A BP Neural Network Based Recommender Framework With Attention Mechanism

Chang-Dong Wang, *Member, IEEE*, Wu-Dong Xi, Ling Huang, *Member, IEEE*, Yin-Yu Zheng, Zi-Yuan Hu, and Jian-Huang Lai, *Senior Member, IEEE*

Abstract—Recently, some attempts have been made in introducing deep neural networks (DNNs) to recommender systems for generating more accurate prediction due to the nonlinear representation learning capability of DNNs. However, they inevitably result in high computational and storage costs. Worse still, due to the relatively small number of ratings that can be fed into DNNs, they may easily suffer from the overfitting issue. To tackle these issues, we propose a novel recommendation framework based on Back Propagation (BP) neural network with attention mechanism, namely BPAM++. In particular, the BP neural network is utilized to learn the complex relationship between the target user and his/her neighbors and the complex relationship between the target item and its neighbors. Compared with DNNs, the shallow neural network, i.e., BP neural network, can not only reduce the computational and storage costs, but also alleviate the overfitting issues in DNNs caused by a relatively small number of ratings. In addition, an attention mechanism is designed to capture the global impact of the nearest users of the target user on their nearest target user sets. Extensive experiments conducted on eight benchmark datasets confirm the effectiveness of the proposed model.

Index Terms—Recommender system, BP neural network, attention mechanism, computational and storage costs, overfitting

1 Introduction

The era of information explosion has arrived, in which people can hardly hit what they really prefer when dealing with a huge number of choices. To tackle this problem, personalized recommender systems have been proposed and widely applied in e-commerce platforms and news/music/movie/education platforms [1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11]. The basic idea of recommender systems is to recommend the most suitable items to users according to the hidden preferences of users discovered from the historical behavior data. Collaborative Filtering (CF) is one of most classical technologies in personalized recommender systems, which infers users' preferences from their historical behavior [12], [13], [14], [15], [16].

The traditional CF recommendation algorithms are generally divided into two categories: Matrix Factorization (MF) methods [17], [18], [19], [20] and neighborhood-based CF methods [21], [22], [23], [24]. The MF methods map the users and items into a common representation space. Then the users' ratings on items are modeled as the inner product of their latent vectors [25], [26], [27], [28]. However, the MF methods easily suffer from the sparsity issue due to the long-tailed distribution of the rating data in the real-world applications [29]. On the other hand, the neighborhood-based CF algorithms predict the target ratings by averaging (weighted) ratings of similar entities (users or items). In [30], both user and item information are taken into account to improve the prediction quality, but it only achieves some performance improvement. This indicates that it is hard to predict the target ratings accurately by utilizing the linear combination of similar entities' ratings.

Inspired by the significant success of deep learning, some efforts have been made in utilizing the representation learning abilities of Deep Neural Networks (DNNs) to learn users' preference [31], [32], [33], [34], [35]. For instance, in [32], a novel Deep Matrix Factorization (DMF) model with deep neural network was proposed, which mapped the users and items into a common low-dimensional space with non-linear projections to make prediction. However, the high computational and storage costs caused by the complex structures prevent it from applying to large-scale data [36]. As shown in our experiments, when dealing with the classical Amazon (Kindle Store) dataset which consists of 68,223 users, 61,934 items and 982,619 ratings, most of the existing DNNs-based approaches fail to generate the results. Besides, compared with the massive parameters of DNNs, there are only a relatively small number of ratings that can be fed into DNNs as the training samples due to the sparsity issue in recommender systems, which can easily lead to overfitting.

To address the above issues, this paper proposes a novel recommendation framework called BPAM++, which is based on Back Propagation (BP) neural network[1] [37] with

- *Chang-Dong Wang, Wu-Dong Xi, Zi-Yuan Hu, and Jian-Huang Lai are with the School of Data and Computer Science, and the Key Laboratory of Machine Intelligence and Advanced Computing, Ministry of Education, Sun Yat-sen University, Guangzhou 510006, China. E-mail: changdongwang@hotmail.com, {xiwd, huzy33}@mail2.sysu.edu.cn, stsljh@mail.sysu.edu.cn.*
- *Ling Huang is with the College of Mathematics and Informatics, South China Agricultural University, Guangzhou 510642, China. E-mail: huanglinghl@hotmail.com.*
- *Yin-Yu Zheng is with the Lingnan (University) College, Sun Yat-sen University, Guangzhou 510006, China. E-mail: zhengyy.sysu@foxmail.com.*

Manuscript received 24 Dec. 2019; revised 4 Sept. 2020; accepted 9 Sept. 2020. Date of publication 16 Sept. 2020; date of current version 3 June 2022. (Corresponding author: Chang-Dong Wang.) Recommended for acceptance by G. Koutrika. Digital Object Identifier no. 10.1109/TKDE.2020.3023976

1. In this paper, following the early literature, BP neural network refers to the classical shallow neural network having only one hidden layer, as opposed to DNNs having several hidden layers.

1041-4347 © 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: ANNA UNIVERSITY. Downloaded on February 07,2023 at 09:43:58 UTC from IEEE Xplore. Restrictions apply.

attention mechanism [38], [39]. By introducing the BP neural network into the neighborhood-based CF algorithm, the ratings of the nearest users and items are fed into the BP neural network instead of undertaking a linear combination in the traditional algorithms. Moreover, for a given user-item pair composed of the target user and the target item, we consider not only the ratings of the nearest users of the target user to the target item but also the ratings of the target user to the nearest items of the target item. In this manner, the proposed BPAM++ model can capture not only the complex relationship between the target user and his/her neighbors, but also the complex relationship between the target item and its neighbors. If only considering the ratings of the nearest users of the target user to the target item, BPAM++ will degenerate into our previously proposed BPAM model [40]. Considering the large number of items, we utilize the shallow network, i.e., BP neural network, to reduce the computational and storage costs. Unlike considering all ratings in the existing DNNs-based CF algorithms, the influence of non-similar users and non-similar items can be eliminated by only selecting nearest users' ratings and nearest items' ratings respectively. Moreover, since the BP neural network has a relatively small number of parameters, the proposed BPAM++ model is able to perform well even in the case of only a relatively small number of ratings.

In addition, the attention mechanism is incorporated into the BP neural network to capture the global impact of the nearest users of the target user on their nearest target user sets. The nearest target user set of a user is defined to be a set of users such that this user is one of the nearest users of each user in this set. To this end, a global attention matrix is introduced, which enables the model to share weights across different target users and obtain more accurate weights of the target user from the nearest users. In this way, a unified training model has been constructed, which consists of not only the local weights between the input layer and the hidden layer in the BP neural network but also the global attention matrix encoding the global impact. When predicting the ratings of the target user, BPAM++ takes into account both the local weights and the global attention matrix to make better prediction.

The main contributions of this work are as follows.

- A novel neighborhood-based CF recommendation framework called BPAM++ is proposed, which overcomes the high computational and storage costs and alleviates the overfitting issues in DNNs.
- The BP neural network is utilized to model not only the complex relationship between the target user and his/her neighbors, but also the complex relationship between the target item and its neighbors, instead of undertaking a linear combination of the neighbor ratings in the traditional algorithms.
- A novel attention mechanism is introduced to capture the global impact of the nearest users of the target user on their nearest target user sets by means of introducing a global attention matrix.
- Extensive experiments on eight real-world datasets are conducted to demonstrate the effectiveness of the proposed model. The results show that BPAM++ outperforms other state-of-the-art algorithms, and the proposed attention mechanism improves its performance significantly.

The rest of this paper is organized as follows. In Section 2, we will briefly introduce the related work. Section 3 will describe in detail the proposed model. In Section 4, we will report the experimental results. Finally, the conclusion will be drawn in Section 5.

Some part of this work was first presented in [40].

2 Related Work

2.1 Neighborhood-Based CF

Neighborhood-based CF algorithms are intuitive and interpretable, which calculate the similarity between entities (users or items) and then predict the ratings based on the similar entities. Various algorithms have been developed to improve the prediction accuracy from different aspects [1], [3], [41], [42], [43]. In [41], a method was proposed to simultaneously derive the interpolation weights as a global solution to an optimization problem, leading to some improvement of the prediction accuracy. In [42], the temporal information was utilized to improve the accuracy of CF algorithms. In [43], a similarity measure was proposed for neighborhood-based CF, which utilized all rating information comprehensively for locating useful neighbors of an active user in the sparse rating matrix, and did not depend on co-rated items. In our previous work [1], based on the observation of the Rogers' innovation adoption curve [44], a new recommendation algorithm termed INVBCF was proposed, which was able to recommend new items and niche items to users by means of innovators, so as to achieve the balance between serendipity and accuracy. In [45], a novel SVD++ model was proposed, which is an extension to the SVD-based latent factor model by integrating the neighborhood models. Specifically, in the neighborhood models, a second set of item factors was added to model the item-item similarity. Yan *et al.* [46] proposed a new collaborative filtering algorithm, which utilized Gaussian mixture model to cluster users and items respectively and extracted new features to build a new interaction matrix. Meanwhile, a new similarity calculation method was proposed, which was combined by triangle similarity and Jaccard similarity. However, these models only achieve some performance improvement, since they still utilize the linear combination of similar entities' ratings to predict the target rating. In this paper, different from these models, both the ratings of the nearest users of the target user to the target item and the ratings of the target user to the nearest items of the target item are fed into BP neural network with attention mechanism and mapped nonlinearly to the target user's ratings.

2.2 DNNs-Based CF

Recently, due to the nonlinear representation learning abilities, DNNs have been introduced in recommender systems to learn users' preference for items [32], [34], [47], [48], [49]. DNNs are utilized to learn the complex mapping relationship between user-item the latent factor representations and the matching ratings. For example, in [32], a deep learning architecture called DMF was presented to learn a common low dimensional space for the representations of users and items, where a two-pathway neural network architecture

Authorized licensed use limited to: ANNA UNIVERSITY. Downloaded on February 07,2023 at 09:43:58 UTC from IEEE Xplore. Restrictions apply.

was used to replace the linear embedding operation. In [47], a Collaborative Deep Learning (CDL) was proposed, which jointly performed deep representation learning for the content information and collaborative filtering for the rating matrix. In [34], three instantiations, namely Generalized Matrix Factorization (GMF), Multi-Layer Perceptron (MLP) and Neural Matrix Factorization (NeuMF), were proposed, which modeled user-item interactions in different ways. In addition, autoencoders have been applied to learn the user and item latent factors for recommendation systems [50], [51]. To overcome the sparse nature of the ratings and the side information and learn user and item vectors from intentionally corrupted inputs, Li *et al.* [50] proposed a general deep architecture for collaborative filtering by combining probabilistic matrix factorization with marginalized Denoising Stacked Auto-Encoders (DSAE). Fan *et al.* [52] presented a novel graph neural network framework (GraphRec) for social recommendations, which modeled the user-user social graph and the user-item graph coherently and provided a principled approach to jointly capture interactions and opinions in the user-item graph. In our previous work [49], a general Deep Collaborative Filtering (DeepCF) framework was proposed to combine the strength of the representation learning-based CF methods and the matching function learning-based CF methods.

These DNNs-based models have greatly improved the prediction accuracy, but they suffer from the issue of high computational and storage costs. If there are only a relatively small number of samples fed into DNNs, these DNNs-based models with massive parameters can easily lead to overfitting. However, the BP neural network with a relatively small number of parameters is able to maintain good performance even in the case of only a relatively small number of samples.

2.3 Attention Mechanism in Recommendation Systems

Attention-based architectures, which learn to focus their "attention" to specific parts [53] or combine both local and global information [54], have shown great potential in recommendation algorithms. For instance, in [53], the attention mechanism was introduced to capture the varying attention vectors of each specific user-item pair. In [54], an effective attention-based Convolutional Neural Networks (CNNs) was proposed for performing the hashtag recommendation task, which combined the local attention channel and the global channel to obtain the final embedding of the microblog. Attention factorization machine was designed to improve the performance of factorization in [55], which discriminated the importance of different feature interactions via a neural attention network. In [56], a Neural Attentional Regression model with Review-level Explanations (NARRE) was proposed, which utilized a novel attention mechanism to automatically assign weights to reviews and explore the usefulness of reviews in a distant supervised manner. Tran *et al.* [57] proposed a Medley of Sub-Attention Networks (MoSAN) for group recommendation, which considered user-user interactions using sub-attention networks. In [58], an attention-based user model called ATRank was proposed, which utilized the attention mechanism to model user behavior after considering the influences brought by other behaviors. And it's observed that self-attention with time encoding is faster than the complex Recurrent Neural Networks (RNN) and Convolutional Neural Networks (CNN) structures in the sequential behavior encoding.

Motivated by the successes of various models, in this paper, we adopt the attention mechanism to capture the global impact of the nearest users of the target user on their nearest target user sets by means of introducing a global attention matrix. In this way, both the local weights and the global attention matrix are integrated to make more accurate predictions.

3 The Proposed Model

In this section, we will describe in detail the proposed BPAM++ model. Before introducing the BPAM++ model, we will first introduce the preliminaries and problem statement. Then, we will elaborate the construction of input vector, including the rating vector of nearest users, the rating vector of nearest items, and rating vector processing. Next, we will illustrate the architectures and implementations of BPAM++. Finally, we will give some analysis about the proposed BPAM++ model and its previous version, namely BPAM. For clarity, Table 1 summarizes the main notations used in this paper.

3.1 Preliminaries and Problem Statement

Suppose that in the recommender system we are given a user set $\mathcal{U}$ and an item set $\mathcal{V}$ respectively. Following [59], a user-item rating matrix $\mathbf{R} \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{V}|}$ is constructed from users' ratings on items as follows:

$$\mathbf{R}_{u,i} = \begin{cases} r_{u,i}, & \text{if user } u \text{ has rated item } i \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

where $r_{u,i}$ denotes the rating of user u on item i.

The goal of recommendation algorithms is to estimate the missing ratings in the rating matrix $\mathbf{R}$ [60]. The model-based methods generate the predicted rating $\hat{r}_{u,i}$ of user u to item i as follows:

$$\hat{r}_{u,i} = f(u, i|\Theta), \tag{2}$$

where f denotes the mapping function that maps the model input, e.g., the ratings of neighbor users or items, to the predicted rating of the corresponding user-item pair composed of user u and item i by utilizing the model parameters Θ [49].

The mapping function of neighborhood-based CF is a linear combination of neighbors' ratings, where the model parameters are mainly the weights obtained from different similarity functions. Despite their widespread application in industry, it is usually difficult for these simple linear mapping functions to make accurate rating prediction. On the other hand, the mapping functions of DNNs-based CF inherit the nonlinear representation learning capability from DNNs, which therefore are able to learn the nonlinear mapping relationship between the user-item latent factor representations and the matching ratings. However, due to the sparsity issue in recommender systems, there are only a relatively small number of ratings to be fed into DNNs with a large number of parameters, easily leading to the overfitting

Authorized licensed use limited to: ANNA UNIVERSITY. Downloaded on February 07,2023 at 09:43:58 UTC from IEEE Xplore. Restrictions apply.

TABLE 1
Summary of the Main Notations

$\mathcal{U}, \mathcal{V}$	User set, item set
$\mathbf{R} \in \mathbb{R}^{\lvert\mathcal{U}\rvert \times \lvert\mathcal{V}\rvert}$	Rating matrix
$r_{u,i}$	Rating of user u to item i
$\hat{r}_{u,i}$	Predicted rating of user u to item i
Θ	Model parameters
k	Pre-specified number of nearest users of each user u
$\mathbf{n}^u \in \mathbb{R}^{1\times k}$	KNU index vector of user u, i.e. the index vector storing the indexes of k nearest users (KNU) of user u
n_j^u	Index of the j-th nearest user of user u
$\mathbf{p}_i^u \in \mathbb{R}^{k\times 1}$	KNU rating vector of user u on item i, i.e. the ratings of the k nearest users of user u on item i
l	Pre-specified number of nearest items of each item i
$\mathbf{n}^i \in \mathbb{R}^{1\times l}$	LNI index vector of item i, i.e. the index vector storing the indexes of l nearest items (LNI) of item i
n_j^i	Index of the j-th nearest item of item i
$\mathbf{q}_i^u \in \mathbb{R}^{l\times 1}$	LNI rating vector of item i from user u, i.e. the ratings of user u to the l nearest items of item i
$\bar{\mathbf{p}}_i^u \in \mathbb{R}^{k\times 1}$	Post-processed KNU rating vector of user u on item i
$\bar{\mathbf{q}}_i^u \in \mathbb{R}^{l\times 1}$	Post-processed LNI rating vector of item i from user u
$\mathbf{d}_i^u \in \mathbb{R}^{(k+l+1)\times 1}$	Training sample for the given user-item pair composed of user u and item i
$\mathbf{x}_i^u \in \mathbb{R}^{(k+l)\times 1}$	Input rating vector
$\mathcal{T}^u$	Nearest target user set of user u, i.e., user u is one of the k nearest users of each user in $\mathcal{T}^u$
$\mathbf{A} \in \mathbb{R}^{\lvert\mathcal{U}\rvert\times o}$	Global attention matrix for all users, where o is the number of neurons in the hidden layer
$\mathbf{A}_{u,*} \in \mathbb{R}^{1\times o}$	The u-th row of $\mathbf{A}$, i.e., the global attention (impact) of user u on his/her nearest target user set
π_u	Number of items that user u has rated
$\mathcal{D}^u$	Training set for user u
$\mathbf{h}_i^u \in \mathbb{R}^{o\times 1}$	Values of the hidden layer
$\mathbf{W}^u \in \mathbb{R}^{(k+l)\times o}$	Local weight matrix of user u in the BP neural network
$\mathbf{A}_{\mathbf{n}^u,*} \in \mathbb{R}^{k\times o}$	Global attention sub-matrix of $\mathbf{A}$ corresponding to the k nearest users of user u
$\mathbf{b}_{input}^u \in \mathbb{R}^{o\times 1}$	Bias vector of user u among the input layer and the hidden layer
$\mathbf{w}_{hidden}^u \in \mathbb{R}^{o\times 1}$	Weight vector of user u among the hidden layer and the output layer
b_{hidden}^u	Bias value of user u among the hidden layer and the output layer
α	Trade-off parameter used to tune the importance of the global impact
N	Number of the tested ratings

issue. Moreover, the training process of DNNs usually results in high computational and storage costs.

In this paper, we utilize the BP neural network as the mapping function instead of DNNs to learn the complex relationship between the target user and his/her neighbors and the complex relationship between the target item and its neighbors so as to avoid the aforementioned issues. The BP neural network is the abbreviation of error back propagation neural network, which is one of the most widely used neural network models [37]. The BP neural network has shown the ability of highly non-linear mapping, and can well fit the non-linear relationship between the given user-item pairs and their neighbors. Most importantly, the BP neural network is a shallow network. Therefore it can be trained efficiently and make accurate prediction even if there are not a large number of samples in recommender systems.

3.2 Input Vector Construction

3.2.1 Rating Vector of Nearest Users

The general process for BPAM++ is illustrated in Fig. 1. After extracting the rating matrix $\mathbf{R}$ from the database, for each user u, a set of k nearest users (KNU) can be obtained by calculating the cosine similarity between the rating vectors of users, where k is the pre-specified number of nearest users. Let $\mathbf{n}^u = [n_1^u, n_2^u, \ldots, n_k^u]$ denote the KNU index vector of user u, i.e., the index vector storing the indexes of k nearest users (KNU) of user u, where $n_j^u, \forall j = 1, \ldots, k$ denotes the index of the jth nearest user of user u.

Then for each user-item pair composed of user u and item i, a KNU rating vector $\mathbf{p}_i^u \in \mathbb{R}^{k\times 1}$ of user u on item i, i.e., the ratings of the k nearest users of user u on item i, is obtained from the rating matrix $\mathbf{R}$ with each entry $\mathbf{p}_i^u[j]$ being defined as follows:

$$\mathbf{p}_i^u[j] = \mathbf{R}_{n_j^u, i}, \quad \forall j = 1, \ldots, k. \tag{3}$$

That is, $\mathbf{p}_i^u[j]$ is the rating value of user n_j^u to item i.

3.2.2 Rating Vector of Nearest Items

Similarly, for each item i, a set of l nearest items (LNI) can be obtained by calculating the cosine similarity between the rating vectors of items, where l is the pre-specified number of nearest items. Let $\mathbf{n}^i = [n_1^i, n_2^i, \ldots, n_l^i]$ denote the LNI index vector of item i, i.e., the index vector storing the indexes of l nearest items (LNI) of item i, where $n_j^i, \forall j = 1, \ldots, l$ denotes the index of the jth nearest item of item i.

Then for each user-item pair composed of user u and item i, a LNI rating vector $\mathbf{q}_i^u \in \mathbb{R}^{l\times 1}$ of item i from user u, i.e., the ratings of user u to the l nearest items of item i, is obtained from the rating matrix $\mathbf{R}$ with each entry $\mathbf{q}_i^u[j]$ being defined as follows:

$$\mathbf{q}_i^u[j] = \mathbf{R}_{u, n_j^i}, \quad \forall j = 1, \ldots, l. \tag{4}$$

That is, $\mathbf{q}_i^u[j]$ is the rating value of user u to item n_j^i.

3.2.3 Rating Vector Processing

Recommendation algorithms are known to suffer from serious sparsity problem. That is, the number of ratings per user or per item obeys the long-tailed distribution [61]. Hence some of the values in the above defined KNU rating vectors and LNI rating vectors are zero. However, some users prefer to rate higher, while others prefer to rate lower. If feeding different users' zero-rating into the BP neural network, the preference biases won't be distinguished.

To this end, for each user-item pair composed of user u and item i, the KNU rating vector $\mathbf{p}_i^u$ of user u on item i is post-processed to be a new KNU rating vector $\bar{\mathbf{p}}_i^u \in \mathbb{R}^{k\times 1}$ of user u on item i as follows:

$$\bar{\mathbf{p}}_i^u[j] = \begin{cases} \text{mean}(\mathbf{R}_{n_j^u,*}), & \text{if } \mathbf{p}_i^u[j] = 0 \\ \mathbf{p}_i^u[j], & \text{otherwise} \end{cases}, \quad \forall j = 1, \ldots, k, \tag{5}$$

where $\mathbf{R}_{n_j^u,*}$ denotes the ratings of the jth nearest user of user u, and $\text{mean}(\mathbf{R}_{n_j^u,*})$ denotes the mean value of the non-zero elements of $\mathbf{R}_{n_j^u,*}$. That is, the missing rating $\mathbf{p}_i^u[j]$ in the KNU rating vector is estimated by the mean rating value of the corresponding user n_j^u.

Similarly, the LNI rating vector $\mathbf{q}_i^u$ of item i from user u is post-processed to be a new LNI rating vector $\bar{\mathbf{q}}_i^u \in \mathbb{R}^{l\times 1}$ of item i from user u as follows:

Authorized licensed use limited to: ANNA UNIVERSITY. Downloaded on February 07,2023 at 09:43:58 UTC from IEEE Xplore. Restrictions apply.

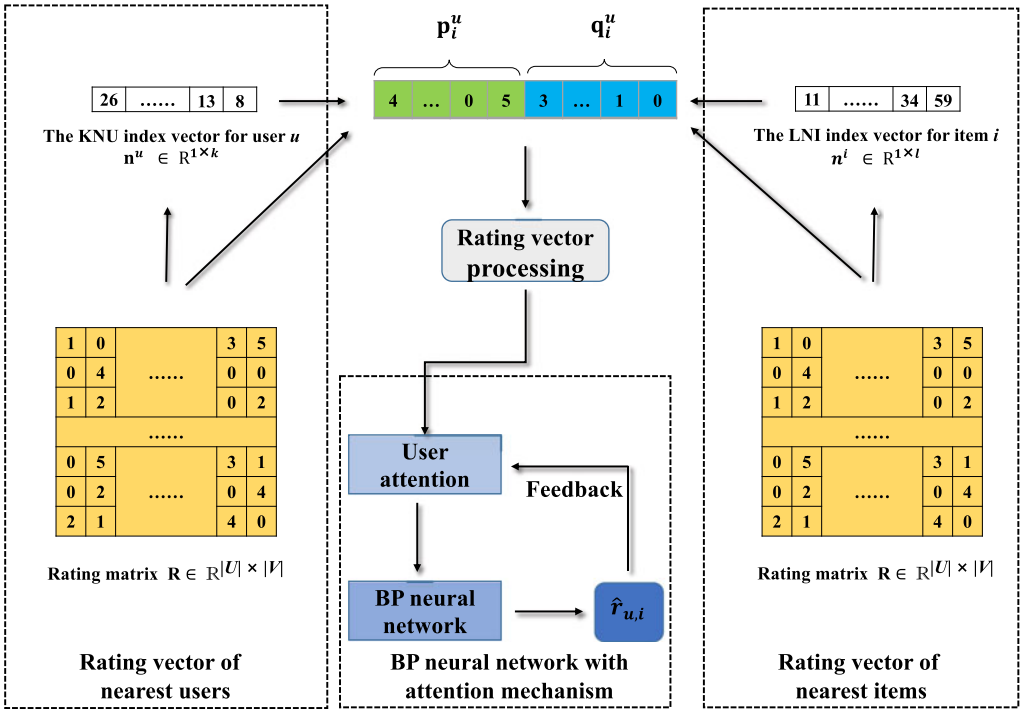


Fig. 1. The general process for BPAM++.

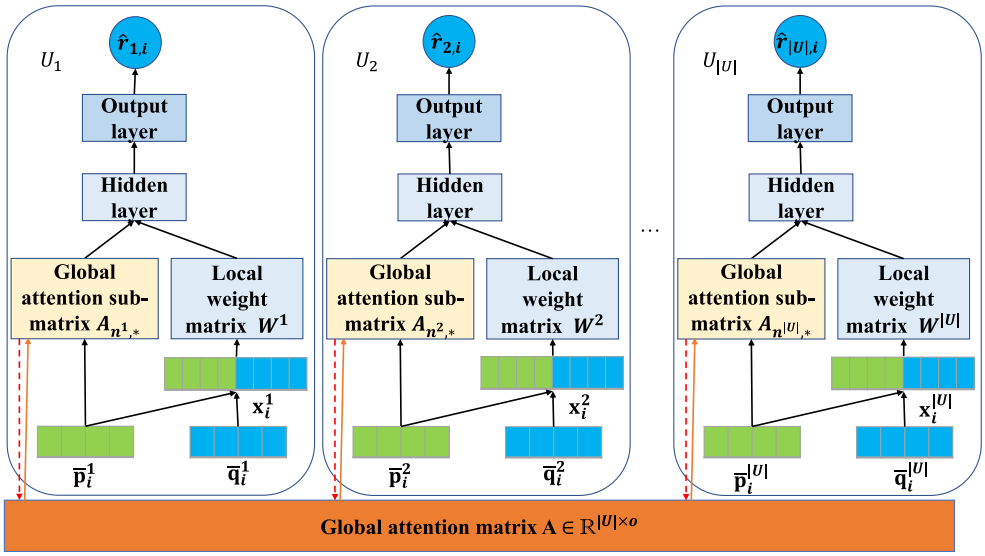


Fig. 2. The architecture of the BPAM++ model.

$$\mathbf{q}_i^u[j] = \begin{cases} \text{mean}(\mathbf{R}_{u,*}), & \text{if } \mathbf{q}_i^u[j] = 0 \\ \mathbf{q}_i^u[j], & \text{otherwise,} \end{cases} \quad \forall j = 1, \ldots, l, \tag{6}$$

where $\mathbf{R}_{u,*}$ denotes the ratings of user u, and $\text{mean}(\mathbf{R}_{u,*})$ denotes the mean value of the nonzero elements of $\mathbf{R}_{u,*}$. That is, the missing rating $\mathbf{q}_i^u[j]$ in the LNI rating vector is estimated by the mean rating value of user u.

In this way, for the given user-item pair composed of user u and item i, a training sample of the BP neural network can be obtained, i.e.,

$$\mathbf{d}_i^u = \begin{bmatrix} \mathbf{x}_i^u \\ r_{u,i} \end{bmatrix}, \tag{7}$$

where $\mathbf{x}_i^u = \begin{bmatrix} \mathbf{p}_i^u \\ \mathbf{q}_i^u \end{bmatrix} \in \mathbb{R}^{(k+l)\times 1}$ denotes the input vector fed into the BP neural network, which is aggregated by applying a concatenation operation on $\mathbf{p}_i^u$ and $\mathbf{q}_i^u$. In the training process, the error between the predicted rating $\hat{r}_{u,i}$ and the real rating $r_{u,i}$ will be fed back to adjust the parameters.

3.3 BP Neural Network With Attention Mechanism

The architecture of the proposed BP neural network with Attention Mechanism (BPAM++) model is illustrated in Fig. 2, where the input layer, hidden layer and output layer are the three layers in the BP neural network. By introducing the attention mechanism [62], the proposed BPAM++ model is able to capture not only the complex relationship

Authorized licensed use limited to: ANNA UNIVERSITY. Downloaded on February 07,2023 at 09:43:58 UTC from IEEE Xplore. Restrictions apply.

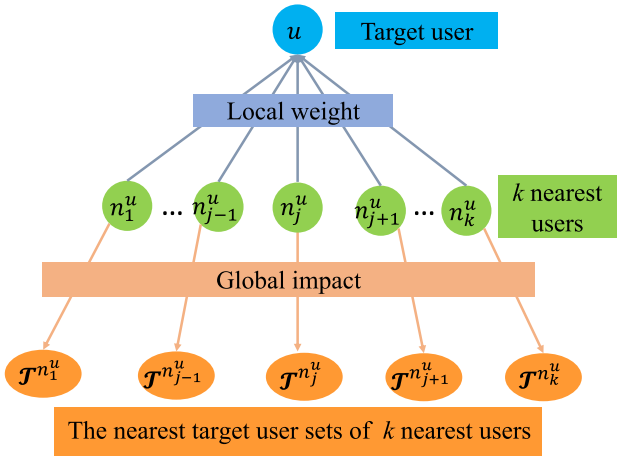


Fig. 3. Illustration of the nearest target user sets of k nearest users. For each nearest user $n_j^u \in \mathbf{n}^u$, we regard the weight of n_j^u on the target user u encoded by the BP neural network as the local weight, and the impact of n_j^u on the nearest target user set $\mathcal{T}^{n_j^u}$ encoded by the global attention matrix as the global impact.

between the target user and his/her neighbors and the complex relationship between the target item and its neighbors, but also the global impact of the nearest users of the target user on their nearest target user sets.

Definition 1 (Nearest target user set). *For each user $u \in \mathcal{U}$, the nearest target user set of user u, denoted as $\mathcal{T}^u$, is defined as*

$$\mathcal{T}^u = \{u' \in \mathcal{U} | u \in \mathbf{n}^{u'}\}. \tag{8}$$

That is, user u is one of the k nearest users of each user in $\mathcal{T}^u$.

Notice that the sizes of the nearest target user sets of different users may be different. Obviously, for each of the k nearest users of the target user u, i.e., $n_j^u, \forall j = 1, \ldots, k$, his/her nearest target user set contains user u, i.e., $u \in \mathcal{T}^{n_j^u}$, since $n_j^u \in \mathbf{n}^u$.

As shown in Fig. 3, when predicting the ratings of the target user u, for each nearest user $n_j^u \in \mathbf{n}^u$, the global impact on his/her nearest target user set is considered, which is defined and explained as follows.

Definition 2 (Global impact). *The global impact is defined to be the impact of n_j^u on the nearest target user set $\mathcal{T}^{n_j^u}$, which is captured by a global attention matrix.*

In particular, let $\mathbf{A} \in \mathbb{R}^{|\mathcal{U}| \times o}$ denote the global attention matrix for all users, where o is the number of neurons in the hidden layer. The uth row $\mathbf{A}_{u,*} \in \mathbb{R}^{1 \times o}$ of $\mathbf{A}$ denotes the global attention (impact) of user u on his/her nearest target user set $\mathcal{T}^u$.

For each target user u, let $\{i_1^u, i_2^u, \ldots, i_{\pi_u}^u\}$ denote the indexes of all the items that user u has rated, where π_u is the number of items that user u has rated. The training set for user u is composed of all the samples $\{\mathbf{d}_{i_1^u}^u, \mathbf{d}_{i_2^u}^u, \ldots, \mathbf{d}_{i_{\pi_u}^u}^u\}$ corresponding to the π_u items that user u has rated

$$\mathcal{D}^u = \left\{ \begin{bmatrix} \mathbf{x}_{i_1^u}^u \\ r_{u,i_1^u} \end{bmatrix}, \ldots, \begin{bmatrix} \mathbf{x}_{i_{\pi_u}^u}^u \\ r_{u,i_{\pi_u}^u} \end{bmatrix} \right\}. \tag{9}$$

The training process of BPAM++ is divided into two stages, namely forward propagation and error back propagation [63].

3.3.1 Forward Propagation

For any training sample $\mathbf{d}_i^u \in \mathcal{D}^u$, the forward propagation, i.e., the rating prediction process, can be defined as

$$\begin{aligned} \mathbf{h}_i^u &= \delta(\mathbf{W}^{u\top} \mathbf{x}_i^u + \alpha(\mathbf{A}_{\mathbf{n}^u,*})^\top \mathbf{p}_i^u + \mathbf{b}_{input}^u) \\ \hat{r}_{u,i} &= \delta(\mathbf{w}_{hidden}^{u\top} \mathbf{h}_i^u + b_{hidden}^u). \end{aligned} \tag{10}$$

In the above equation, the notations along with their underlying rationales are elaborated as follows.

- $\mathbf{h}_i^u \in \mathbb{R}^{o \times 1}$ denotes the values of the hidden layer.
- δ denotes the activation function $Sigmoid$.
- $\mathbf{W}^u \in \mathbb{R}^{(k+l) \times o}$ denotes the local weight matrix of user u in the BP neural network that is shared by all the user-item pairs containing user u. In particular, The local weight matrix is able to learn the complex relationship between the target user and his/her neighbors and the complex relationship between the target item and its neighbors.
- $\mathbf{A}_{\mathbf{n}^u,*} \in \mathbb{R}^{k \times o}$ denotes the global attention sub-matrix of $\mathbf{A}$ composed of the n_1^uth, n_2^uth, $\ldots$, n_k^uth rows of $\mathbf{A}$ corresponding to the k nearest users of user u. Each row of $\mathbf{A}_{\mathbf{n}^u,*}$ encodes the global attention (impact) of the corresponding nearest user of user u on his/her nearest target user set. And all the k rows of $\mathbf{A}_{\mathbf{n}^u,*}$ together encode the global attention of all the k nearest users of user u on their nearest target user sets.
- $\mathbf{b}_{input}^u \in \mathbb{R}^{o \times 1}$ denotes the bias vector among the input layer and hidden layer.
- $\mathbf{w}_{hidden}^u \in \mathbb{R}^{o \times 1}$ and $b_{hidden}^u \in \mathbb{R}^{1 \times 1}$ denote the weight vector and the bias value among the hidden layer and output layer.
- α is the trade-off parameter which is used to tune the importance of the global impact.

The objective function of the BPAM++ model is defined as follows:

$$E = \frac{1}{2 \times |\mathcal{U}|} \sum_{u=1}^{|\mathcal{U}|} E^u + \frac{\lambda}{2} ||\mathbf{A}||^2, \tag{11}$$

where

$$E^u = \frac{1}{\pi_u} \sum_{i=1}^{\pi_u} (\hat{r}_{u,i} - r_{u,i})^2 + \lambda(||\mathbf{w}_{hidden}^u||^2 + ||\mathbf{W}^u||^2), \tag{12}$$

and the regularization terms are added to prevent overfitting.

Before training the model, the model parameters $\Theta = \{\mathbf{A}, \{\mathbf{W}^u, \mathbf{b}_{input}^u, \mathbf{w}_{hidden}^u, b_{hidden}^u, \forall u \in \mathcal{U}\}\}$ are initialized randomly. And Stochastic Gradient Descent (SGD) is adopted to adjust the above model parameters in the direction of the negative gradient of the objective value.

3.3.2 Error Back Propagation

For any training sample $\mathbf{d}_i^u \in \mathcal{D}^u$, the error back propagation, i.e., parameter update, can be defined as

$$\begin{cases} \triangle \mathbf{A}_{\mathbf{n}^u,*} = -\eta\alpha \mathbf{p}_i^u \mathbf{e}_i^{u\top} + \lambda \mathbf{A}_{\mathbf{n}^u,*} \\ \triangle \mathbf{W}^u = -\eta \mathbf{x}_i^u \mathbf{e}_i^{u\top} + \lambda \mathbf{W}^u \\ \triangle \mathbf{b}_{input}^u = -\eta \mathbf{e}_i^u \\ \triangle \mathbf{w}_{hidden}^u = -\eta g_i^u \mathbf{h}_i^u + \lambda \mathbf{w}_{hidden}^u \\ \triangle b_{hidden}^u = -\eta g_i^u, \end{cases} \tag{13}$$

Authorized licensed use limited to: ANNA UNIVERSITY. Downloaded on February 07,2023 at 09:43:58 UTC from IEEE Xplore. Restrictions apply.

where $\eta \in (0,1)$ is the learning rate, and the scalar value g_i^u and the vector $\mathbf{e}_i^u \in \mathbb{R}^{o\times 1}$ are defined as

$$\begin{aligned} g_i^u &= \hat{r}_{u,i}(1-\hat{r}_{u,i})(\hat{r}_{u,i}-r_{u,i}) \\ \mathbf{e}_i^u &= \mathbf{h}_i^u \circ (\mathbf{I}-\mathbf{h}_i^u) \circ \mathbf{w}_{hidden}^u g_i^u, \end{aligned} \tag{14}$$

with $\circ$ denoting the Hadamard product, i.e., the product of the corresponding elements of the two vectors, and $\mathbf{I}$ denoting the vectors of all ones.

In summary, by introducing the attention mechanism, BPAM++ combines the local weight matrix and the global attention matrix to predict the missing ratings. The training process of BPAM++ is to alternately iterate the forward propagation and the error back propagation until the stopping condition is reached. The whole procedure of BPAM++ is summarized in Algorithm 1. Finally, the predicted ratings can be obtained via Eq. (10).

Algorithm 1. The Algorithm Framework of BPAM++

Input: Rating matrix: $\mathbf{R}$; number of nearest users: k; number of nearest items: l; trade-off parameter: α.

```
Randomly initialize model parameters:
  Θ = {A, {W^u, b^u_input, w^u_hidden, b^u_hidden, ∀u ∈ U}}.
Obtain the k nearest users (KNU) of each user u, i.e., the
  KNU index vector n^u = [n^u_1, n^u_2, ..., n^u_k].
Obtain the l nearest items (LNI) of each item i, i.e., the LNI
  index vector n^i = [n^i_1, n^i_2, ..., n^i_l].
repeat
  for all u ∈ U do
    Form the KNU rating vector p^u_i of user u on each item i.
    Form the LNI rating vector q^u_i of each item i from user u.
    Perform rating vector processing on p^u_i and q^u_i via
    Eqs. (5) and (6) respectively.
    Construct the training sample d^u_i for each item i via
    Eq. (7).
    for all d^u_i ∈ D^u do
      Predict the rating r̂_{u,i} via Eq. (10).
      Update A_{n^u,*}, W^u, b^u_input, w^u_hidden, b^u_hidden via Eqs. (13)
      and (14).
    end for
  end for
until Eq. (11) converges
```

Output: Model parameters: Θ.

3.4 Discussion

In our previous work [40], a special case of BPAM++, namely BPAM, was proposed. In what follows, we give the analysis of the two models, namely BPAM++ and BPAM.

- BPAM can be regarded as a special case of BPAM++. In particular, BPAM does not consider the ratings of the target user to the nearest items of the target item. Therefore, if only considering the ratings of the nearest users of the target user to the target item, BPAM++ is equivalent to BPAM. From the perspective of the model, if the input vector $\mathbf{x}_i^u$ is only composed of $\mathbf{p}_i^u$, then the BP neural network is only used to capture the complex relationship between the target users and their neighbors. We argue that capturing the complex relationship between the target items and their neighbors by means of considering the ratings of the target user to the nearest items of the target item has potential to improve the recommendation performance.
- BPAM++ and BPAM are general frameworks for recommendation. We can obtain the nearest neighbors of the users and items via different methods. For example, the similarities between users or items can also be obtained by means of computing the Pearson correlation coefficient [64] or integrating the users' social relationship [65] and content information of the items [7].
- We perform the time complexity analysis on the proposed BPAM++ and BPAM models. Generally, the time complexity is affected by the size of the BP neural networks, the epochs of iterator t, the number of training samples s. Since BPAM can be considered as one special case of BPAM++, we only need to preform the time complexity on BPAM++. As illustrated in Fig. 2, the input size and the output size between the input layer and the hidden layer of the BP neural network are $k+l$ and o respectively. The input size and the output size between the hidden layer and the output layer of the BP neural network are o and 1 respectively. When we learn the complex relationship between the neighbor ratings and target ratings in the proposed BPAM++ and BPAM models, the time complexity of forward propagation and error backward propagation is bounded by the matrix operation in the BP neural networks with attention mechanism. The time complexity of forward propagation can be calculated as $O((k+l+1)*o+k*o)$, where $k*o$ denotes the amount of computation in the attention mechanism. Therefore, the overall time complexity of BPAM++ is $O(t*s*(2*k+l+1)*o)$, which can be simplified to $O(t*s*(2*k+l)*o)$. On the other hand, the computation consumed by the DNNs-based methods is $O(t*s*(w_1*w_2+w_2*w_3\ldots+w_{h-1}*w_h))$, where w_h is the number of neurons in the hth layer. For example, in DeepCF [49], w_1 is equivalent to $|\mathcal{U}|+|\mathcal{V}|$, which are the number of users and items. As illustrated in Section 4, $(2*k+l) << (|\mathcal{U}|+|\mathcal{V}|)$. Therefore, the time complexity of BPAM++ is much smaller than DeepCF.

4 Experiments

4.1 Experimental Setting

4.1.1 Datasets

The experiments are conducted on eight real-world publicly available datasets obtained from the following four main sources.

1) MovieLens[2]: MovieLens datasets have been widely used for movie recommendation. These datasets are collected from the MovieLens website by the GroupLens Research. In our experiment, the ml-latest dataset (*abbr.* ml-la) is adopted.
2) Filmtrust[3]: The filmtrust dataset [66] is crawled from the entire Filmtrust website in June 2011, which contains 1,508 users, 2,071 items and 35,497 ratings.

2. https://grouplens.org/datasets/movielens/
3. https://www.librec.net/datasets.html

Authorized licensed use limited to: ANNA UNIVERSITY. Downloaded on February 07,2023 at 09:43:58 UTC from IEEE Xplore. Restrictions apply.

TABLE 2
Statistics of the Eight Datasets

Datasets	#Users	#Items	#Ratings	Sparsity	Scale
ml-la	610	9,724	100,836	98.30%	[0.5, 5]
filmtrust	1,508	2,071	35,497	98.86%	[0.5, 5]
jd-1	24,983	100	1,810,455	27.53%	[−10, 10]
jd-2	23,500	100	1,708,993	27.28%	[−10, 10]
jd-3	24,938	100	616,912	75.26%	[−10, 10]
Automotive	2,928	1,835	20,473	99.62%	[1, 5]
Beauty	22,362	12,101	198,502	99.93%	[1, 5]
Kindle Store	68,223	61,934	982,619	99.98%	[1, 5]

3) Jester[4]: The Jester dataset [67] is collected from the jester online Joke recommender system spanning April 1999 - May 2003, which contains over 4.1 million continuous ratings (-10.00 to $+10.00$) of 100 jokes from 73,421 users. In our experiment, three widely tested datasets, namely jester-data-1 (*abbr.* jd-1), jester-data-2 (*abbr.* jd-2), and jester-data-3 (*abbr.* jd-3), are adopted.
4) Amazon[5]: This dataset [68] is obtained from Julian McAuley, which contains users' rating data of 24 domains in Amazon spanning May 1996 - July 2014. In particular, three datasets, namely Automotive, Beauty, and Kindle Store, are adopted.

The statistics of these eight datasets are summarized in Table 2. Notice that in order to verify the accuracy and effectiveness of the proposed models in the case of different data sizes (measured by the number of users and items), experiments are carried out on both large-scale dataset, namely Kindle Store, and other small datasets, respectively. As illustrated in Table 2, the rating scales of the jester datasets (jd-1, jd-2, jd-3) are significantly different from those of other datasets. In order to better show the prediction effect, we quantify their scales to $[0, 5]$. We randomly split each dataset into the training set and testing set with ratio 3:1 for each user. We randomly select a quarter of the samples in the training set as the validation set to select the appropriate hyper-parameters.

4.1.2 Evaluation Measures

We utilize the root-mean-square error (*RMSE*) and mean-absolute-error (*MAE*) to evaluate the performance of the prediction results

$$\begin{aligned} RMSE &= \sqrt{\frac{1}{N}\sum_{u,i}\left(r_{u,i}-\hat{r}_{u,i}\right)^2} \\ MAE &= \frac{1}{N}\sum_{u,i}|r_{u,i}-\hat{r}_{u,i}|, \end{aligned} \tag{15}$$

where $\hat{r}_{u,i}$ denotes the predicted rating value, $r_{u,i}$ is the actual rating, and N denotes the number of tested ratings. Smaller values of RMSE and MAE indicate the better performance.

4.2 Comparison Results

We compare the proposed BPAM++ and BPAM methods with the following five state-of-the-art methods:

- *PMF*[6] [69] is a probabilistic algorithm that adopts a probabilistic linear model with Gaussian observation noise to model the user preference matrix as a product of lower-rank user matrix and item matrix.
- *SVD++*[7] [45] is an extension to SVD-based latent factor model by integrating the neighborhood models. Specifically, in the neighborhood models, a second set of item factors is added to model the item-item similarity.
- *NeuMF*[8] [34] combines Generalized Matrix Factorization (GMF) and Multi-Layer Perceptron (MLP) to learn the useritem interaction function.
- *DMF*[9] [32] utilizes deep neural network to learn a common low dimensional space for the representations of users and items. It uses a two-pathway neural network architecture to replace the linear embedding operation used in vanilla matrix factorization.
- *DeepCF*[10] [49] incorporates collaborative filtering methods based on representation learning and matching function learning to learn the complex matching function and low-rank relations between users and items.

All the baselines add the regularization term in their loss functions to overcome the overfitting issues. However, most DNN-based models have high complexity and massive parameters. If there are only a relatively small number of ratings, these DNNs-based models may suffer from overfitting. On the contrary, the proposed BPAM++ model first obtains the nearest neighbors of the given user and item, and then feeds the rating information of their nearest neighbors into the shallow BP neural network for rating prediction. Through the combination of the neighbor model and the shallow BP neural network, it can reduce the number of parameters in the DNNs-based models while ensuring the same nonlinear representation learning capability as the DNNs-based models. That is, we reduce the number of parameters by feeding only the rating information of nearest neighbors to the shallow BP neural network, thereby alleviating the overfitting issues caused by the relatively small number of ratings.

The parameters for the five state-of-the-art methods are tuned as suggested by the authors. For the PMF model, the numbers of the factors for users and items are both set to 30. The regularization parameters of users and items are set to 0.001 and 0.0001 respectively. For the SVD++ model, the numbers of the factors of users and items and the neighbor size are set to 30. We use the following values as suggested by the author for the meta parameters in the SVD++ model: $\gamma_1 = \gamma_2 = 0.007$, $\gamma_3 = 0.001$, $\gamma_6 = 0.005$, and $\gamma_7 = \gamma_8 = 0.015$. For the NeuMF model, the numbers of the predicted factors and the embedding sizes of users and items in the Generalized Matrix Factorization part are set to 8 and 16 respectively. For the DMF model, the numbers of the predicted factors for users and items are both set to 64, and the numbers of the first layer for users and items are set to 512 and

4. http://eigentaste.berkeley.edu/dataset/
5. http://jmcauley.ucsd.edu/data/amazon/
6. https://github.com/xuChenSJTU/PMF
7. https://github.com/hiro-00/recommender
8. https://github.com/ZJ-Tronker/Neural_Collaborative_Filtering-1
9. https://github.com/RuidongZ/Deep_Matrix_Factorization_Models
10. https://github.com/familyld/DeepCF

Authorized licensed use limited to: ANNA UNIVERSITY. Downloaded on February 07,2023 at 09:43:58 UTC from IEEE Xplore. Restrictions apply.

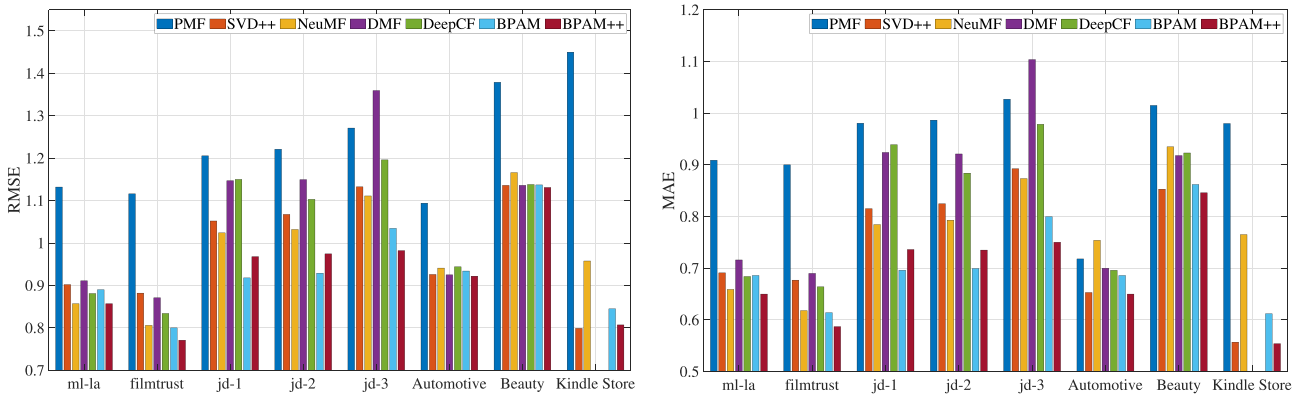


Fig. 4. Comparison results in terms of average RMSE and MAE over 5 runs obtained by different models on the eight datasets.

1024 respectively. For the DeepCF model, the sizes of the MLP layers in the representation learning part are set to $[512, 256, 128, 64]$, and in the matching function learning part, the same setting as DMF is used. For these deep models, namely NeuMF, DMF and DeepCF, the batch sizes are set to 256. For the BPAM++ and BPAM models, we tune the numbers of nearest users and items in [3, 5, 7, 9,11,13] and [3, 5, 7, 9] respectively. We test the attention ratio from 0 to 2 with an interval of 0.2. All the experiments are implemented by Python running on a server with Intel Xeon E5-2630 CPU, GeForce GTX 1080 Ti and 64 GB memory. For the DNNs-based methods, GPU is adopted, while for the other methods including BPAM++ and BPAM, GPU is not adopted.

4.2.1 Comparison on Accuracy

The accuracy comparison results in terms of RMSE and MAE are shown in Fig. 4. In this experiment, we run each algorithm on each dataset 5 times and report the average performance over 5 runs. When training the models on the largest dataset, i.e., Kindle Store, memory error occurs in the DMF and DeepCF models. Because they need to feed the user-item rating matrix into the MLP layers to get the embeddings of users and items, but for the Kindle Store dataset, the memory occupied by the entire rating matrix is $|\mathcal{U}| \cdot |\mathcal{V}| \cdot 32$ bits, which is approximately equal to 125 GB, far exceeding the machine's 64 GB memory. Although storing the user-item rating matrix via a sparse matrix can reduce the memory requirement, as implemented by the other methods, the sparse matrix can not be fed into the DMF and DeepCF models directly. Therefore, both DMF and DeepCF encounter memory error issues. We record their corresponding unpredictable results as "zero". That is, the DMF and DeepCF models are displayed without histograms on the Kindle Store dataset in Fig. 4. According to Fig. 4, we have the following key observations.

- First, PMF is a classical CF method only considering latent factors, but SVD++ combines the latent factor and neighbor models for rating prediction. The BP neural network can be thought as a version of a factorization-based approach, where factors are indirectly learned as part of the back propagation based on training process. Therefore, the proposed BPAM++ and BPAM models can be regarded as attempts combining the latent factor model and the neighbor model. As shown in Fig. 4, SVD++, BPAM++ and BPAM have achieved significant improvements over PMF on all the datasets in terms of RMSE and MAE, which indicates the effectiveness of combining the latent factor model and the neighbor model. In particular, the proposed BPAM++ model outperforms SVD++ on most of the datasets in terms of both RMSE and MAE. The only exception occurs on Kindle Store in terms of RMSE, where SVD++ performs slightly better than BPAM++. The reason is that, SVD++ only utilizes the matrix factorization to capture the linear relationship between users and items, but BPAM++ can better model the non-linear relationship between them via the BP neural network.
- Second, the proposed BPAM++ model outperforms the three deep neural network-based algorithms, namely NeuMF, DMF and DeepCF on all the datasets in terms of RMSE and MAE. Another observation is that on the Amazon datasets, namely Automotive, Beauty and Kindle Store, as the data size increases (from Automotive to Kindle Store), the proposed BPAM++ model still maintains the good performance. In particular, the DMF and DeepCF models fail to generate predicted ratings on Kindle Store due to memory error. And the performance of BPAM++ is much better than that of NeuMF on Kindle Store.
- Overall, the proposed BPAM++ or BPAM models outperform the other state-of-the-art methods on most of the datasets in terms of both RMSE and MAE. The results have confirmed the effectiveness of utilizing shallow BP neural network and attention mechanism in recommender systems.

4.2.2 Comparison on Computational and Storage Costs

In our experiments, we not only compare the accuracy achieved by the proposed models and the baselines, but also compare the computational and storage costs of these models. We regard the calculation amount required for predicting the rating of a user-item pair as the computational cost, and the amount of parameters needed to be stored as the storage cost. The comparison results in terms of computational and storage costs are shown in Tables 3 and 4 respectively.

From Table 3, we can observe that the computational costs of PMF, SVD++, BPAM, and BPAM++ models are significantly lower than the computational costs of these

Authorized licensed use limited to: ANNA UNIVERSITY. Downloaded on February 07,2023 at 09:43:58 UTC from IEEE Xplore. Restrictions apply.

TABLE 3
Comparison Results in Terms of the Computational Costs of Different Models in Milliseconds, i.e., the Calculation Amount Required for Predicting the Rating of a User-Item Pair

	PMF	SVD++	NeuMF	DMF	DeepCF	BPAM	BPAM++
Cost	31	150	2,736	98,432	270,464	162	204

DNNs-based models namely NeuMF, DMF and DeepCF. In the BPAM++ and BPAM models, for the same user, all items share a BP neural network. The number of parameters stored in the BP neural network and the attention vector corresponding to the given user is slightly more than the number of user latent factors in the PMF and SVD++ model.

From Table 4, we can find that the storage costs of the BPAM++ and BPAM models on all datasets are much smaller than the storage cost of the DNNs-based models. On the Jester dataset, since the number of items is only 100, the storage cost of BPAM++ and BPAM is slightly larger than that of the PMF and SVD++ models. On the other datasets, since all items of the same user share the same BP neural network, the storage costs of the BPAM++ and BPAM models are smaller than that of the PMF and SVD++ models. In summary, compared with DNNs, BPAM++ can reduce the computational and storage costs.

4.2.3 Comparison on Addressing Overfitting

As aforementioned, in the DNNs-based models, a relatively small number of ratings may cause the overfitting issue. In this part, we will analyze the performance of BPAM++ and the DNNs-based models under different sparsity. On each dataset, we divide users into different groups based on the number of their observed ratings. Due to the space limit, we only show the performance comparison of the BPAM++ and the DNNs-based models on the ml-la dataset. In Fig. 5, the horizontal axis shows the user group information. That is, [64, 128) means that for the users in this group, the numbers of their observed ratings are between 64 and 128. We can see that, as the sparsity increases, the gap between the performance of the DNNs-based models and BPAM++ gradually increases. In particular, the performance of BPAM++ and the DNNs-based models is close on the group [256, 2698), where 2,698 is the maximum number of a user's observed ratings. But on the group [0,64), BPAM++ has a significant performance improvement compared with the DNNs-based models.

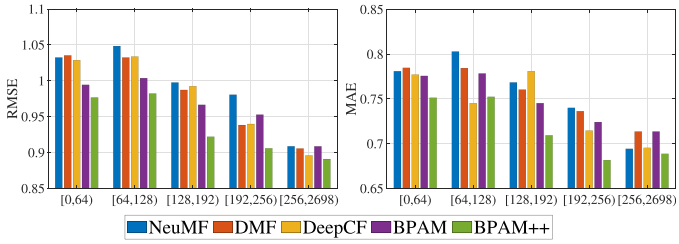


Fig. 5. Comparison results on addressing overfitting: The performance of BPAM++ and the DNNs-based models under different sparsity on the ml-la dataset.

4.3 Component Analysis

The difference between BPAM and BPAM++ is that BPAM does not consider the target user's ratings on the nearest items. In addition, in the design of BPAM++, we utilize rating vector processing to distinguish users' rating preferences (higher or lower) and introduce the global attention weights to prevent the model from local optimum. To validate the effectiveness of considering the target user's ratings on the nearest items, rating vector processing and attention mechanism in the proposed BPAM++ model, we compare BPAM++ with BPAM and the following three variants: BPCF-WP (without considering the target user's ratings on the nearest items, rating vector processing and attention mechanism), BPCF (without considering the target user's ratings on the nearest items and attention mechanism), BPAM-WP (without considering the target user's ratings on the nearest items and rating vector processing). As shown in Fig. 6, we have the following observations.

1) The methods with rating vector processing, i.e., BPAM and BPCF have achieved significantly better results than those methods without rating vector processing, i.e., BPAM-WP and BPCF-WP. It demonstrates the effectiveness of rating vector processing before the input vectors are fed into the BP neural network. Specifically, on the particularly sparse datasets, namely Automotive and Kindle Store, whose sparsity rates are 99.63 and 99.98 percent respectively, these methods with rating vector processing show a more noticeable performance improvement. It confirms that the proposed rating vector processing is able to well alleviate the issue of data sparsity in the recommender systems.
2) BPAM outperforms BPCF on most datasets in terms of RMSE and MAE, which validates that our attention mechanism in BPAM can effectively capture the

TABLE 4
Comparison Results in Terms of the Storage Costs of Different Models on the Eight Datasets, i.e., the Amount of Parameters Needed to be Stored

	ml-la	filmtrust	jd-1	jd-2	jd-3	Automotive	Beauty	Kindle Store
PMF	310,020	107,370	752,490	708,000	751,140	142,890	1,033,890	3,904,710
SVD++	612,074	173,079	780,573	734,600	779,178	202,703	1,431,383	5,892,887
NeuMF	829,312	288,912	2,009,232	1,890,592	2,005,632	383,632	2,759,632	10,415,152
DMF	5,701,632	2,702,848	25,732,096	24,213,504	25,686,016	4,036,096	29,192,704	101,668,864
DeepCF	8,519,168	3,791,104	32,325,376	30,427,136	32,267,776	5,427,456	38,187,264	135,161,088
BPAM	43,615	74,646	1,236,658	1,163,250	1,234,431	92,232	704,403	1,193,902
BPAM++	60,390	122,148	1,573,929	1,480,500	1,571,094	163,968	939,204	2,387,805

Authorized licensed use limited to: ANNA UNIVERSITY. Downloaded on February 07,2023 at 09:43:58 UTC from IEEE Xplore. Restrictions apply.

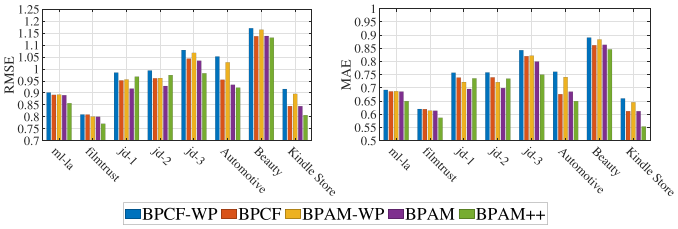


Fig. 6. Component analysis: Comparing BPAM++ with BPAM and three variants.

global attention weights about users' neighbors on their nearest target user set. It is worth pointing out that the performance of BPAM reported in Fig. 6 uses the default setting of the hyper-parameter α. Further improvements can be observed by tuning hyper-parameters, which will be reported in Section 4.4.

3) BPAM++ is inferior to BPAM on the jd-1 and jd-2 datasets but better than BPAM on the other six datasets, which indicates that the rating information on the nearest items of the target user can significantly improve the performance of recommendation. The possible reason why BPAM performs better than BPAM++ on jd-1 and jd-2 is that the rating matrices are relatively dense on the two datasets, resulting in that the ratings of nearest users are sufficient to predict the ratings of the target users. In addition, the number of items is relatively small and the similarity of the ratings between different items is poor. Then a certain amount of noise is introduced to the BPAM++ model, resulting in the relatively worse performance of BPAM++ compared with BPAM.

4.4 Sensitivity Analysis of Hyper-Parameters

In this section, we analyze the impact of the three hyper-parameters, namely the number k of user neighbors, the number l of item neighbors and the trade-off parameter α.

First, we take the values of k and l in range $[3, 5, 7, 9, 11, 13]$ and $[3, 5, 7, 9]$ respectively and report the results by the heat map. According to the results in Figs. 7 and 8, we can observe that setting the number k of user neighbors as 3 or 5 is not enough and larger number of user's neighbors is helpful. The best values of RMSE and MAE are obtained when setting $k = 9$ on filmtrust, $k = 7$ on Automotive and Beauty, and $k = 13$ on the other five datasets. However, we can observe that the best values of RMSE and MAE can be obtained when setting $l = 7$ on filmtrust and Automotive, $l = 9$ on ml-la and $l = 3$ on the other five datasets. Overall, the optimal values of k and l are around 7 to 13 and 3 to 9 respectively. Due to the small number of ratings per user,

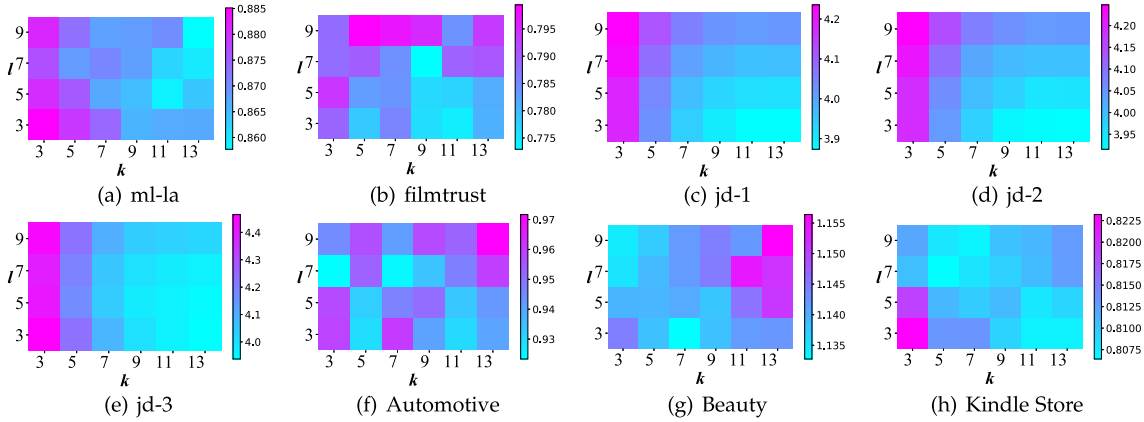


Fig. 7. Parameter analysis: The RMSE values obtained by BPAM++ with varying number k of users' neighbors and number l of items' neighbors.

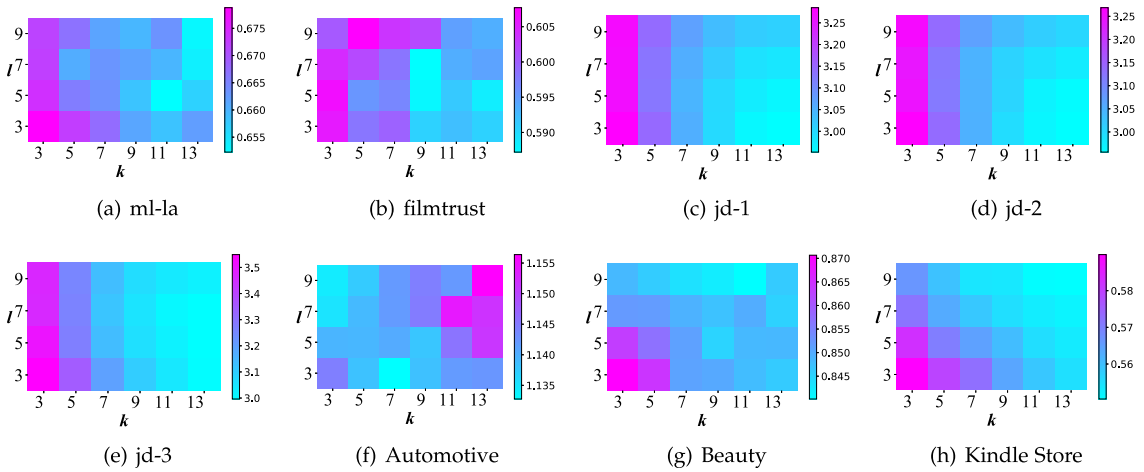


Fig. 8. Parameter analysis: The MAE values obtained by BPAM++ with varying number k of users' neighbors and number l of items' neighbors.

Authorized licensed use limited to: ANNA UNIVERSITY. Downloaded on February 07,2023 at 09:43:58 UTC from IEEE Xplore. Restrictions apply.

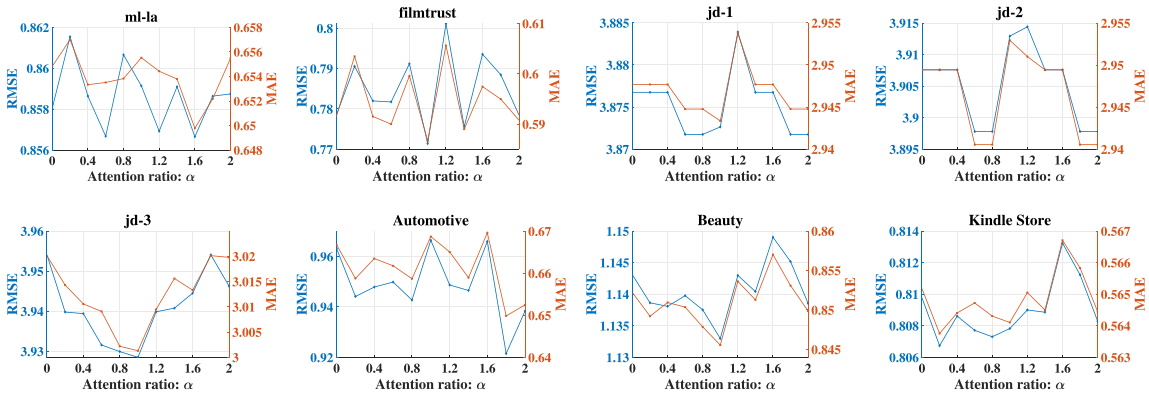


Fig. 9. Parameter analysis: The RMSE and MAE values obtained by BPAM++ with varying attention ratio α.

TABLE 5
Case Study: Some Sample Rating Records of User 16 and Several of his/her Nearest Target Users on the ml-la Dataset

Nearest target user	User	Movie	The rating of nearest target user	The rating of user
17	**16**	Movie 260: Star Wars: Episode IV - A New Hope (1977)	**5**	**3**
17	**16**	Movie 296: Pulp Fiction (1994)	**5**	**3**
131	**16**	Movie 3949: Requiem for a Dream (2000)	**2.5**	**4**
131	**16**	Movie 4878: Donnie Darko (2001)	**1.5**	**3.5**
137	**16**	Movie 858: Godfather, The (1972)	**5**	**2.5**
137	**16**	Movie 7153: Lord of the Rings: The Return of the King, The (2003)	**5**	**3.5**

too many nearest items may reduce the prediction performance of the proposed BPAM++ model.

Second, we study the impact of the trader-off parameter α and take the value in the range $[0, 0.2, 0.4, \ldots, 2]$. The results are plotted in Fig. 9. Specifically, if we set α to 0, the global impact of the neighbors of the target user will not be considered in the proposed BPAM++ model. Although BPAM++ achieves the best results in terms of RMSE and MAE when taking various values of α on various datasets, they have a significant improvement over the performance when α is set to 0. It demonstrates that the proposed attention mechanism improves the performance of the proposed BPAM++ model.

4.5 Case Study

For illustration purpose, we will conduct a case study on the ml-la dataset. Some sample rating records of user 16 and his/her several nearest target users in the training set are shown in Table 5. From this example, we can find that the ratings of user 16 on these movies are significantly different from the ratings of his/her nearest target users. That is, user 16 can not accurately reflect the preferences of his/her nearest target user set. Taking the nearest target user 131 as an example, we analyze the impact of user 16 in the BP neural network corresponding to the nearest target user 131 and the global impact captured by global attention to illustrate the effectiveness of the attention mechanism. As shown in Fig. 10, user 16 obtains a final weight of 0.5794 in the BP neural network, and a final weight of -0.2985 through the global attention matrix. Since the global attention weight can capture these rating information in Table 5, it can reduce the impact of user 16 on his/her nearest target user set. Finally, combined with the global impact captured by the global attention matrix, the impact of user 16 on the nearest target user 131 can be reduced, thereby achieving more accurate prediction. In summary, the reason why the attention mechanism is effective is that it can share all the rating information of the nearest target user set and encode the captured global impact in the global attention matrix, and therefore learn more accurate weights of the users *w.r.t.* their nearest target user sets.

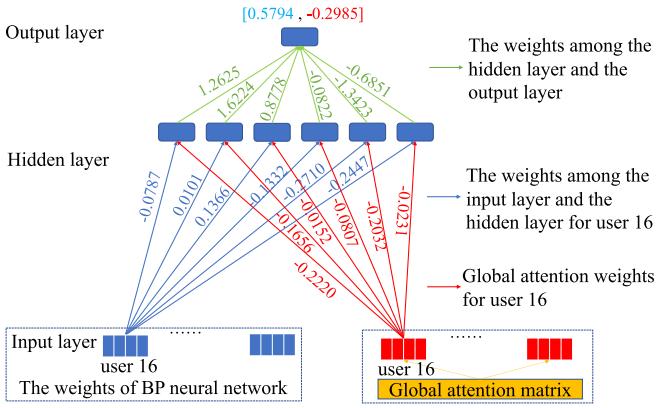


Fig. 10. Case study: Illustration of the impact of user 16 in the BP neural network on the nearest target user 131 and the global impact captured by global attention.

5 Conclusion

In this paper, we have proposed a novel recommender framework based on BP neural network with attention mechanism, called BPAM++. In the proposed BPAM++ framework, the BP neural network is utilized to learn the complex relationship between the target users and their neighbors and the complex relationship between the given items and their neighbors. In this way, BPAM++ outperforms SVD++,

Authorized licensed use limited to: ANNA UNIVERSITY. Downloaded on February 07,2023 at 09:43:58 UTC from IEEE Xplore. Restrictions apply.

which only utilizes the traditional matrix factorization and neighbor models. Compared with DNNs, shallow BP neural network can not only reduce the computational and storage costs, but also prevent the model from overfitting caused by the small number of ratings. Besides, an attention mechanism is introduced in BPAM++ to capture the global attention weights about users' impact on their nearest target user set. We also compare BPAM++ with our previously proposed BPAM model [40], in which the rating information of the target user is not considered. In particular, BPAM can be regarded as a special case of BPAM++. Extensive experiments on eight benchmark datasets demonstrate that our proposed model distinctly outperforms state-of-the-art methods.

In this paper, we mainly focus on the rating matrix in the recommender systems. In the future, we aim to introduce content information of users and items or social relationship of users to calculate similarities between users and items instead of only utilizing the rating vectors of users and items, so that we can obtain their more accurate nearest neighbors. Richer information usually leads to better performance. Moreover, except for the simple concatenation, it is also encouraged to explore other aggregation methods to construct the input vector.

Acknowledgments

This work was supported by NSFC (61876193) and Guangdong Natural Science Funds for Distinguished Young Scholar (2016A030306014).

References

[1] C.-D. Wang, Z.-H. Deng, J.-H. Lai, and P. S. Yu, "Serendipitous recommendation in E-commerce using innovator-based collaborative filtering," *IEEE Trans. Cybern.*, vol. 49, no. 7, pp. 2678–2692, Jul. 2019.

[2] J. Ma, J. Wen, M. Zhong, W. Chen, and X. Li, "MMM: Multi-source multi-net micro-video recommendation with clustered hidden item representation learning," *Data Sci. Eng.*, vol. 4, no. 3, pp. 240–253, 2019.

[3] Q.-Y. Hu, L. Huang, C.-D. Wang, and H.-Y. Chao, "Item orientated recommendation by multi-view intact space learning with overlapping," *Knowl.-Based Syst.*, vol. 164, pp. 358–370, 2019.

[4] P. Hu, R. Du, Y. Hu, and N. Li, "Hybrid item-item recommendation via semi-parametric embedding," in *Proc. Int. Joint Conf. Artif. Intell.*, 2019, pp. 2521–2527.

[5] C.-C. Hsu, M.-Y. Yeh, and S.-D. Lin, "A general framework for implicit and explicit social recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 12, pp. 2228–2241, Dec. 2018.

[6] C. Xu *et al.*, "Recurrent convolutional neural network for sequential recommendation," in *Proc. Int. Conf. World Wide Web*, 2019, pp. 3398–3404.

[7] O. Barkan, N. Koenigstein, E. Yogev, and O. Katz, "CB2CF: A neural multiview content-to-collaborative filtering model for completely cold item recommendations," in *Proc. 13th ACM Conf. Recommender Syst.*, 2019, pp. 228–236.

[8] L. Huang, C.-D. Wang, H.-Y. Chao, J.-H. Lai, and P. S. Yu, "A score prediction approach for optional course recommendation via cross-user-domain collaborative filtering," *IEEE Access*, vol. 7, pp. 19 550–19 563, 2019.

[9] W. Fu, Z. Peng, S. Wang, Y. Xu, and J. Li, "Deeply fusing reviews and contents for cold start users in cross-domain recommendation systems," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 94–101.

[10] A. Ferraro, "Music cold-start and long-tail recommendation: Bias in deep representations," in *Proc. 13th ACM Conf. Recommender Syst.*, 2019, pp. 586–590.

[11] J. Chen *et al.*, "Co-purchaser recommendation for online group buying," *Data Sci. Eng.*, vol. 5, no. 3, pp. 280–292, 2020.

[12] X. He, X. Du, X. Wang, F. Tian, J. Tang, and T.-S. Chua, "Outer product-based neural collaborative filtering," in *Proc. Int. Joint Conf. Artif. Intell.*, 2018, pp. 2227–2233.

[13] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl, "An algorithmic framework for performing collaborative filtering," *ACM SIGIR Forum*, vol. 51, no. 2, pp. 227–234, 2017.

[14] J. B. Schafer, D. Frankowski, J. L. Herlocker, and S. Sen, "Collaborative filtering recommender systems," in *The Adaptive Web: Methods and Strategies of Web Personalization*. Berlin, Germany: Springer, 2007, pp. 291–324.

[15] D.-K. Chae, J.-S. Kang, S.-W. Kim, and J. Choi, "Rating augmentation with generative adversarial networks towards accurate collaborative filtering," in *Proc. Int. Conf. World Wide Web*, 2019, pp. 2616–2622.

[16] H. Shin, S. Kim, J. Shin, and X. Xiao, "Privacy enhanced matrix factorization for recommendation with local differential privacy," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 9, pp. 1770–1782, Sep. 2018.

[17] R. Du, J. Lu, and H. Cai, "Double regularization matrix factorization recommendation algorithm," *IEEE Access*, vol. 7, pp. 139 668–139 677, 2019.

[18] C. Chen, Z. Liu, P. Zhao, J. Zhou, and X. Li, "Privacy preserving point-of-interest recommendation using decentralized matrix factorization," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 257–264.

[19] M. Vlachos, C. Dünner, R. Heckel, V. G. Vassiliadis, T. P. Parnell, and K. Atasu, "Addressing interpretability and cold-start in matrix factorization for recommender systems," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 7, pp. 1253–1266, Jul. 2019.

[20] Y. He, C. Wang, and C. Jiang, "Correlated matrix factorization for recommendation with implicit feedback," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 3, pp. 451–464, Mar. 2019.

[21] D. Valcarce, A. Landin, J. Parapar, and Á. Barreiro, "Collaborative filtering embeddings for memory-based recommender systems," *Eng. Appl. AI*, vol. 85, pp. 347–356, 2019.

[22] M. Gao, B. Ling, L. Yang, J. Wen, Q. Xiong, and S. Li, "From similarity perspective: A robust collaborative filtering approach for service recommendations," *Front. Comput. Sci.*, vol. 13, no. 2, pp. 231–246, 2019.

[23] Y. Gao and L. Ran, "Collaborative filtering recommendation algorithm for heterogeneous data mining in the internet of things," *IEEE Access*, vol. 7, pp. 123 583–123 591, 2019.

[24] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: Item-to-item collaborative filtering," *IEEE Internet Comput.*, vol. 7, no. 1, pp. 76–80, Jan./Feb. 2003.

[25] T. Hofmann, "Latent semantic models for collaborative filtering," *ACM Trans. Inf. Syst.*, vol. 22, no. 1, pp. 89–115, 2004.

[26] S. Zheng, C. H. Q. Ding, and F. Nie, "Regularized singular value decomposition and application to recommender system," 2018, *arXiv:1804.05090*.

[27] D. Agarwal and B.-C. Chen, "Regression-based latent factor models," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2009, pp. 19–28.

[28] Y. Jhamb and Y. Fang, "A dual-perspective latent factor model for group-aware social event recommendation," *Inf. Process. Manage.*, vol. 53, no. 3, pp. 559–576, 2017.

[29] L. Hu, L. Cao, J. Cao, Z. Gu, G. Xu, and D. Yang, "Learning informative priors from heterogeneous domains to improve recommendation in cold-start user domains," *ACM Trans. Inf. Syst.*, vol. 35, pp. 13:1–13:37, 2016.

[30] J. Wang, A. P. de Vries, and M. J. T. Reinders, "Unifying user-based and item-based collaborative filtering approaches by similarity fusion," in *Proc. 29th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2006, pp. 501–508.

[31] C. Shi, B. Hu, W. X. Zhao, and P. S. Yu, "Heterogeneous information network embedding for recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 2, pp. 357–370, Feb. 2019.

[32] H.-J. Xue, X. Dai, J. Zhang, S. Huang, and J. Chen, "Deep matrix factorization models for recommender systems," in *Proc. Int. Joint Conf. Artif. Intell.*, 2017, pp. 3203–3209.

[33] W. Yan, D. Wang, M. Cao, and J. Liu, "Deep auto encoder model with convolutional text networks for video recommendation," *IEEE Access*, vol. 7, pp. 40 333–40 346, 2019.

[34] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proc. Int. Conf. World Wide Web*, 2017, pp. 173–182.

[35] H. Liu *et al.*, "NRPA: Neural recommendation with personalized attention," in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2019, pp. 1233–1236.

Authorized licensed use limited to: ANNA UNIVERSITY. Downloaded on February 07,2023 at 09:43:58 UTC from IEEE Xplore. Restrictions apply.

[36] S. Wang, L. Hu, L. Cao, X. Huang, D. Lian, and W. Liu, "Attention-based transactional context embedding for next-item recommendation," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 2532–2539.
[37] A. T. Goh, "Back-propagation neural networks for modeling complex systems," *Artif. Intell. Eng.*, vol. 9, no. 3, pp. 143–151, 1995.
[38] K. Xu *et al.*, "Show, attend and tell: Neural image caption generation with visual attention," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 2048–2057.
[39] S. Wang, J. Zhang, and C. Zong, "Learning sentence representation with guidance of human attention," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, 2017, pp. 4137–4143.
[40] W.-D. Xi, L. Huang, C.-D. Wang, Y.-Y. Zheng, and J. Lai, "BPAM: Recommendation based on BP neural network with attention mechanism," in *Proc. Int. Joint Conf. Artif. Intell.*, 2019, pp. 3905–3911.
[41] R. M. Bell and Y. Koren, "Scalable collaborative filtering with jointly derived neighborhood interpolation weights," in *Proc. 7th IEEE Int. Conf. Data Mining*, 2007, pp. 43–52.
[42] J. Rongfei, J. Maozhong, and L. Chao, "Using temporal information to improve predictive accuracy of collaborative filtering algorithms," in *Proc. 12th Int. Asia-Pacific Web Conf.*, 2010, pp. 301–306.
[43] B. K. Patra, R. Launonen, V. Ollikainen, and S. Nandi, "A new similarity measure using Bhattacharyya coefficient for collaborative filtering in sparse data," *Knowl.-Based Syst.*, vol. 82, pp. 163–177, 2015.
[44] C. C. Krueger, "The impact of the Internet on business model evolution within the news and music sectors," PhD Thesis, pp. 1–397, 2006.
[45] Y. Koren, "Factorization meets the neighborhood: A multifaceted collaborative filtering model," in *Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2008, pp. 426–434.
[46] H. Yan and Y. Tang, "Collaborative filtering based on gaussian mixture model and improved Jaccard similarity," *IEEE Access*, vol. 7, pp. 118 690–118 701, 2019.
[47] H. Wang, N. Wang, and D.-Y. Yeung, "Collaborative deep learning for recommender systems," in *Proc. 21st ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2015, pp. 1235–1244.
[48] A. van den Oord, S. Dieleman, and B. Schrauwen, "Deep content-based music recommendation," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2013, pp. 2643–2651.
[49] Z.-H. Deng, L. Huang, C.-D. Wang, J.-H. Lai, and P. S. Yu, "DeepCF: A unified framework of representation learning and matching function learning in recommender system," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 61–68.
[50] S. Li, J. Kawale, and Y. Fu, "Deep collaborative filtering via marginalized denoising auto-encoder," in *Proc. ACM Int. Conf. Inf. Knowl. Manage.*, 2015, pp. 811–820.
[51] S. Sedhain, A. K. Menon, S. Sanner, and L. Xie, "AutoRec: Autoencoders meet collaborative filtering," in *Proc. Int. Conf. World Wide Web*, 2015, pp. 111–112.
[52] W. Fan *et al.*, "Graph neural networks for social recommendation," in *Proc. Int. Conf. World Wide Web*, 2019, pp. 417–426.
[53] Z. Cheng, Y. Ding, X. He, L. Zhu, X. Song, and M. S. Kankanhalli, "A3NCF: An adaptive aspect attention model for rating prediction," in *Proc. Int. Joint Conf. Artif. Intell.*, 2018, pp. 3748–3754.
[54] Y. Gong and Q. Zhang, "Hashtag recommendation using attention-based convolutional neural network," in *Proc. Int. Joint Conf. Artif. Intell.*, 2016, pp. 2782–2788.
[55] J. Xiao, H. Ye, X. He, H. Zhang, F. Wu, and T.-S. Chua, "Attentional factorization machines: Learning the weight of feature interactions via attention networks," in *Proc. Int. Joint Conf. Artif. Intell.*, 2017, pp. 3119–3125.
[56] C. Chen, M. Zhang, Y. Liu, and S. Ma, "Neural attentional rating regression with review-level explanations," in *Proc. Int. Conf. World Wide Web*, 2018, pp. 1583–1592.
[57] L. V. Tran, T.-A. N. Pham, Y. Tay, Y. Liu, G. Cong, and X. Li, "Interact and decide: Medley of sub-attention networks for effective group recommendation," in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2019, pp. 255–264.
[58] C. Zhou *et al.*, "ATRank: An attention-based user behavior modeling framework for recommendation," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 4564–4571.
[59] J. Zhu *et al.*, "Broad learning based multi-source collaborative recommendation," in *Proc. ACM Int. Conf. Inf. Knowl. Manage.*, 2017, pp. 1409–1418.
[60] P. Li, Z. Wang, Z. Ren, L. Bing, and W. Lam, "Neural rating regression with abstractive tips generation for recommendation," in *Proc. 40th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2017, pp. 345–354.
[61] L. Jing, P. Wang, and L. Yang, "Sparse probabilistic matrix factorization by laplace distribution for collaborative filtering," in *Proc. Int. Joint Conf. Artif. Intell.*, 2015, pp. 1771–1777.
[62] J. Chen, H. Zhang, X. He, L. Nie, W. Liu, and T.-S. Chua, "Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention," in *Proc. 40th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2017, pp. 335–344.
[63] J. Li, J.-H. Cheng, J.-Y. Shi, and F. Huang, "Brief introduction of back propagation (BP) neural network algorithm and its improvement," in *Proc. Advances Comput. Sci. Inf. Eng.*, 2012, pp. 553–558.
[64] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "GroupLens: An open architecture for collaborative filtering of netnews," in *Proc. ACM Conf. Comput. Supported Cooperative Work*, 1994, pp. 175–186.
[65] Y.-M. Li, C.-T. Wu, and C.-Y. Lai, "A social recommender mechanism for e-commerce: Combining similarity, trust, and relationship," *Decis. Support Syst.*, vol. 55, no. 3, pp. 740–752, 2013.
[66] G. Guo, J. Zhang, and N. Yorke-Smith, "A novel Bayesian similarity measure for recommender systems," in *Proc. Int. Joint Conf. Artif. Intell.*, 2013, pp. 2619–2625.
[67] K. Y. Goldberg, T. Roeder, D. Gupta, and C. Perkins, "Eigentaste: A constant time collaborative filtering algorithm," *Inf. Retrieval*, vol. 4, no. 2, pp. 133–151, 2001.
[68] R. He and J. J. McAuley, "Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering," in *Proc. 25th Int. Conf. World Wide Web*, 2016, pp. 507–517.
[69] A. Mnih and R. R. Salakhutdinov, "Probabilistic matrix factorization," in *Proc. 20th Int. Conf. Neural Inf. Process. Syst.*, 2008, pp. 1257–1264.

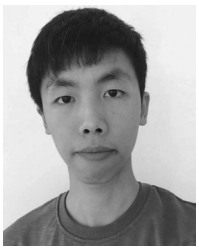
Chang-Dong Wang (Member, IEEE) received the PhD degree in computer science from Sun Yat-sen University, Guangzhou, China, in 2013. He is a visiting student with the University of Illinois at Chicago from January 2012 to November 2012. He joined Sun Yat-sen University in 2013, where he is currently an associate professor with the School of Data and Computer Science. His current research interests include machine learning and data mining. He has published more than 70 scientific papers in international journals and conferences such as the *IEEE Transactions on Pattern Analysis and Machine Intelligence*, the *IEEE Transactions on Knowledge and Data Engineering*, the *IEEE Transactions on Cybernetics*, the *IEEE Transactions on Neural Networks and Learning Systems*, KDD, AAAI, and IJCAI. His ICDM 2010 paper won the Honorable Mention for best research paper awards. He was awarded 2015 Chinese Association for Artificial Intelligence (CAAI) Outstanding Dissertation. He is an associate editor of the *Journal of Artificial Intelligence Research*.

Wu-Dong Xi received the BS degree in computer science from Sun Yat-sen University, Guangzhou, China, in 2019. He is currently working toward the master's degree in computer science at Sun Yat-sen University, Guangzhou, China. He has published one paper in IJCAI 2019. His research interest is data mining.

Ling Huang (Member, IEEE) received the PhD degree in computer science from Sun Yat-sen University, Guangzhou, China, in 2020. She joined South China Agricultural University, in 2020 as an associate professor. She has published more than 10 papers in international journals and conferences such as the *IEEE Transactions on Knowledge and Data Engineering*, the *IEEE Transactions on Cybernetics*, the *IEEE Transactions on Neural Networks and Learning Systems*, *IEEE Transactions on Industrial Informatics*, the *ACM Transactions on Knowledge Discovery from Data*, the *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, the *Pattern Recognition*, KDD, AAAI, IJCAI, and ICDM. Her research interest is data mining.

Authorized licensed use limited to: ANNA UNIVERSITY. Downloaded on February 07,2023 at 09:43:58 UTC from IEEE Xplore. Restrictions apply.

Yin-Yu Zheng received the BS degree in traffic engineering from Sun Yat-sen University, Guangzhou, China, in 2019. She is currently working toward the master's degree in management engineering and science at Sun Yat-sen University, Guangzhou, China. She has published one paper in IJCAI 2019. Her research interest is data mining.



Zi-Yuan Hu is currently working toward the undergraduate degree. He is good at algorithm design and implementation. His research interest is data mining.

Jian-Huang Lai (Senior Member, IEEE) received the MSc degree in applied mathematics and the PhD degree in mathematics from Sun Yat-sen University, Guangzhou, China, in 1989 and 1999. He joined Sun Yat-sen University in 1989 as an assistant professor, where he is currently a professor with the School of Data and Computer Science. His current research interests include the areas of digital image processing, pattern recognition, multimedia communication, wavelet and its applications. He has published more than 200 scientific papers in the international journals and conferences on image processing and pattern recognition, such as the *IEEE Transactions on Pattern Analysis and Machine Intelligence*, the *IEEE Transactions on Knowledge and Data Engineering*, the *IEEE Transactions on Neural Networks and Learning Systems*, the *IEEE Transactions on Image Processing*, the *IEEE Transactions on Systems, Man, and Cybernetics Part B: Cybernetics*, the *Pattern Recognition*, ICCV, CVPR, IJCAI, ICDM, and SDM. He serves as a standing member of the Image and Graphics Association of China, and also serves as a standing director of the Image and Graphics Association of Guangdong.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/csdl.

Authorized licensed use limited to: ANNA UNIVERSITY. Downloaded on February 07,2023 at 09:43:58 UTC from IEEE Xplore. Restrictions apply.