0.8454
	LCR	0.7161	0.7783	0.6933	0.7340	0.6381	0.8510
	BPRMF	0.7144	0.7660	0.7510	0.7833	0.6514	0.8420
	APPL	0.7289	0.7882	0.7355	0.7971	0.6695	0.8607
	CoFiSet	0.7879	0.8411	0.8049	0.8359	0.7396	0.8522
	RBRP	0.7832	0.8389	0.8026	0.8274	0.7386	0.8493
	SPRMF	**0.8192**	**0.8483**	**0.8185**	**0.8387**	**0.7533**	**0.8702**
NDCG@10	Prod2Vec	0.6910	0.6907	0.6957	0.7140	0.6520	0.8727
	BPMF	0.7596	0.7948	0.7705	0.7733	0.7466	0.8816
	SVD++	0.7553	0.7880	0.7597	0.7689	0.7550	0.8727
	LCR	0.7532	0.7895	0.7323	0.7783	0.7277	0.8753
	BPRMF	0.7493	0.7797	0.7820	0.8065	0.7539	0.8718
	APPL	0.7610	0.7970	0.7733	0.8211	0.7603	0.8838
	CoFiSet	0.8157	0.8516	0.8216	0.8469	0.8173	0.8796
	RBRP	0.8122	0.8490	0.8231	0.8412	0.8157	0.8774
	SPRMF	**0.8327**	**0.8550**	**0.8332**	**0.8508**	**0.8280**	**0.8917**

Table 6
Time complexity analysis.

	ML-100K		ML-1M		ML-10M		ML-LS		Yahoo!R3		Yahoo!R4	
	train	test	train	test	train	test	train	test	train	test	train	test
BPR	10.19	8.09	151.1	73.57	1623.81	611.95	13.71	6.21	32.76	23.1	17.78	13.00
APPL	0.25	7.93	2.15	74.98	21.55	635.25	0.39	6.28	0.55	23.17	0.35	13.88
RBPR	1.86	8.96	3.80	144.23	23.30	2,795.12	3.12	9.46	3.08	47.34	3.49	19.04
SPR	0.20	7.66	1.95	73.22	20.40	613.72	0.38	5.71	0.50	23.19	0.35	14.00

5. Related work

In this section, recent pairwise methods are first discussed. To overcome the impact of imbalanced data distributions, we then illustrate some samplers. Finally, we summarize the difference between SPR and existing methods.

5.1. Pairwise method

The pairwise method, which is a type of recommender method, has a deep research history and is widely used in online services. The pairwise method directly learns the partial order of items, and it can produce more personalized recommendation results than other methods. There are also two types of pairwise methods: the pairwise method that uses implicit feedback, and the pairwise method that adds rating information.

5.1.1. Pairwise method using implicit feedback

Implicit feedback information mainly contains user interaction information, i.e., whether the user has interacted with an item. Different from the classification problem, there are multiple possible reasons why users in a recommendation do not interact with items. It could be that the user does not like the item or the user does not see the item. In other words, "unknown data" cannot be considered to be real negative samples [19]. The pairwise method assumes that a user is more likely to prefer an interacting item to a non-interacting item. Therefore, the pairwise method models the recommendation problem as a ranking problem, and it hopes to learn from the local ranking information and the global ranking information.

As the first pairwise method, Bayesian personalized ranking (BPR) is widely used in personalized recommender systems [11, 12]. It has been empirically shown that BPR is a remarkable achievement with implicit feedback owing to its ranking-oriented pairwise assumption. Some follow-up studies have extended BPR [9–12]. The pairwise method itself has also undergone much development. The GBPR model introduces group preferences to relax the individual and independence assumptions [16]. GBPR relaxes independent users into user groups through a collaborative filtering idea; in other words, a group of users prefer item i to item j, which leads to cooperation among users. At the same time, the UGPMF model uses the existing user demographic information to construct similarity graphs at the user end, and it then uses those graphs to standardize the pairwise matrix decomposition process based on BPR [31]. We note here that the user demographics include age, gender, and occupation. CoFiSet [29] defines a user's preference on a set of items (item-set) instead of on a single item only. It is a new and relaxed assumption of pairwise preferences over item sets. The relaxed assumption can give more accurate pairwise preference relationships.

5.1.2. Pairwise method adding rating information

Explicit feedback mainly refers to user ratings on items they purchased or viewed in the past. There is an overlapping subspace between explicit feedback and implicit feedback in most models [17]. Thus, it is beneficial for the impacts of the different feedbacks to be balanced.

The APPL model uses pointwise rating feedback and implicit feedback alternately, and its performance is significantly better than that of previous work [17]. The predicted scores can be fitted into pairwise information and ratings at the same time. The RPR-NMF model uses the relative pairwise relationship as a constraint to make the learned low-dimensional representation [32]. The relative pairwise relationship extends the partial order between two items to between three items. CMR [33] satisfies three different constraints: (1) the rowwise order constraint, i.e., the order of any pair of rating scores a user gives to two items should be preserved; (2) the columnwise order constraint, i.e., the order of any pair of rating scores an item received from two users should be estimated correctly; and (3) the pointwise prediction constraint, i.e., the prediction of a rating score should be close to its real value. The PrefPMFSI model uses probabilistic matrix factorization to generate an efficient ranking of items [34]. The user- and item-side information are integrated into the model

using the matrix cofactorization technique. Meanwhile, the RBPR model combines a rating-oriented approach of probabilistic matrix factorization (PMF) and a pairwise ranking-oriented approach of Bayesian personalized ranking (BPR) [18]. Therefore, RBPR makes full use of ratings and implicit feedback data.

5.2. Sampler for pairwise method

In many real-world learning systems, the data matrix can be highly dimensional but sparse [35,36]. This circumstance poses an imbalanced learning problem since the scale of missing entries is usually much larger than that of the observed entries, and these missing entries cannot be ignored due to the valuable negative signal. However, it is widely known that the performance of BPR depends largely on the quality of the negative samples.

To overcome this problem, a nonuniform item sampler is proposed [20]. The proposed sampler is context-dependent (for each user), and it oversamples informative pairs to speed up convergence. The oversampling process that relies on the predicted score can help the model to correctly distinguish between positive and negative samples. The APPLE model improved BPR by defining an adaptive objective function and gradient [14]. Then, APPLE defined the utility of a randomly sampled triple to strengthen the training results of the small deviation triples in training. The WalkRanker model constructs a user–item bipartite graph to represent the relationships between user–user, user–item, and item–item [15]. It first uses the random walk method to extract positive samples from short random walk sequences dynamically. Then, a rank-aware negative sampling method is used to extract negative samples. Finally, WalkRanker applies the BPR optimization criterion to learn based on the MRR loss function. A study [37] developed an efficient optimization method that includes all of the missing entries as negative. Ding (2018) proposed that the performance of BPR does not decrease but increases after reducing the sampling space [21]. Thus, BPR+view samples an item pair (i, j) from three candidate sets: purchased items, viewed (but not purchased) items, and remaining items. The best probabilities are [0.3, 0.3, 0.4].

5.3. Summary

SPR, which is proposed in this paper, is a new pairwise method that narrows the score between similar item pairs in such a way that the impact of imbalanced datasets is avoided.

The difference between recent pairwise methods (such as GBPR and CoFiSet) and SPR is that SPR adds a constraint that requires similar items to have similar scores. In contrast, GBPR divides the positive samples into different groups g, and the score of the group is equal to the average score of all of the positive samples in the group $\hat{r}_{gi} = \sum_{w \in g} \hat{r}_{wi}$. GBPR uses a fused preference $\hat{r}_{gui} = \hat{r}_{gi} + (1 - \rho)\hat{r}_{ui}$ instead of the original predicted score. Finally, the objective function mainly optimizes the difference between the mixture of positive samples and negative samples $\hat{r}_{gui} - \hat{r}_{uj}$.

GBPR and SPR have different performances on the positive samples. GBPR uses a variety of groups, while SPR penalizes the difference between the two samples within the group. During training, GBPR always applies a positive gradient to all of the items in the group, while SPR applies a negative gradient to items with higher scores. SPR can enhance the similarity of items in a group and the differences between groups. Meanwhile, CoFiSet is a method that uses logisticMF to predict the occurrence of interaction. The difference between CoFiSet and SPR is similar to that between GBPR and SPR.

At the same time, APPL [17] and RBPR [18] require that the predicted score be as close to the real score as possible while maintaining the relative relationship. In other words, $(r_{ui} - \hat{p}_{ui})^2 + (r_{uj} - \hat{p}_{uj})^2 - (r_{ui} - r_{uj})\hat{p}_{uij}$. Moreover, negative samples have two gradients from $(r_{uj} - \hat{p}_{uj})^2$ and $-(r_{ui} - r_{uj})\hat{p}_{uij}$. However, in SPR, the scoring information is used to classify items and sample similar items, and there could be a difference between the predicted score and the actual score. Furthermore, SPR relaxes the relationship between negative samples.

6. Conclusions

In this paper, we focus on improving the effectiveness of BPR when the real data distribution is imbalanced. First, we propose a novel personalized recommendation method called similarity pairwise ranking (SPR), which considers the similarity between two positive samples in terms of user preferences. SPR eliminates the differences in the scores between popular and personalized items through the similarity between items, and overcomes the impact of imbalanced datasets. Through SPR, other recommendation models can generate a more personalized recommendation to meet the individual needs of users. We use SPRMF to prove the effectiveness of the SPR method. The experimental results on six real datasets show that the proposed SPR method can increase NDCG. Moreover, SPR improves the performance of personalized recommendations, and it demonstrates superior recommendation quality over recent state-of-the-art methods.

In this paper, similarity comes from user ratings. However, ratings are not always available. In the real environment, there are more click events. Sampling good similar item pairs is a significant problem. There is other information, such as text-rich and mate-data, that can be used to obtain similarity. In the future, we plan to further characterize and explore the relationship between popular items and personalized items. In addition, we will apply SPR to other methods, including deep learning and graph networks.

CRediT authorship contribution statement

Junrui Liu: Conceptualization, Methodology, Software, Writing – original draft. **Zhen Yang:** Methodology, Supervision, Formal analysis, Investigation. **Tong Li:** Conceptualization, Methodology, Supervision, Validation, Writing – review & editing. **Di Wu:** Writing – review & editing. **Ruiyi Wang:** Visualization, Writing – review & editing.

Declaration of competing interest

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

Acknowledgments

This work is partially supported by Beijing Natural Science Foundation Project, China (No. Z200002), the Project of Beijing Municipal Education Commission, China (No. KM202110005025), and Engineering Research Center of Intelligent Perception and Autonomous Control, Ministry of Education.

Appendix A. Supplementary materials

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.knosys.2021.107828.

References

[1] C.A. Gomez-Uribe, N. Hunt, The netflix recommender system: Algorithms, business value, and innovation, ACM Trans. Manag. Inform. Syst. (TMIS) 6 (4) (2015) 1–19.

[2] G. Zhou, X. Zhu, C. Song, Y. Fan, H. Zhu, X. Ma, Y. Yan, J. Jin, H. Li, K. Gai, Deep interest network for click-through rate prediction, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018, pp. 1059–1068.

[3] C. Li, Z. Liu, M. Wu, Y. Xu, H. Zhao, P. Huang, G. Kang, Q. Chen, W. Li, D.L. Lee, Multi-interest network with dynamic routing for recommendation at Tmall, in: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, 2019, pp. 2615–2623.

[4] M. Grbovic, V. Radosavljevic, N. Djuric, N. Bhamidipati, J. Savla, V. Bhagwan, D. Sharp, E-commerce in your inbox: Product recommendations at scale, in: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2015, pp. 1809–1818.

[5] C. Wang, D.M. Blei, Collaborative topic modeling for recommending scientific articles, in: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2011, pp. 448–456.

[6] G. Linden, B. Smith, J. York, Amazon. com recommendations: Item-to-item collaborative filtering, IEEE Internet Comput. 7 (1) (2003) 76–80.

[7] J. Liu, W. Pan, Z. Ming, CoFiGAN: Collaborative filtering by generative and discriminative training for one-class recommendation, Knowl.-Based Syst. 191 (2020) 105255.

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

[9] S. Zhao, I. King, M.R. Lyu, Geo-pairwise ranking matrix factorization model for point-of-interest recommendation, in: International Conference on Neural Information Processing, Springer, 2017, pp. 368–377.

[10] R. He, J. McAuley, VBPR: visual bayesian personalized ranking from implicit feedback, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 30, no. 1, 2016.

[11] H. Zhang, J. McAuley, Stacked mixed-order graph convolutional networks for collaborative filtering, in: Proceedings of the 2020 SIAM International Conference on Data Mining, SIAM, 2020, pp. 73–81.

[12] Z. Liu, M. Wan, S. Guo, K. Achan, P.S. Yu, Basconv: Aggregating heterogeneous interactions for basket recommendation with graph convolutional neural network, in: Proceedings of the 2020 SIAM International Conference on Data Mining, SIAM, 2020, pp. 64–72.

[13] 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.

[14] H. Zhong, W. Pan, C. Xu, Z. Yin, Z. Ming, Adaptive pairwise preference learning for collaborative recommendation with implicit feedbacks, in: Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, 2014, pp. 1999–2002.

[15] L. Yu, C. Zhang, S. Pei, G. Sun, X. Zhang, Walkranker: A unified pairwise ranking model with multiple relations for item recommendation, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 32, no. 1, 2018.

[16] W. Pan, L. Chen, Gbpr: Group preference based bayesian personalized ranking for one-class collaborative filtering, in: Twenty-Third International Joint Conference on Artificial Intelligence, 2013.

[17] Y. Lei, W. Li, Z. Lu, M. Zhao, Alternating pointwise-pairwise learning for personalized item ranking, in: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, 2017, pp. 2155–2158.

[18] J. Feng, Z. Xia, X. Feng, J. Peng, Rbpr: A hybrid model for the new user cold start problem in recommender systems, Knowl.-Based Syst. 214 (2021) 106732.

[19] Y. Hu, Y. Koren, C. Volinsky, Collaborative filtering for implicit feedback datasets, in: 2008 Eighth IEEE International Conference on Data Mining, IEEE, 2008, pp. 263–272.

[20] S. Rendle, Factorization machines, in: 2010 IEEE International Conference on Data Mining, IEEE, 2010, pp. 995–1000.

[21] J. Ding, F. Feng, X. He, G. Yu, Y. Li, D. Jin, An improved sampler for bayesian personalized ranking by leveraging view data, in: Companion Proceedings of the the Web Conference 2018, 2018, pp. 13–14.

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

[23] D. Cai, X. He, J. Han, T.S. Huang, Graph regularized nonnegative matrix factorization for data representation, IEEE Trans. Pattern Anal. Mach. Intell. 33 (8) (2010) 1548–1560.

[24] J. Tang, H. Gao, X. Hu, H. Liu, Exploiting homophily effect for trust prediction, in: Proceedings of the Sixth ACM International Conference on Web Search and Data Mining, 2013, pp. 53–62.

[25] K. Verbert, N. Manouselis, X. Ochoa, M. Wolpers, H. Drachsler, I. Bosnic, E. Duval, Context-aware recommender systems for learning: a survey and future challenges, IEEE Trans. Learn. Technol. 5 (4) (2012) 318–335.

[26] R. Salakhutdinov, A. Mnih, Bayesian probabilistic matrix factorization using Markov chain Monte Carlo, in: Proceedings of the 25th International Conference on Machine Learning, 2008, pp. 880–887.

[27] 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.

[28] J. Lee, S. Bengio, S. Kim, G. Lebanon, Y. Singer, Local collaborative ranking, in: Proceedings of the 23rd International Conference on World Wide Web, 2014, pp. 85–96.

[29] W. Pan, L. Chen, Z. Ming, Personalized recommendation with implicit feedback via learning pairwise preferences over item-sets, Knowl. Inf. Syst. 58 (2) (2019) 295–318.

[30] G. Guo, J. Zhang, Z. Sun, N. Yorke-Smith, LibRec: A Java Library for Recommender Systems, in: UMAP Workshops, Vol. 4.

[31] L. Du, X. Li, Y.-D. Shen, User graph regularized pairwise matrix factorization for item recommendation, in: International Conference on Advanced Data Mining and Applications, Springer, 2011, pp. 372–385.

[32] S. Jiang, K. Li, R.Y. Da Xu, Relative pairwise relationship constrained non-negative matrix factorisation, IEEE Trans. Knowl. Data Eng. 31 (8) (2018) 1595–1609.

[33] J. Hu, P. Li, Collaborative multi-objective ranking, in: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, 2018, pp. 1363–1372.

[34] A. Pujahari, D.S. Sisodia, Pair-wise preference relation based probabilistic matrix factorization for collaborative filtering in recommender system, Knowl.-Based Syst. (2020) 105798.

[35] X. He, J. Tang, X. Du, R. Hong, T. Ren, T.-S. Chua, Fast matrix factorization with nonuniform weights on missing data, IEEE Trans. Neural Netw. Learn. Syst. 31 (8) (2019) 2791–2804.

[36] R. Zhang, S. Niu, Y. Li, Robust sequence embedding for recommendation, in: International Conference on Knowledge Science, Engineering and Management, Springer, 2020, pp. 114–122.

[37] H.-F. Yu, M. Bilenko, C.-J. Lin, Selection of negative samples for one-class matrix factorization, in: Proceedings of the 2017 SIAM International Conference on Data Mining, SIAM, 2017, pp. 363–371.