SN Computer Science (2022) 3:309
https://doi.org/10.1007/s42979-022-01187-w

ORIGINAL RESEARCH

Application of MLP Based on Joint Similar Groups in User Interest Expression and Recommendation Service

Zhenyu Li[1] · Shuqing Li[1] · Yunhan Liu[1] · Zhan Bu[1]

Received: 2 February 2022 / Accepted: 2 May 2022 / Published online: 27 May 2022

Abstract

Latent factor model is better at capturing global information, but not local information. Therefore, many models combine the nearest neighbor information in the latent factor model to improve the performance in recommender system. However, the current fusion models pay no attention to the relationship between the accuracy of local information and model performance. In addition, the expression of local information is different between explicit neighbors and implicit neighbors, so that explicit feedback and implicit feedback have different values. Current fusion models usually utilize only one kind of feedback to capture the local information, which leads to the insufficiency and inaccuracy for using local information. These exited methods do not effectively learn the deep correlation between the similar users and similar items which all can represent the local information. Utilizing the recommender system of deep learning, we propose a fusion MLP model based on Joint Similar Groups, which utilizes both explicit feedback and implicit feedback in local information and learn the deep correlation between similar users and similar items. Experiments on two datasets show that our modes outperform state-of-art algorithms in explicit recommendation task.

Keywords Local information · Latent factor · MLP · User profile · Recommender system

Introduction

In the era of rapid growth of data, recommender system has been proved to be a valuable way for users to deal with information overload, and has become one of the most popular tools in Internet applications [1], including e-commerce, social networking, and so on. Personalized recommendation service is specially an important strategy to improve users' experience and recommendation efficiency, which can generally be divided into three categories: (1) Recommender system based on Collaborative Filtering (CF) that utilizes users and items interaction information; (2) Content-Based (CB) recommender system that utilizes more feature properties of users and items; (3) Hybrid recommender system combining the CF and CB [2]. Among them, CF includes lots of classical models. For example, traditional methods include k-neighbor-based CF and latent factor model. The former calculates the similarity through the interaction of users and items, and produces recommendations by screening the groups with high similarity [3], while most of the groups with low similarity are not taken into account. The latter maps users and items to a shared latent space and generates recommendations through the correlation of latent features.

Mining the correlation information between users and items, which can be divided into local information and global information, is the most important in the recommender system. First, there are various types of local information in the recommender system, which represents the strong correlation among users and items such as users or items with high similarity [4]. In algorithms for mining local information, the typical one is the k-neighbor-based CF, which only utilizes a small set of users or items with high behavioral similarity while most of the users or items with low similarity are not taken into account. That is to say, this method does not make use of more global information to capture the weak but useful correlation. Meanwhile, global correlation is a latent factor model that utilizes all ratings to learn the overall correlation among users and items. However, most latent factor models are difficult to

✉ Shuqing Li
leeshuqing@gmail.com

✉ Yunhan Liu
61096003@qq.com

[1] College of Information Engineering, Nanjing University of Finance and Economics, Nanjing 210023, China

capture local information. In fact, the traditional latent factor model and the neighborhood-based collaborative filtering are different strategies to mine these two kinds of information, respectively, so that they are usually only capable of capturing one of these information.

In recent years, deep learning has been gradually applied in many research fields, including computer vision, natural language processing, audio recognition, and so on. In the field of recommender system, deep learning has also achieved great improvement [5]. Among them, CF based on deep learning also has a wide range of applications, which are mainly the extension of latent factor model. In addition, due to the excellent fitting ability of neural network, its performance is usually better than the traditional Matrix Factorization (MF) model. However, like traditional MF, it is also insensitive to local information.

Because of this limitation, many researches propose fusion models to improve performance by adding local information to latent factor models. However, existed researches on the value and accuracy of local information in fusion model is insufficient. In the selection of local information, these kind of fusion models usually only utilize one kind of feedback to generate neighbors. For example, they often focus on the implicit interaction neighbors, and pay little attention to the value of explicit feedback. In fact, explicit and implicit feedback have different values when expressing user preferences. Specifically, the explicit feedback indicates exactly how interested the user is in the item, and implicit feedback indicate indirectly whether the user is interested in the item. However, most of models combining explicit feedback and implicit feedback are usually in the global perspective. In the fusion model, the value of local information can significantly affect the performance of the fusion model, and using explicit and implicit feedback in local perspective is also worth further exploring. Therefore, we analyse and compare the differences between the explicit and implicit neighbors of users and items from the local perspective, and explore the effective utilization of more accurate and valuable local information combining with the CF based on deep learning methods. Meanwhile, in the fusion model of global and local information, we utilize two kinds of relationships to find their deep correlations, in which relationship of user-similar items can represent the similarity between the user and the item' neighbors and relationship of item-similar users can reflect similarity between the item and the user' neighbors, respectively.

Related Work

CF Based on Deep Learning

In CF based on deep learning, Restricted Boltzmann Machine (RBM) is an early well-known model [6]. Then, auto-encoders and its variants have been applied in CF [7], such as Probabilistic Rating Auto-encoder [8] and Deep Collaborative Filtering via Marginalized Denoising Auto-encoder [9]. Deep MF (DMF) [5] and Convolutional MF (CMF) [10] are MF models based on deep neural networks. These models belong to representation-based learning models, which utilize inner product or cosine similarity of latent vectors to reconstruct all ratings [11], and combine the representation ability of deep learning with CF.

Another type is using neural network to learn the matching function between user and item [12]. One of the most famous is Neural network-based CF (NCF) [13], which is a CF method based on Multi-Layer Perceptron (MLP). It can match and evaluate the relevance of users and items through two-way modeling. This framework completely replaces the inner product modeling of MF, and the recommendation performance is more accurate. Then, Joint Neural CF (JNCF) [14] and Outer product NCF (ONCF) [15] enhance the performance of MLP model from different perspectives.

Our model is also a CF model based on matching function, which models the correlation between users and items based on MLP. In addition, to improve the accuracy of description of users and items, our model makes use of more local information contained in similar users and similar items.

Local Information in Latent Factor Model

Latent factor model has weak ability to capture local information. Therefore, many fusion models try to improve the effect by integrating local information. The SVD++ improves the traditional SVD, takes the interaction neighbors as the local information, and combines the MF with the neighbor-based CF, so that the accuracy of the rating prediction was improved. In the deep recommender system, Neighbor-based NCF (NNCF) [16] utilizes the implicit interaction network of users and items to construct neighbors, and adds it to NCF model as local information. Collaborative Memory Network (CMN) utilizes memory components to search similar users, and give higher weight to the most similar users by neural attention mechanism [17]. Since the user similarity can be generated by implicit interaction, CMN actually integrates implicit interaction neighbors into the model as local information. Collaborative deep recommendation with Global and Local Item Correlation (GLICR) [18] tightly couples deep neural network and MF, and introduces Manifold Regularization for the first time to learn global and local information simultaneously from auxiliary data of items, which is a model that uses side information to learn global and local item information. In addition, Global and Local SLIM (GLSLIM) [19] utilizes users'

implicit behavior to cluster, and trains multiple different local models to learn local information, so as to effectively improve the performance.

In addition to using auxiliary data, most models only use the implicit interaction between users and items to capture local information by calculating the similarity. However, they has not conducted adequate researches on the relationship between accuracy of local information and fusion models.

In fact, the interaction between users and items can be grouped as implicit feedback and explicit feedback according to rating behavior and specific rating value [12]. Specifically, the rating score indicates exactly how interested the user is in the item, and rating behavior indicate whether the user is interested in the item. Moreover, existing recommender systems have proved that combining these two kinds of information in the latent factor model can improve the performance. SVD++ [4] decomposes the explicit feedback matrix and adds the calculation of implicit vector to the prediction function, so as to improve the performance. Deep MF(DMF) significantly improves the performance in top-N recommendation by constructing a matrix with explicit rating and non-preference implicit rating, and using the latent factor model of deep structure [5]. Deep Feedback Network (DFN) utilizes a variety of explicit and implicit positive and negative feedback information in multi-layer neural network to learn the unbiased preference of users [20].

However, both MF and deep learning models are weak in the combination of explicit feedback and implicit feedback in latent factor model since they are mostly carried out from a global perspective. We find that the accuracy of local information can significantly enhance the performance, and the use of single feedback cannot sufficiently and accurately reflect the local information. These models mentioned above only focus on implicit feedback to select neighbors, which only represent the implicit local correlation and contain insufficient information. Therefore, different from the current model, on the basis of learning the global information of users and items, our model integrates the local information and innovatively integrates the explicit and implicit feedback in the local perspective.

Moreover, since users' preference of similar items are likely to be similar and the same is true for item, previous studies have not focused on the deep correlation between users-similar items and item-similar users. Therefore, in the fusion of local information and global information, we not only use MLP to model user-item, but also model user-similar items and item-similar users, respectively.

In a word, deep collaborative filtering is difficult to capture local correlation because it belongs to latent factor model, which is the common limitation of these models, and the existing models lack the research on the deep correlation of similar users and similar items. To solve this problem, we use explicit and implicit feedback to fuse global and local relationship in the model proposed below.

Contribution

Notations and Descriptions

The mathematical notations used in this paper are summarized in Table 1.

Main Contribution

Based on the above two points, we propose a MLP model based on Joint Similar Groups (MLP-JSG), where the similar group contains similar users and similar items. The theoretical basis is as follows: (1) Learning the complex nonlinear interaction of users and item latent vectors through MLP is more effective than only learning linear interaction through inner product in MF; (2) The explicit and implicit feedback of users and items are different in expressing preferences, so that their neighbors also contain different local information; (3) The latent features of users and items with similar behaviors are similar, so that the deep correlation between similar users and similar items by MLP can be mined; (4) Fusion of local information can effectively improve the performance of latent factor model.

The main contributions in this paper are illustrated as follows:

1. To capture global information and local information more effectively, we construct three MLPs to model the correlation including user-similar items, item-similar users and user-item, and then improve the recommendation effect by fusing their high-order interaction vectors. In the fusion of higher-order vectors, the element prod-

Table 1 Notations and descriptions

Symbol	Notation and description
$\boldsymbol{P_u}$,$\boldsymbol{q_i}$	User and item latent vectors in MLP
U	Explicit rating matrix
I	Implicit rating matrix
r_{ui}	An explicit user rating of an item
b_{ui}	An implicit user rating of an item
sim$U$	Similar user groups
sim$I$	Similar item groups
W	Weight matrix of fully connected layer
x	One-hot vector of users and items
g	Activation function
b	Bias in the MLP layer
$\boldsymbol{a_n}$,$\boldsymbol{b_n}$,$\boldsymbol{c_n}$	Vector in our proposed model
$\tilde{r}_{ui}$	Prediction rating of the model

uct is used to learn the linear and nonlinear relationship of higher-order interaction vectors at the same time.
2. To enhance the ability to learn interaction from a local perspective, we investigate the differences between the explicit and implicit neighbors of users and items, propose a strategy combining both neighbors, then select the nearest neighbors calculated by two kinds of feedback as the similar users and similar items. Experiments have verified that it is superior to the single feedback.
3. Whether datasets or algorithms, we try to use the simplest settings to verify whether the model is effective in the experiment, and to prove the scalability of the model. In addition, most deep learning recommender systems focuses on implicit top-N recommendation while we test it on the explicit recommendation task of rating prediction. The results show that the accuracy of the model is better than the current popular CF algorithms based on deep learning.

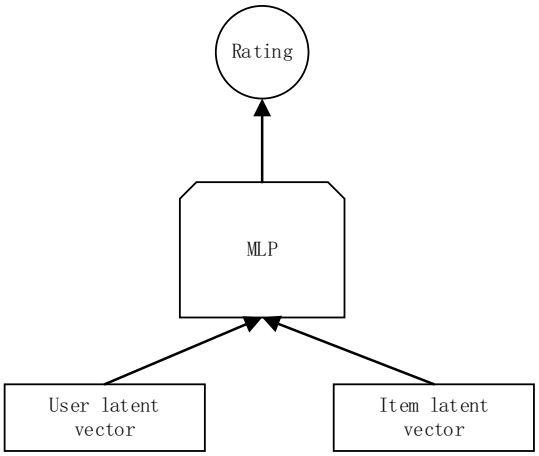

Fig. 1 Framework of NCF

MLP-Based CF

MF models maps users and items into the shared latent space. By learning the similarity function matching rating between latent vectors of users and items, the rating can have different meaning according to the different recommendation tasks. However, all ratings can represent the user's degree of preference for the item. Its main idea is to decompose the rating matrix into two low rank matrices representing users and items, and the row and column vectors of the matrix represent the latent characteristics of users and items [21]. Finally, the matrix is fitted by inner product of vectors and gradient descent.

However, MF has its own limitations. Because the inner product only makes linear combination of latent features, it will lead to the lack of feature expression ability in these models, and the recommendation result will be prone to bias. Therefore, NCF uses MLP instead of inner product to learn higher-order feature interaction between latent vectors of users and items, as shown in Fig. 1.

The latent vectors of users and items are input into MLP after processing. The similarity function between users and items is learned through MLP so that preference rating can be output. The formal process is shown in formula (1).

$$\tilde{r}_{ui} = F_n(F_{n-1} \cdots (F_1([\mathbf{p_u}, \mathbf{q_i}] + b))), \tag{1}$$

F_n represents the active function of each layer of MLP, and p_u and q_i are the latent vectors of user and item. Because the linear combination will limit the expression of the model, the two vectors p_u and q_i are connected by vector connection instead of inner product. In addition, MLP learning similar function can approach the characteristics of any function theoretically using deep neural network.

Framework of MLP-JSG

The general framework of our proposed model is shown in Fig. 2. The bottom input has four parts: similar users, similar items, user, and item. In this figure, the MLP on the left is used to learn the correlation between the user and similar items, while the MLP on the right is used to learn the correlation between the item and similar users. For example, to deduce the correlation between the item and similar users, the similarity between the user and remaining users need to be calculated using the corresponding behavior information of items. The inverted table of similarity for users can be generated, then the top-n users with the highest similarity as similar users to be embedded in MLP will be selected. Items are handled in the same way as users. Through MLP learning of these four parts, three high-order interaction vectors are obtained, and the interaction vectors can be fused to output the final rating.

The main purpose of modeling similar groups is to learn the correlation of their latent features, so as to integrate local information into the model. We take the CF based on user's neighbors as an example. Similar users are considered to have consistent preferences for each item. However, predicting preferences only through similar behaviors is not enough to express their latent feature interaction. This is one of the reasons why the effectiveness of traditional CF based on neighbor does not work well. Meantime, the interaction between similar items and users is same. Owing to MLP can assign and train the weight of each vector latent factor through multiple hidden layers, which is enough to express the latent feature interaction among user-similar items and item-similar users.

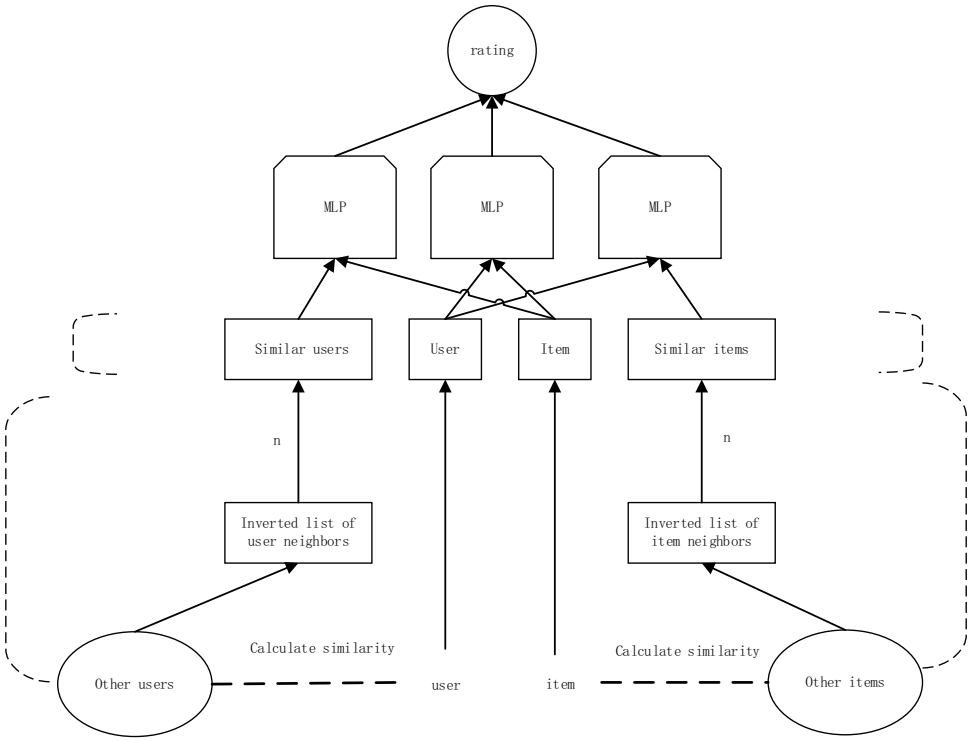

Fig. 2 General framework of MLP-JSG

Utilization of Local Information

Let the user and item in the datasets be u and i, respectively, then each user and item in the datasets can be expressed as $u \in U$ and $i \in I$. Let R be the explicit feedback matrix and B be the implicit feedback matrix. Therefore, we can set $b_{ui} \in B$ as one case of implicit behavior and $r_{ui} \in R$ as the explicit behavior. Since we only utilize rating score and rating behavior, b_{ui} indicates whether the user has rating behavior for the item, r_{ui} indicates the user's specific rating for the item, and B can be reconstructed by binarization, as shown in formula (2).

$$b_{ui} = \begin{cases} 1 & r_{ui} > 0 \\ 0 & r_{ui} = 0 \end{cases}. \tag{2}$$

The nearest neighbor inverted list of users and items can be generated by calculating similarity, and the top-n users and items can be selected as similar users and items. In neighbor-based recommendation, there are many ways to calculate the similarity, and the difference between explicit and implicit neighbor leads to a lot of improved algorithms. However, we are committed to proving the universality and extensibility of the model by the simplest method, so we choose cosine similarity as the measurement method, and let j and k be two users or two items. The similarity calculation method is shown in formula (3).

$$\text{sim}(j,k) = \cos(M_j, M_k) = \frac{M_j \cdot M_k}{\left\|M_j\right\| \left\|M_k\right\|}, \tag{3}$$

where M_j and M_k represent two rows or two columns of the selection matrix. When M is R, explicit similarity is obtained, and when M is B, implicit similarity is obtained. We use these two similarities to sort from the high to the low, get the inverted table of user u and item i, and select the first n items. Then we can get four similar groups: u_n's explicit similar users $\text{sim}U^e{}_n$, u_n's implicit similar items $\text{sim}U^i{}_n$, i_n's explicit similar items $\text{sim}I^e{}_n$, and i_n's implicit similar items $\text{sim}I^i{}_n$.

Owing to explicit feedback and implicit feedback are different in expressing the preference between users and items, the similar groups obtained by these two methods usually only show partial overlap. We also use the same method to calculate the similar users and similar items. Taking u_1 as an example, the similar users are usually presented as shown in Fig. 3. We can see that only part of their neighbors are same.

As shown in Fig. 4, we combine the two types of similar users to select the nearest neighbors. Among the first n neighbors of $\text{sim}U^e{}_1$ and $\text{sim}U^i{}_1$, the same neighbors u_2, u_7, u_{23}, and u_{55} are selected as the overlapping neighbors in $\text{sim}U^j{}_1$, which reflects the more strict selection criteria. Because explicit feedback and implicit feedback have their own limitations, explicit feedback is affected by user

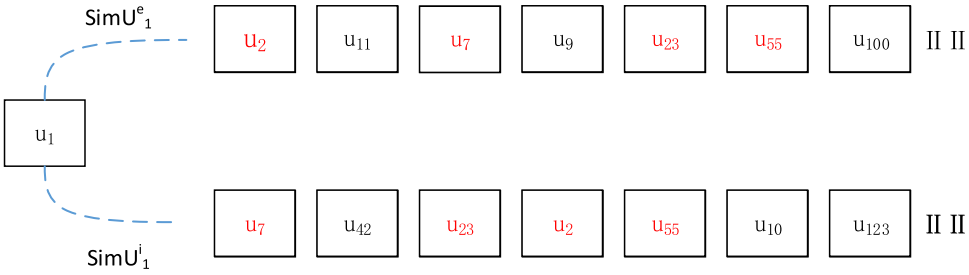

Fig. 3 Explicit and implicit similar users

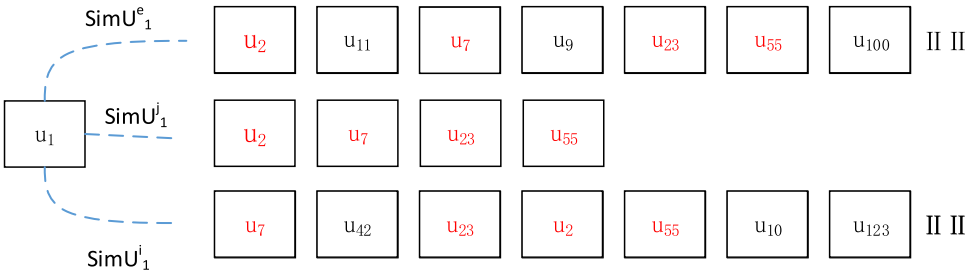

Fig. 4 Joint similar users

activity, item popularity or other factors, and implicit feedback can only reflect whether the user is interested in the item, not the degree of interest. Both of them have their own characteristics in reflecting local information so that we choose their overlapping users as similar users. Under this condition, these overlapping users represents a more accuracy neighbors that satisfies two types of feedback, so the local information reflected by them will be more accurate and valuable. On the contrary, for other users, they can be considered as users which is relatively inaccurate and have greater deviation in reflecting local information, and should be removed from the final similar users. Because we combine the two kinds of feedback to select the similar users, it's called the joint similar users. In addition, we filter similar items in the same way as users, and these two methods are called Joint Similar Group (JSG) together.

MLP-JSG Prediction Model

We apply the local information of 3.5 to the general framework in 3.4, and propose the MLP-JSG rating prediction model. On the fusion of global information and local information, we use explicit ratings for learning the relationship among user-similar items and item-similar users, respectively, as shown in Fig. 5.

1. Embedding layer

The bottom input of the model are four item vectors, for user u, item i, joint similar users $\text{sim}U^j = \{u_1, u_2 \ldots u_n\}$, joint similar items $\text{sim}I^j = \{i_1, i_2 \ldots i_n\}$. To investigate the recommendation effect of the model without adding any auxiliary information, these four parts are encoded by ID in one-hot mode. Then, by embedding the four one-hot vectors, the dense representation is performed, as shown in formula (4).

$$\mathbf{y} = \mathbf{W}^{\mathbf{T}}\mathbf{x} \tag{4}$$

Here y is the embedding vector, x is the one-hot vector, W^T is the weight matrix, and the embedding process is realized through a dense connection layer. Four latent vectors $\boldsymbol{u}$, $\boldsymbol{i}$, $\textbf{sim}\boldsymbol{U}$, $\textbf{sim}\boldsymbol{I}$ are generated. Next, the deep correlation between the four through MLP layers can be learned.

2. MLP layers

In Fig. 5, $\boldsymbol{u}$-$\boldsymbol{i}$ MLP layerN means that multiple hidden layers are used to learn the deep correlation between $\boldsymbol{u}$ and $\boldsymbol{i}$ two vectors. Similarly, $\textbf{sim}\boldsymbol{U}$-$\boldsymbol{i}$ represents similar users and items, and $\boldsymbol{u}$-$\textbf{sim}\boldsymbol{I}$ represents user and similar items. The formal result of $\boldsymbol{u}$-$\boldsymbol{i}$ MLP is shown in formula (5).

$$\mathbf{a_0} = [\mathbf{u}, \mathbf{i}],\ \mathbf{a_1} = g\left(\mathbf{W_1^T} \times \mathbf{a_0} + b\right) \cdots \mathbf{a_n} = g\left(\mathbf{W_n^T} \times \mathbf{a_{n-1}} + b\right), \tag{5}$$

W^T is the weight matrix of MLP model training for each layer, a_0 means to connect the $\boldsymbol{u}$ and $\boldsymbol{i}$ vectors, b is the bias factor in the layer, and $\boldsymbol{a_n}$ is the high-order interaction vector of users and items after MLP learning. Similarly, the $\boldsymbol{U}$-$\boldsymbol{i}$ MLP and $\boldsymbol{u}$-$\boldsymbol{I}$ MLP are shown in formulae (6) and (7).

$$\mathbf{b_0} = [\text{simU}, \mathbf{i}],\ \mathbf{b_1} = g\left(\mathbf{W_1^T} \times \mathbf{b_0} + b\right) \cdots \mathbf{b_n} = g\left(\mathbf{W_n^T} \times \mathbf{b_{n-1}} + b\right) \tag{6}$$

$$\mathbf{c_0} = [\mathbf{u}, \textbf{simI}],\ \mathbf{c_1} = g\left(\mathbf{W_1^T} \times \mathbf{c_0} + b\right) \cdots \mathbf{c_n} = g\left(\mathbf{W_n^T} \times \mathbf{c_{n-1}} + b\right) \tag{7}$$

The high-order interaction vectors $\boldsymbol{a_n}$, $\boldsymbol{b_n}$ and $\boldsymbol{c_n}$ can be obtained through three MLPs, and the output result after

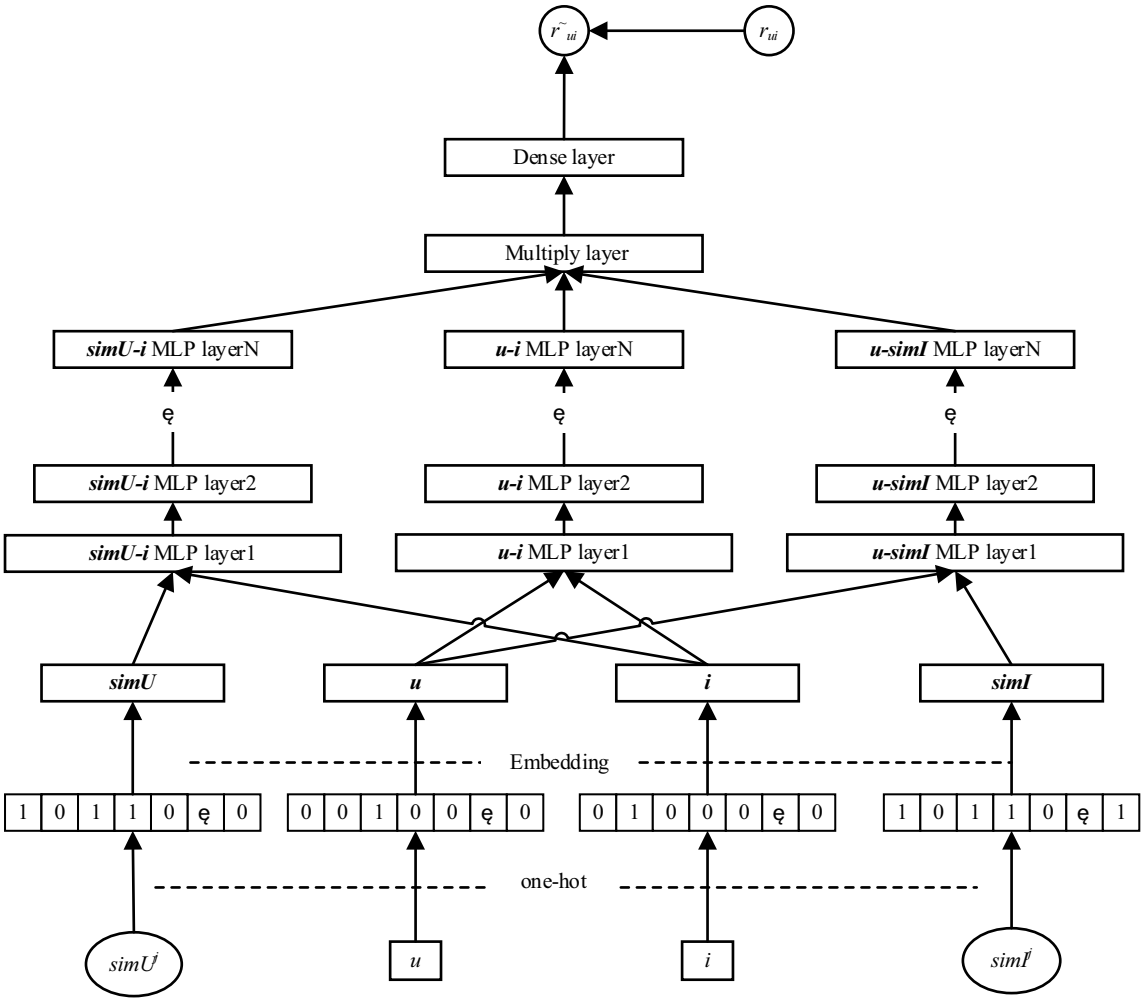

Fig. 5 MLP-JSG prediction model

combining the three vectors is shown in formula (8). [$\boldsymbol{a_n}$ $\boldsymbol{b_n}$ $\boldsymbol{c_n}$] represents the integration of these three vectors, and g is the activation function.

$$\tilde{r}_{ui} = g(\mathbf{W} \times [\mathbf{a_n}\, \mathbf{b_n}\, \mathbf{c_n}] + b). \tag{8}$$

In MLP layers, $\boldsymbol{u}$ and $\boldsymbol{i}$ is embedded in the input layer by one-hot encoding, so it represents the latent vectors of users and items, respectively. The principle of $\mathbf{sim}\boldsymbol{U}$, $\mathbf{sim}\boldsymbol{I}$ is the same, which represents the latent vector of similar users and similar items, and the latent factor of vector is similar feature.

Neither inner product nor simple vector connection is enough to express the interaction relationship between users and items in CF, so $\boldsymbol{a_n}$, $\boldsymbol{b_n}$ and $\boldsymbol{c_n}$ are needed to learn the high-level feature interaction of users and items. Here $\boldsymbol{a_n}$ is the interaction vector of user-item latent vector, $\boldsymbol{b_n}$ is the interaction vector for user-similar items, and $\boldsymbol{c_n}$ is for item-similar users. This model trains $\boldsymbol{a_n}$, $\boldsymbol{b_n}$ and $\boldsymbol{c_n}$ by adding multiple hidden layers. At the end of the model, the three high-order interaction vectors are fused, then the results are output through a dense connection layer.

The way to fuse the three higher-order vectors is to combine these vectors through the element product (the product of the elements at the corresponding positions of these three vectors), as shown in formula (9).

$$\tilde{r}_{ui} = g(\mathbf{W} \times (\mathbf{a_n} \circ \mathbf{b_n} \circ \mathbf{c_n}) + b). \tag{9}$$

The symbol ∘ represents the product of elements, which is similar to the linear inner product of MF. The process of making the product of elements for these three vectors can be regarded as a linear combination of higher-order interactive features. Therefore, our model has the ability to learn both linear and nonlinear relations of vectors.

3. Loss function

For the rating prediction model, since the ultimate goal is to output the user's explicit preference for items, the model needs to solve the regression problem. Therefore, the objective function is set as MSE function, and the weight matrix of each layer is trained by the MSE difference between the predicted rating and the real rating, as shown in formula (10).

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(r_{ui} - \tilde{r}_{ui})^2. \tag{10}$$

Experiments

Dataset and Evaluation

As shown in Table 2, We evaluated our models on four public datasets: MovieLens-100k, MovieLens-Latest-small, Book-crossing, Amazon Musical Instruments. The two MovieLens datasets have been processed by the provider. For the Book-crossing and Amazon Musical Instruments, users with at least 10 items are selected as valid data, and each valid item has been rated by at least 10 users. The table shows the filtered datasets.

In addition, the Book-Crossing dataset contains implicit ratings in the table, and real implicit feedback is marked in the dataset to represent interaction between a user and a book. Therefore, we use real implicit feedback in this dataset to calculate similarity, rather than the binary simulation of explicit feedback in the other datasets.

These four datasets are divided into training set and test set according to the ratio of 8:2. The model also needs to use validation set which is also divided into training set and validation set in training data according to the ratio of 8:2.

Since the output of the model is an explicit ratings, the Root Mean Square Error (RMSE) and Mean Squared Error (MAE) are used as evaluation metrics in the experimental results. The lower these two metrics are, the better performance is. Among them, RMSE is more sensitive to the values with high errors, which can better reflect the difference of algorithm effect.

Table 2 Datasets

Datasets	Users	Items	Ratings	Sparsity (%)
MovieLens-100k	943	1682	100,000	93.7
MovieLens-latest-small	610	9724	100,836	98.3
Book-crossing	12,587	15,294	419,075	99.9
Amazon musical instruments	15,785	7146	114,768	99.8

Comparison Algorithms

Since our model belongs to CF, some representative algorithms of CF for comparison are selected, which are listed as follows:

1. MLP: MLP-JSG is improved by combining MLP with Joint similar groups, so it is compared with the original model.
2. NeuMF: It combines two models of Generalized MF (GMF) and MLP, which is the most powerful model in the generalization of NCF framework.
3. Basic kNN: Here we choose the standard user-based algorithm.
4. kNN with Mean: Considering the different rating habit of users, this neighbor-based CF is proposed to remove the influence of rating habit.
5. SVD++: SVD++ combines explicit and implicit feedback from a global perspective, and utilizes the interaction neighbor as local information, which is one of the best methods in MF.
6. The MLP based on Implicit Similar Groups (MLP-ISG): To highlight the value of combining explicit and implicit feedback in the local information, we only use binary implicit feedback to find the similar groups on the basis that all variables of the method are consistent.

Parameter Setting

MLP, MLP-JSG, NeuMF and MLP-ISG belong to neural network algorithms, and their parameters are set to uniform values. As shown in Table 3, the number of latent factors represents the number of neurons in the last layer, which is set to 32. MLP-JSG, MLP-ISG, MLP, kNN Basic and kNN with Mean need to set the nearest neighbors. The number of them is increased by 1%, 5%, 10%, 15%, 20% and 25%, which takes into account the large difference in the number of users and items, so similar groups are selected according to the top percentage.

Analysis of Results

Figures 6 and 7 show the experimental results of MovieLens-100 k and MovieLens-Latest-small, respectively. Figure (a) (d) represents the comparison between

Table 3 Parameter setting

Loss function	Optimizer	Embedding dimension
MSE	SGD	64
Regularization	Activation function	Latent factors
L2	ReLu	32

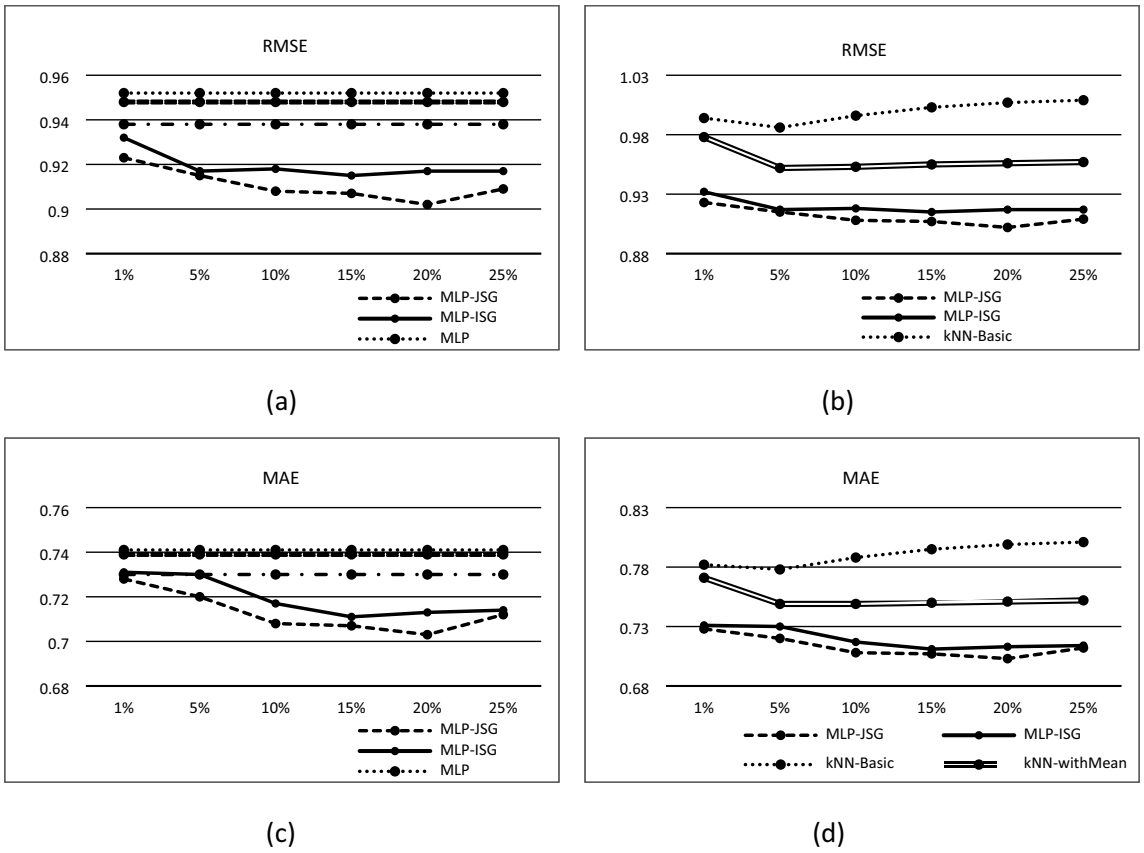

Fig. 6 Result of MovieLens-100k

MLP-JSG and other latent factor model, and Figure (b) (d) represents its comparison with kNN-based CF. Figure (a) (b) represents the result of RMSE, and Figure (c) (d) represents the result of MAE. The abscissa of all figures represents the original percentage of the number of neighbors in similar groups.

It can be seen clearly that MLP-JSG outperforms these comparison algorithms in the both overall and best performance in these two datasets, and achieves the best after embedding more than 10% of the nearest neighbors. In all Figure(a)(d) of two figures above, NeuMF performs best in the comparison algorithm, which is consistent with the performance of NCF framework in the implicit feedback experiment. The performance of MLP-JSG is much better than that of MLP and SVD++. In Figure(b)(d), the performance of the two kNN algorithms is weaker than latent factor model, and their performance is obviously weaker than that of MLP-JSG even when 1% nearest neighbor in joint similar group is selected. It is worth noting that kNN algorithm actually utilizes more nearest neighbors than MLP-JSG in each percentage, because we filter by combining two types of feedback when generating similar groups, and only overlapping nearest neighbors are selected, which further shows the effectiveness of our algorithm.

In these two figures, for the MLP-JSG model which only uses a single implicit feedback to find the nearest neighbors, its performance is also better than comparison algorithms, which shows the effectiveness of MLP for modeling user-similar items and item-similar users in their deep correlation, and proves that our model achieves ideal results in the fusion of global information and local information. Therefore, it proves that user and similar items, item and similar users have deep correlation, and their value is worth making full use of. However, its performance is weaker than MLP-JSG, which indicates that using both explicit feedback and implicit feedback can enhance the accuracy and effectively improve the performance of the latent factor model. At the same time, it is proved that the accuracy of local information can affect the fusion model performance, and it is insufficient to obtain local information only using one kind of feedback.

In addition, Book-Crossing and Amazon Musical Instruments datasets are used to evaluate the recommendation effect of our model on very sparse datasets. The results are shown in Figs. 8 and 9. It should be noticed that kNN algorithm is not compared here, because its performance is very poor on these datasets with high sparsity, so it is not worthy of comparison.

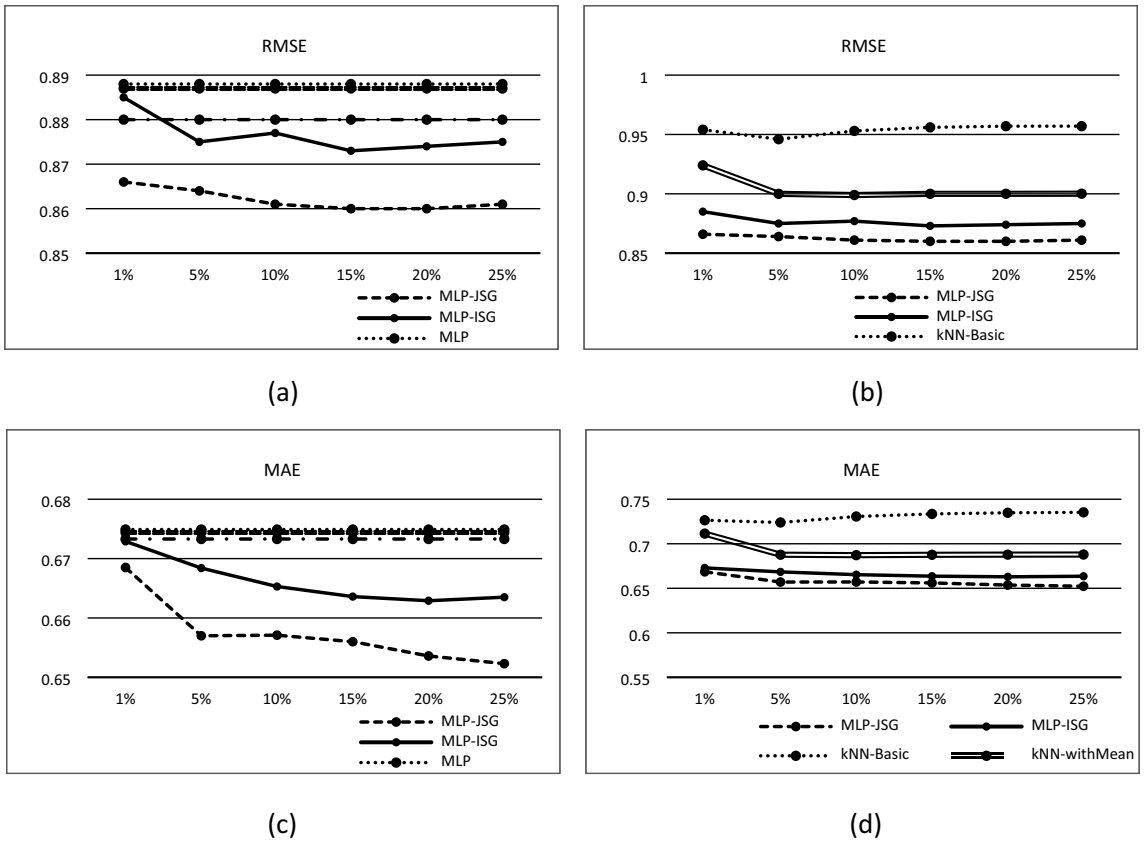

Fig. 7 Result of Movielens-latest-small

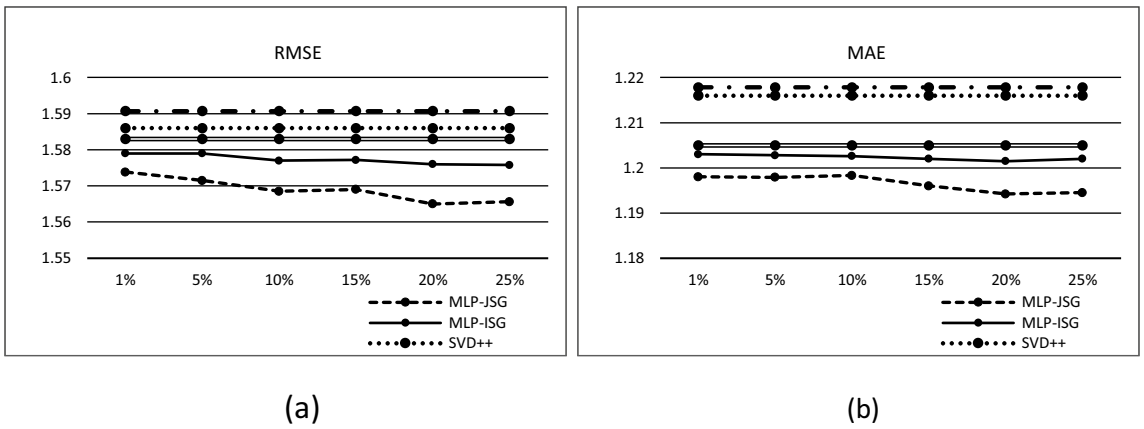

Fig. 8 Result of book-crossing

In Figs. 8 and 9, we can see that even on datasets with 99.8% and 99.9% sparsity, the performance of our model is still better than current popular algorithms of deep recommender system. Based on the experiment of Book-crossing, it proves that the real implicit feedback can be combined with explicit feedback locally to find the nearest neighbor, and the performance is better than MLP-JSG with implicit feedback alone.

However, it can be clearly found that the performance improvement of our model on sparse datasets is not as good as that on relatively dense datasets. For example, on MovieLens-100k, the RMSE of MLP-JSG decreased by

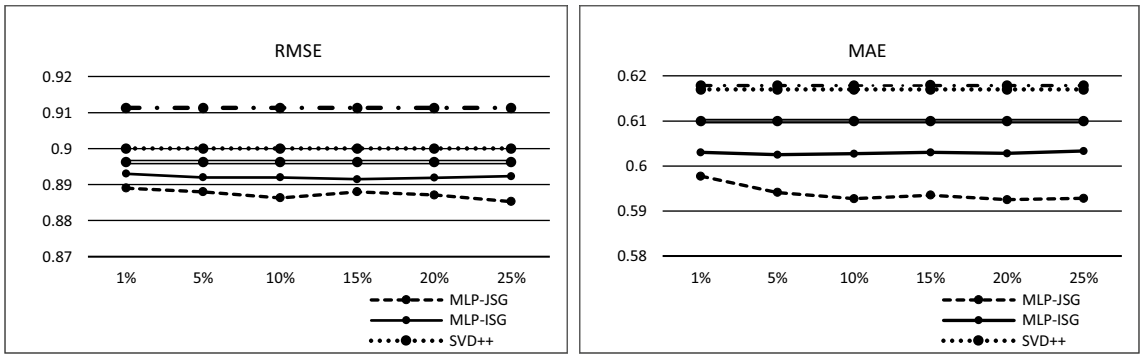

Fig. 9 Result of amazon musical instruments

Table 4 Comparison of running time

Datasets	MLP (s)	NeuMF (s)	MLP-JSG (s)	MLP-ISG (s)
MovieLens-100k	31.71	32.33	36.17	35.30
MovieLens-latest-small	34.15	35.03	38.29	37.68
Book-crossing	57.56	60.23	75.10	73.49
Amazon musical instruments	52.31	54.52	63.43	60.26

more than 0.035 compared with SVD++ at most, while on Book-Crossing and Amazon Musical Instruments datasets, it decreased by more than 0.02 at most, which shows that our model performs better on relatively dense datasets. This reason is easy to explain, because our model needs to calculate the local correlation of nearest neighbor. The accuracy of nearest neighbor can affect the performance of the model. In the case of extremely sparse data, it is difficult to acquire the nearest neighbors for ensuring the accuracy of result.

Therefore, it will also be a research focus in the future. Facing the problem of high data sparsity, how to mine more accurate nearest neighbors as local correlations is still a challenge which needs further exploration.

Finally, the running time needs to be evaluated between our model and compared algorithms on four datasets, which is shown in Table 4. There is a large gap in sparsity in these four datasets and it is also reflected in the difference of running time. In addition, our model is only compared to the neural network model such as MLP, NeuMF and MLP-ISG, because these neural network models usually require longer running time.

It also can be seen from Table 4 that our model takes longer running time than the ordinary deep collaborative filtering model because of two points: (1) our model needs longer time to calculate similar users and similar items for obtaining local correlation. The running time needed of Movielens-Latest-small is similar to MovieLens-100 k. However, in the Book-Crossing, because we used real implicit feedback data and the amount of implicit feedback data is huge so that it took a long time to calculate the implicit similarity. (2) Compared with NeuMF and MLP, our model has more latent vectors and more complex network structure.

In addition, the running time difference is obviously on the Amazon Musical Instruments and Book-Crossing datasets, because it takes longer time to calculate the similarity for more users and items, and the vector dimensions are larger. However, our model is still within a reasonable range for running time needed.

Discussion

In the fusion model of latent factor and local information, the accuracy of local information is very important when using the interaction behavior to find the nearest neighbors as the local information. At the same time, the explicit and implicit feedback can effectively improve the performance of the model. Therefore, we use explicit and implicit feedback to select more valuable neighbors. In addition to the correlation between users and items, the correlation between users or items and their similar groups is also studied using MLP to learn complex nonlinear relationship. Finally, it makes up the performance limitation of MLP caused by the lack of ability to capture local correlation and the inadequate usage of feedback information. Only by making full use of explicit rating which is obtained by binarization in our method, it can achieve ideal performance. Furthermore, the performance on the Book-Crossing proves that our model can combine explicit feedback with real implicit feedback.

In addition, this model has strong extensibility. Because we only choose the cosine similarity to verify whether the model is effective for the combining of explicit and implicit similar groups, there is a lot of improved algorithms which are worthy of further exploration and application. In this current research, only explicit ratings are utilized in

experiments. We will take advantage of richer explicit and implicit feedback to improve performance if more information can be obtained in specific applications.

Acknowledgements This work has been supported by Major Natural Science Research Projects of Colleges and Universities in Jiangsu Province of China (19KJA510011), National Natural Science Foundation of China (71871109).

Declarations

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Ricci F, Rokach L, Shapira B. Recommender systems: introduction and challenges. In: Recommender systems handbook. Boston: Springer; 2015. p. 1–34.
2. Lu J, Wu D, Mao M, Wang W, Zhang G. Recommender system application developments: a survey. Decis Support Syst. 2015;74:12–32.
3. Goldberg K, Roeder T, Gupta D, Perkins C. Eigentaste: a constant time collaborative filtering algorithm. Inf Retriev. 2001;4(2):133–51.
4. Koren Y. 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.
5. Xue HJ, Dai X, Zhang J, Huang S, Chen J. Deep matrix factorization models for recommender systems. In: IJCAI, Vol. 17. 2017. pp. 3203–3209.
6. Salakhutdinov R, Mnih A, Hinton G. Restricted Boltzmann machines for collaborative filtering. In: Proceedings of the 24th international conference on Machine learning. 2007. pp. 791–798.
7. Sedhain S, Menon AK, Sanner S, Xie L. Autorec: autoencoders meet collaborative filtering. In: Proceedings of the 24th international conference on World Wide Web. 2015. pp. 111–112.
8. Liang H, Baldwin T. A probabilistic rating auto-encoder for personalized recommender systems. In: Proceedings of the 24th ACM International on conference on information and knowledge management. 2015. pp. 1863–1866.
9. Li S, Kawale J, Fu Y. Deep collaborative filtering via marginalized denoising auto-encoder. In: Proceedings of the 24th ACM international on conference on information and knowledge management. 2015. pp. 811–820.
10. Kim D, Park C, Oh J, Lee S, Yu H. Convolutional matrix factorization for document context-aware recommendation. In: Proceedings of the 10th ACM conference on recommender systems. 2016. pp. 233–240.
11. Singhal A, Sinha P, Pant R. Use of deep learning in modern recommendation system: a summary of recent works. 2017. arXiv preprint arXiv:1712.07525.
12. Deng ZH, Huang L, Wang CD, Lai JH, Philip SY. Deepcf: a unified framework of representation learning and matching function learning in recommender system. In: Proceedings of the AAAI conference on artificial intelligence, vol. 33, No. 01. 2019. pp. 61–68.
13. He X, Liao L, Zhang H, Nie L, Hu X, Chua TS. Neural collaborative filtering. In: Proceedings of the 26th international conference on world wide web. 2017. pp. 173–182.
14. Chen W, Cai F, Chen H, Rijke MD. Joint neural collaborative filtering for recommender systems. ACM Trans Inf Syst (TOIS). 2019;37(4):1–30.
15. He X, Du X, Wang X, Tian F, Tang J, Chua TS. Outer product-based neural collaborative filtering. 2018. arXiv preprint arXiv:1808.03912.
16. Bai T, Wen JR, Zhang J, Zhao WX. A neural collaborative filtering model with interaction-based neighborhood. In: Proceedings of the 2017 ACM on conference on information and knowledge management. 2017. pp. 1979–1982.
17. Ebesu T, Shen B, Fang Y. Collaborative Memory Network for Recommendation Systems[C]. In: The 41st international ACM SIGIR conference on research & development in information retrieval. 2018. pp. 515–524.
18. Liu H, Liu H, Ji Q, Zhao P, Wu X. Collaborative deep recommendation with global and local item correlations. Neurocomputing. 2020;385:278–91.
19. Christakopoulou E, Karypis G. Local item-item models for top-n recommendation. In: Proceedings of the 10th ACM conference on recommender systems. 2016. pp. 67–74.
20. Xie R, Ling C, Wang Y, Wang R, Xia F, Lin L. Deep feedback network for recommendation. In: IJCAI. 2020. pp. 2519–2525.
21. Koren Y, Bell R, Volinsky C. Matrix factorization techniques for recommender systems. Computer. 2009;42(8):30–7.