BPAM: Recommendation Based on BP Neural Network with Attention Mechanism

Wu-Dong Xi[1,2], **Ling Huang**[1,2], **Chang-Dong Wang**[1,2], **Yin-Yu Zheng**[1] and **Jianhuang Lai**[1]
[1]School of Data and Computer Science, Sun Yat-sen University, Guangzhou, 510006, China
[2]Guangdong Province Key Laboratory of Computational Science, Guangzhou, 510275, China
m13719336821@163.com, huanglinghl@hotmail.com, changdongwang@hotmail.com,
zhengyy.sysu@foxmail.com, stsljh@mail.sysu.edu.cn

Abstract

Inspired by the significant success of deep learning, some attempts have been made to introduce deep neural networks (DNNs) in recommendation systems to learn users' preferences for items. Since DNNs are well suitable for representation learning, they enable recommendation systems to generate more accurate prediction. 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 lead to over-fitting. To tackle these problems, we propose a novel recommendation algorithm based on Back Propagation (BP) neural network with Attention Mechanism (BPAM). In particular, the BP neural network is utilized to learn the complex relationship of the target users and their neighbors. Compared with deep neural network, the shallow neural network, i.e., BP neural network, can not only reduce the computational and storage costs, but also prevent the model from over-fitting. In addition, an attention mechanism is designed to capture the global impact on all nearest target users for each user. Extensive experiments on eight benchmark datasets have been conducted to evaluate the effectiveness of the proposed model.

1 Introduction

The era of information explosion has arrived, people can hardly hit what they really prefer when dealing with a huge number of choices. To tackle this problem, personalized recommendation systems have been proposed and widely used in e-commerce platforms and news/music/movie/education platforms [Hu *et al.*, 2019; Wang *et al.*, 2018a; Huang *et al.*, 2019]. Collaborative Filtering (CF) is one of most classical technologies in personalized recommendation systems, which infers users' preferences from historical behavior [He *et al.*, 2018].

The traditional CF recommendation algorithms are generally divided into two categories: Matrix Factorization (MF) and neighborhood-based CF methods. The MF methods map the users and items into a common representation space. Then the users' ratings to items are modeled as the inner product of their latent vectors. However, the MF methods easily suffer from the sparsity issue due to the long-tailed distribution of rating data in the real-world applications [Hu *et al.*, 2016]. On the other hand, the neighborhood-based CF algorithms predict the target ratings by averaging (weighted) ratings of similar entities (users or items). In [Wang *et al.*, 2006], 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 [Xue *et al.*, 2017; Deng *et al.*, 2019; He *et al.*, 2017]. In [Xue *et al.*, 2017], a novel deep matrix factorization 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 data [Wang *et al.*, 2018b]. Besides, 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 recommendation systems, which can easily lead to over-fitting of DNNs with massive parameters.

This paper addresses the above issues by proposing a recommendation algorithm based on Back Propagation (BP) neural network [Goh, 1995] with Attention Mechanism [Wang *et al.*, 2016] (BPAM). By introducing the BP neural network into the neighborhood-based CF algorithm, the ratings of similar users are fed into the BP neural network instead of undertaking a linear combination in the traditional algorithms. In this manner, BPAM is able to capture the non-linear relationship between the target user and his/her neighbors. 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 can be eliminated by selecting similar users' ratings. Moreover, since the BP neural network has a relatively small number of parameters, the over-fitting issues suffered by DNNs can be well avoided 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 target user's neighbors by means of introducing their global weights. In this way, a unified training model is constructed, which consists of the local weight and the global attention weight. When predicting the ratings of the target user, BPAM takes into account both local weight and global weight of the target user's neighbors to achieve global optimum.

The main contributions of this work are as follows.

- A novel neighborhood-based CF recommendation algorithm called BPAM is proposed, which overcomes the high computational and storage costs and over-fitting issues in DNNs.
- The BP neural network is utilized to learn the complex relationship between the target user and his/her neighbors instead of undertaking a linear combination in the traditional algorithms.
- A novel attention mechanism is introduced to capture the global impact of the target user's neighbors by means of introducing their global weights.
- 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.

2 Related Work

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 [Bell and Koren, 2007; Jia *et al.*, 2010; Patra *et al.*, 2015; Hu *et al.*, 2019]. In [Bell and Koren, 2007], 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 [Jia *et al.*, 2010], the temporal information was utilized to improve the accuracy of CF algorithms. In [Patra *et al.*, 2015], a similarity measure was proposed for neighborhood-based CF, which utilized all ratings information comprehensively for locating useful neighbors of an active user in the sparse rating matrix, and did not depend on co-rated items. However, these algorithms only achieve some performance improvement, since they still utilize the linear combination of similar entities' ratings to predict the target rating.

Recently, due to the representation learning abilities, DNNs have been introduced in recommender systems to learn users' preference for items [Xue *et al.*, 2017; Wang *et al.*, 2015; He *et al.*, 2017; van den Oord *et al.*, 2013]. DNNs are utilized to learn the complex mapping relationship between user-item latent factor representation and matching rating. For example, in [Xue *et al.*, 2017], a deep learning architecture was presented to learn a common low dimensional space for the representations of users and items, where a two-pathway neural network architecture was used to replace the linear embedding operation. In [Wang *et al.*, 2015], 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 [He *et al.*, 2017], 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. These DNNs-based models have greatly improved the prediction accuracy, but they suffer from the issues of overfitting and high computational and storage costs.

Attention-based architectures, which learn to focus their "attention" to specific parts [Cheng *et al.*, 2018] or combine both local and global information [Gong and Zhang, 2016], have shown great potential in recommendation algorithms. For instance, in [Cheng *et al.*, 2018], the attention mechanism was introduced to capture the varying attention vectors of each specific user-item pair. In [Gong and Zhang, 2016], 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. Motivated by the successes of various models, we adopt the attention mechanism to combine the local weights and global attention weights of neighbors.

3 The Proposed Model

3.1 Preliminaries

Suppose that $\mathcal{U}$ and $\mathcal{V}$ are the user set and the item set respectively, following [Zhu *et al.*, 2017], a user-item rating matrix $\mathbf{R} \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{V}|}$ is constructed from users' explicit feedback as follows,

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

where $r_{u,i}$ represents the rating of user u to item i.

The task of recommendation algorithms is to estimate the missing ratings in the rating matrix $\mathbf{R}$ [Li *et al.*, 2017]. The model-based methods generally assume that the models generate data in such a way as $\hat{y}_{u,i} = f(u, i|\Theta)$, where $\hat{y}_{u,i}$ denotes the prediction of $y_{u,i}$, i.e. the predicted rating of user u to item i, and f denotes the mapping function that maps the model input, e.g. the neighborhoods' ratings, to the predicted rating of the corresponding user-item pair by utilizing the model parameters Θ [Deng *et al.*, 2019]. 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. These simple linear mapping functions are usually hard to make accurate rating prediction. On the other hand, the mapping functions of DNNs-based CF are DNNs, which are used to learn the complex mapping relationship between user-item latent factor representation and matching rating. However, due to the sparsity issue in recommendation system, there are only a relatively small number of ratings, which are fed into DNNs with large number of parameters, leading to overfitting. Moreover, the training process of DNNs usually results in high computational and storage costs. In this paper, we utilize the BP neural network as mapping function to learn

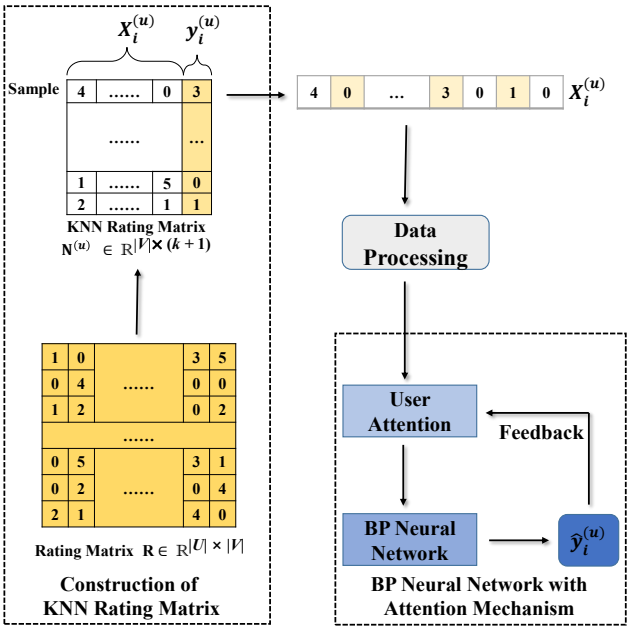

Figure 1: The general process for BPAM

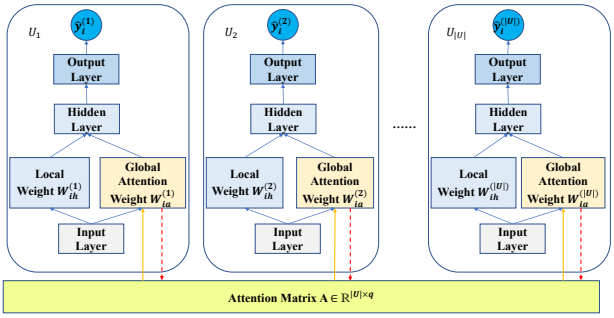

Figure 2: The architecture of BPAM model

the complex relationship of the target user and his/her neighbors.The BP neural network has shown the ability of highly non-linear mapping, and can well fit the non-linear relationship between the target users and their neighbors [Goh, 1995]. Most importantly, it is a shallow network, which can be trained efficiently and make accurate prediction without a large number of samples.

3.2 Construction of KNN Rating Matrix

The general process for BPAM is illustrated in Figure 1. After extracting the rating matrix $\mathbf{R}$ from the database, we obtain the K-Nearest Neighbors (KNN) for each user by calculating the cosine similarity between users [Sarwar *et al.*, 2001]. The KNN rating matrix for each user is formed by intercepting the rating information of the target user and his/her k nearest neighbors, where the KNN rating matrix for user u is denoted by $\mathbf{N}^{(u)} \in \mathbb{R}^{|\mathcal{V}|\times(k+1)}$. The last column in $\mathbf{N}^{(u)}$ denotes the ratings of the target user u to items, which can be used as labels for training and testing, and the first k columns are the ratings of his/her neighbors to items. The i-th row $\mathbf{N}^{(u)}_{i*} \in \mathbf{N}^{(u)}$ corresponds to item i, which is regarded as a training sample $(\mathbf{X}^{(u)}_{i*}, \mathbf{y}^{(u)}_{i*})$ of the BP neural network. $\mathbf{y}^{(u)}_{i*}$ and $\mathbf{X}^{(u)}_{i*}$ are the ratings of the target user u and his/her neighbors to item i respectively. In the local training process, the error between $\hat{\mathbf{y}}^{(u)}_{i*}$ and $\mathbf{y}^{(u)}_{i*}$ will be fed back to adjust the parameters.

3.3 Data Processing

Recommendation algorithms are known to suffer from serious sparsity problem. That is, the number of ratings per user obeys the long-tailed distribution [Jing *et al.*, 2015]. Hence most of the values in the training sample vector 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, we obtain the j-th value $x'^{(u)}_{i,j}$ of the new input vector $\mathbf{X}'^{(u)}_{i*}$ by

$$x'^{(u)}_{i,j} = \begin{cases} \text{mean}(\mathbf{N}^{(u)}_{*j}), & \text{if } x^{(u)}_{i,j} \text{ is zero ;} \\ x^{(u)}_{i,j}, & \text{otherwise} \end{cases} \quad (2)$$

where $\mathbf{N}^{(u)}_{*j}$ is the j-th column of $\mathbf{N}^{(u)}$, that is, the j-th nearest neighbor's ratings of user u. $\text{mean}(\mathbf{N}^{(u)}_{*j})$ represents the mean value of nonzero elements of $\mathbf{N}^{(u)}_{*j}$. In this way, a new KNN rating matrix $\mathbf{N}'^{(u)}$ is formed.

3.4 BP Neural Network with Attention Mechanism

By introducing the attention mechanism [Chen *et al.*, 2017], the proposed BPAM model is able to consider not only the local weight of the neighbors to the target user, but also the global attention weight of these neighbors to all their nearest target users. For instance, assume that the nearest target user set of user u is $\mathcal{T} = \{U_1, U_2, ..., U_t\}$, that is, user u is one of the k nearest neighbors of these target users. When predicting the ratings of the target user U_1, we regard the impact of user u on U_1 as the local weight, and the impact on $\mathcal{T}$ as the global weight. In this manner, we can prevent the model from falling into the local optimum. The architecture of the BPAM model is illustrated in Figure 2, where the input layer, hidden layer and output layer are the three layers in the BP neural network. $\mathbf{A} \in \mathbb{R}^{|\mathcal{U}|\times q}$ denotes the global attention weight matrix for all users, where q is the number of neurons in the hidden layer. The i-th row $\mathbf{A}_{i*} \in \mathbf{A}$ denotes the global attention weight of the user i to his/her nearest target user set. $\mathbf{W}^{(u)}_{ih} \in \mathbb{R}^{k\times q}$ and $\mathbf{W}^{(u)}_{ia} \in \mathbb{R}^{k\times q}$ denote the local weight matrix and the global weight matrix of user u's neighbors respectively. The local weight and the global weight are combined to predict the ratings of the target user.

The training set $\mathcal{D} = \{(\mathbf{X}'^{(u)}_{1*}, \mathbf{y}^{(u)}_{1*}), ..., (\mathbf{X}'^{(u)}_{m_u*}, \mathbf{y}^{(u)}_{m_u*})\}$ is composed of some rows in the new KNN rating matrix $\mathbf{N}'^{(u)}$ obtained from Eq. (2). In these rows, the target user has rated the corresponding items and m_u is the number of

the target user's ratings. The training process of BPAM is divided into two stages, forward propagation and error back propagation [Li *et al.*, 2012]. The forward propagation, i.e. the rating prediction process, can be defined as:

$$\begin{aligned}\mathbf{h}^{(u)} &= \delta((\mathbf{W}_{ih}^{(u)} + \alpha\mathbf{W}_{ia}^{(u)})^\top \mathbf{X'}_{i*}^{(u)} + \mathbf{b}_{ih}^{(u)}) \\ \hat{\mathbf{y}}_{i*}^{(u)} &= \delta({\mathbf{W}_{ho}^{(u)}}^\top \mathbf{h}^{(u)} + \mathbf{b}_{ho}^{(u)})\end{aligned} \tag{3}$$

where $\mathbf{b}_{ih}^{(u)}$ denotes the bias vector among the input layer and hidden layer, $\mathbf{W}_{ho}^{(u)}$ and $\mathbf{b}_{ho}^{(u)}$ denote the weight matrix and the bias vector among the hidden layer and output layer. α is the trade-off parameter which is used to tune the importance of the global weight. $\mathbf{W}_{ho}^{(u)}$, $\mathbf{W}_{ih}^{(u)}$, $\mathbf{b}_{ih}^{(u)}$ and $\mathbf{b}_{ho}^{(u)}$ are initialized randomly before training. But $\mathbf{W}_{ia}^{(u)}$ is formed by intercepting the global attention weights of the user u's neighbors from $\mathbf{A}$ as follows,

$$\mathbf{W}_{ia}^{(u)}[t] = \mathbf{A}[\text{neighbor}(t)] \tag{4}$$

where $\mathbf{W}_{ia}^{(u)}[t]$ denotes the t-th row of $\mathbf{W}_{ia}^{(u)}$ and neighbor(t) denotes the row-index in $\mathbf{A}$ of the t-th nearest neighbor of the target user. The objective function of our BPAM model is defined as follows:

$$E = \frac{1}{2 \times |\mathcal{U}|} \sum_{u=1}^{|\mathcal{U}|} E^{(u)} + \frac{\lambda}{2}||\mathbf{A}||^2 \tag{5}$$

where

$$E^{(u)} = \frac{1}{m_u} \sum_{i=1}^{m_u} (\mathbf{y}_{i*}^{(u)} - \hat{\mathbf{y}}_{i*}^{(u)})^2 + \lambda(||\mathbf{W}_{ho}^{(u)}||^2 + ||\mathbf{W}_{ih}^{(u)}||^2) \tag{6}$$

where the regularization terms are added to prevent overfitting. BPAM adjusts parameters in the direction of the negative gradient of the objective value based on Stochastic Gradient Descent (SGD). For training sample $(\mathbf{X'}_{i*}^{(u)}, \mathbf{y}_{i*}^{(u)})$, the error back propagation, i.e. parameter update, can be defined as:

$$\begin{cases} \triangle\mathbf{W}_{ia}^{(u)} = -\eta\alpha\mathbf{e}_i^{(u)}\mathbf{X'}_{i*}^{(u)} + \lambda||\mathbf{W}_{ia}^{(u)}|| \\ \triangle\mathbf{W}_{ho}^{(u)} = -\eta\mathbf{g}_i^{(u)}\mathbf{h}^{(u)} + \lambda||\mathbf{W}_{ho}^{(u)}|| \\ \triangle\mathbf{b}_{ho}^{(u)} = -\eta\mathbf{g}_i^{(u)} \\ \triangle\mathbf{W}_{ih}^{(u)} = -\eta\mathbf{e}_i^{(u)}\mathbf{X'}_{i*}^{(u)} + \lambda||\mathbf{W}_{ih}^{(u)}|| \\ \triangle\mathbf{b}_{ih}^{(u)} = -\eta\mathbf{e}_i^{(u)} \end{cases} \tag{7}$$

where $\eta \in (0, 1)$ is the learning rate, and

$$\begin{aligned}\mathbf{g}_i^{(u)} &= \hat{\mathbf{y}}_{i*}^{(u)}(1 - \hat{\mathbf{y}}_{i*}^{(u)})(\mathbf{y}_{i*}^{(u)} - \hat{\mathbf{y}}_{i*}^{(u)}) \\ \mathbf{e}_i^{(u)} &= \mathbf{h}^{(u)}(1 - \mathbf{h}^{(u)})\mathbf{W}_{ho}^{(u)}\mathbf{g}_i^{(u)}\end{aligned} \tag{8}$$

After training the local BP neural network of user u, the global attention weight of neighbors in $\mathbf{A}$ will be updated as follows:

$$\mathbf{A}[\text{neighbor}(t)] = \mathbf{W}_{ia}^{(u)}[t] \tag{9}$$

In summary, by introducing the attention mechanism, BPAM combines the local weight and the global weight 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. (3).

Algorithm 1 The algorithm framework of BPAM

Input: $\mathbf{R}$: rating matrix; k: number of neighbors; α: trade-off parameter.
Output: $\mathbf{W}$: BP neural network weights; $\mathbf{A}$: global attention weight; $\mathbf{b}$: bias vector.

```
Randomly initialize W, A and b
Obtain the k-nearest neighbors for each user by calculating the cosine similarity between users
repeat
    for all u ∈ U do
        Form the KNN rating matrix N^(u)
        Form the new KNN rating matrix N'^(u) via Eq. (2)
        Construct the training set D from N'^(u)
        Construct the global weight W_ia^(u) from A via Eq. (4)
        for all (X'_i*^(u), y_i*^(u)) ∈ D do
            Predict the rating ŷ_i*^(u) via Eq. (3)
            Update W_ih^(u), b_ih^(u), W_ho^(u), b_ho^(u), W_ia^(u) via Eq. (7) and Eq. (8)
        end for
        Update A according to the updated W_ia^(u) via Eq. (9)
    end for
until Eq. (5) converges
Return W, A, b
```

Datasets	#Users	#Items	#Ratings	Sparsity	Scale
ml-la	610	9724	100836	98.30%	[0.5, 5]
ml-1m	6040	3706	1000209	95.53%	[1, 5]
ml-10m	69878	10677	10000054	98.66%	[0.5, 5]
filmtrust	1508	2071	35497	98.86%	[0.5, 5]
jd-1	24983	100	1810455	27.53%	[-10, 10]
jd-2	23500	100	1708993	27.28%	[-10, 10]
jd-3	24938	100	616912	75.26%	[-10, 10]
MT	55995	32629	753073	99.96%	[1, 10]

Table 1: Statistics of the eight datasets.

4 Experiments

4.1 Experimental Setting

Dataset. The experiments are conducted on eight real-world publicly available datasets: MovieLens (ml-latest (ml-la), ml-1m, ml-10m)[1], filmtrust[2], jester (jester-data-1 (jd-1), jester-data-2 (jd-2), jester-data-3 (jd-3))[3] and MovieTweetings (MT)[4]. The statistics of these eight datasets are summarized in Table 1. Notice that in order to verify the accuracy and effectiveness of the proposed model in the case of different data sizes (measured by the number of users and items), experiments are carried out on both large-scale datasets, namely ml-10m and MT, and other small datasets, respectively. We randomly split each dataset into the training set and testing set with ratio 3:1 for each user.

[1] https://grouplens.org/datasets/movielens/
[2] https://www.librec.net/datasets.html
[3] http://eigentaste.berkeley.edu/dataset/
[4] https://github.com/sidooms/MovieTweetings

Dataset	Measure	UserCF	PMF	DeepCF	NeuMF	DMF	BPAM	Least improvement	Average improvement
ml-la	RMSE	1.6430	1.1321	0.8810	0.8570	0.5637	**0.5611**	0.46%	80.96%
	MAE	1.2720	0.9091	0.6840	0.6585	0.4330	**0.1580**	174.05%	400.84%
ml-1m	RMSE	1.5757	0.9531	0.8890	0.8730	0.7528	**0.7135**	5.51%	41.38%
	MAE	1.2026	0.8661	0.7035	0.6835	0.6005	**0.3032**	98.05%	167.56%
ml-10m	RMSE	1.6276	1.8291	NA	NA	NA	**0.6892**	136.14%	150.78%
	MAE	1.2489	1.2201	NA	NA	NA	**0.3467**	251.92%	256.07%
filmtrust	RMSE	1.2062	1.1162	0.8335	0.8055	0.5117	**0.4620**	10.76%	93.64%
	MAE	0.9217	0.8999	0.6640	0.6175	0.3858	**0.3099**	24.50%	125.16%
jd-1	RMSE	4.2712	4.8221	4.6000	4.0960	3.7932	**1.4426**	162.94%	199.21%
	MAE	3.5010	1.9612	3.7560	3.1380	3.0186	**0.8710**	125.17%	253.03%
jd-2	RMSE	4.4146	4.8846	4.4120	4.1280	3.8210	**1.4890**	156.62%	190.94%
	MAE	3.6297	1.9731	3.5360	3.1720	3.0275	**0.8785**	124.60%	249.20%
jd-3	RMSE	4.6787	5.0842	4.7840	4.4440	3.1922	**2.4234**	31.72%	83.07%
	MAE	3.8920	2.0541	3.9140	3.4940	2.4581	**1.8284**	12.34%	72.96%
MT	RMSE	5.2289	2.5681	NA	NA	NA	**2.0867**	23.07%	86.83%
	MAE	4.6718	1.2850	NA	NA	NA	**1.0368**	23.94%	187.27%

Table 2: Comparison results by six different methods in terms of RMSE and MAE. The best results are highlighted in bold. The last but one column lists the least improvements achieved by BPAM compared with the second best results. The last column lists the average improvements achieved by BPAM over the five compared methods.

Dataset	Measure	BPCF-WP	BPCF	BPAM-WP	BPAM
ml-la	RMSE	0.5742	0.5736	0.5618	**0.5611**
	MAE	0.1678	0.1672	0.1606	**0.1580**
ml-1m	RMSE	0.7462	0.7434	0.7137	**0.7135**
	MAE	0.3235	0.3233	0.3033	**0.3032**
ml-10m	RMSE	0.7201	0.7156	0.6933	**0.6892**
	MAE	0.3666	0.3635	0.3511	**0.3467**
filmtrust	RMSE	0.4792	0.4790	0.4622	**0.4620**
	MAE	0.3288	0.3287	**0.3099**	**0.3099**
jd-1	RMSE	1.6796	1.6742	1.4859	**1.4426**
	MAE	0.9310	0.9241	0.8783	**0.8710**
jd-2	RMSE	1.6712	1.6703	1.5247	**1.4890**
	MAE	0.9340	0.9309	0.8885	**0.8785**
jd-3	RMSE	2.1449	2.1436	1.8698	**1.8529**
	MAE	1.1154	1.1107	**1.0159**	1.0437
MT	RMSE	2.3833	2.2888	2.3341	**2.0867**
	MAE	1.2676	1.1924	1.2198	**1.0368**

Table 3: Analysis on the impact of attention mechanism and data processing: Comparison results of different variants.

Evaluation measures. We utilize the root-mean-square error (**RMSE**) and mean-absolute-error (**MAE**) to evaluate the performance of the predicted results. Smaller values of RMSE and MAE indicate the better performance.

4.2 Comparison Results

We compare the proposed BPAM method with the following five methods:

- **UserCF** [Herlocker *et al.*, 1999] is a typical recommendation algorithm, which predicts the ratings of the target users based on the ratings of similar users. It is usually utilized as a benchmark of recommendation systems.
- **PMF** [Mnih and Salakhutdinov, 2008] 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.
- **DeepCF** [Deng *et al.*, 2019] 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.
- **NeuMF** [He *et al.*, 2017] combines Generalized Matrix Factorization (GMF) and Multi-Layer Perceptron (MLP) to learn the user–item interaction function.
- **DMF** [Xue *et al.*, 2017] 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.

The comparison results are shown in Table 2. Memory error occurs in DNNs-based models, namely DeepCF, NeuMF and DMF, when training large-scale data due to a mass of parameters. Therefore, we record their corresponding unpredictable results as "NA" in Table 2. Since different datasets have different rating scales, as shown in the "Scale" column in Table 1, the RMSE and MAE values have exhibited some numerical differences on different datasets.

According to Table 2, we have the following key observations. Overall, the proposed BPAM model has achieved significant improvements over the second best method (which may vary from one dataset to another) on most of the datasets in terms of both RMSE and MAE. The only exception occurs on ml-la and ml-1m in terms of RMSE, where only less than 6% least improvements have been achieved. When compared with the five methods, extremely good improvements have been achieved with at least 41.38% average improvements. Another observation is that on the MovieLens dataset-

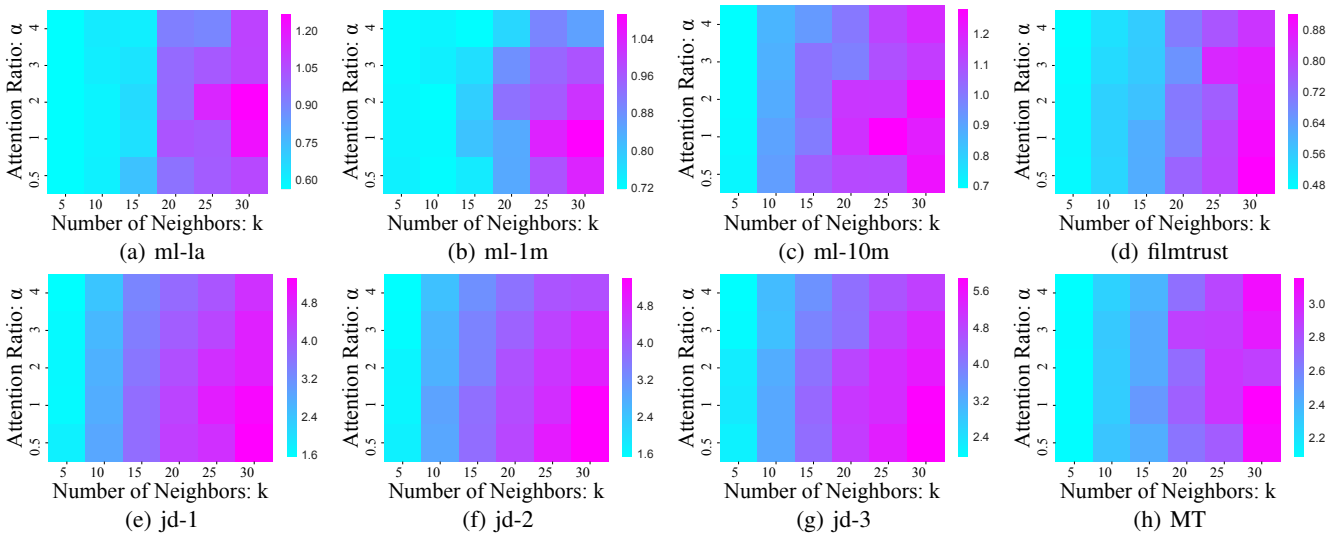

Figure 3: Parameter analysis: The RMSE values obtained by BPAM with varying attention ratio α and number of neighbors k.

s, namely ml-la, ml-1m and ml-10m, as the data size increases (from ml-la to ml-10m), the proposed BPAM model still makes significant improvements. However, the three deep neural network-based algorithms, namely DeepCF, NeuMF and DMF, fail to generate predicted ratings on ml-10m due to memory error. Similarly, on the another large-scale dataset, namely MT, the three existing DNNs-based CF algorithms also fail to generate predicted ratings. In addition, even on the small datasets, the BPAM model still outperforms the three existing deep neural network-based algorithms. This result has confirmed the effectiveness of utilizing shallow BP neural network and attention mechanism in recommender systems.

4.3 Impact of Attention Mechanism and Data Processing

In the design of BPAM, we utilize data processing to represent users' rating preferences (higher or lower) and introduce the global attention weights to prevent the model from local optimum. To validate the effectiveness of data processing and attention mechanism in our model, we compare our model with the following three variants: **BPCF-WP** (without data processing and attention mechanism), **BPCF** (without attention mechanism), **BPAM-WP** (without data processing). As shown in Table 3, we can observe that: (1) The methods with data processing perform a little better than those without data processing, especially on the MT dataset. It demonstrates the effectiveness of data processing before the input vectors are fed into the BP neural network. (2) BPAM outperforms BPCF with a large margin in all cases, which validates that our attention mechanism in BPAM can effectively capture the global attention weights about users' impacts on their nearest target user set.

4.4 Sensitivity Analysis of Hyper-parameters

In this section, we analyze the impact of the two hyper-parameters: k and α by using the heat map. According to the results in Figure 3, the proposed model generates the best performance with $k = 5$ on most of the datasets except ml-1m. On ml-1m, the best performance is achieved with $k = 10$. Smaller number of neighbors usually performs better since it leads to fewer parameters and a relatively small number of samples are sufficient for training. Additionally, we can find that the optimal attention ratio α is around 2 to 4. And compared with $\alpha = 0.5$, the values of RMSE and MAE decrease significantly, which indicates the significance of the global attention weight in BPAM.

5 Conclusion

In this paper, we propose a novel recommendation algorithm based on BP neural network with Attention Mechanism (BPAM). In the proposed algorithm, the BP neural network is utilized to learn the complex relationship between the target users and their neighbors. Compared with DNNs, shallow BP neural network can not only reduce the computational and storage costs, but also prevent the model from over-fitting 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. Extensive experiments on eight benchmark datasets demonstrate that our proposed model distinctly outperforms state-of-the-art methods.

Acknowledgements

This work was supported by NSFC (61876193), Guangdong Natural Science Funds for Distinguished Young Scholar (2016A030306014), Tip-top Scientific and Technical Innovative Youth Talents of Guangdong special support program (2016TQ03X542), and Key Areas Research and Development Program of Guangdong (2018B010109007).

References

[Bell and Koren, 2007] Robert M. Bell and Yehuda Koren. Scalable collaborative filtering with jointly derived neigh-

borhood interpolation weights. In *ICDM*, pages 43–52, 2007.

[Chen *et al.*, 2017] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua. Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention. In *SIGIR*, pages 335–344, 2017.

[Cheng *et al.*, 2018] Zhiyong Cheng, Ying Ding, Xiangnan He, Lei Zhu, Xuemeng Song, and Mohan S Kankanhalli. A3NCF: An adaptive aspect attention model for rating prediction. In *IJCAI*, pages 3748–3754, 2018.

[Deng *et al.*, 2019] Zhi-Hong Deng, Ling Huang, Chang-Dong Wang, Jian-Huang Lai, and Philip S.Yu. DeepCF: A unified framework of representation learning and matching function learning in recommender system. In *AAAI*, 2019.

[Goh, 1995] Anthony TC Goh. Back-propagation neural networks for modeling complex systems. *Artificial Intelligence in Engineering*, 9(3):143–151, 1995.

[Gong and Zhang, 2016] Yuyun Gong and Qi Zhang. Hashtag recommendation using attention-based convolutional neural network. In *IJCAI*, pages 2782–2788, 2016.

[He *et al.*, 2017] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *WWW*, pages 173–182, 2017.

[He *et al.*, 2018] Xiangnan He, Xiaoyu Du, Xiang Wang, Feng Tian, Jinhui Tang, and Tat-Seng Chua. Outer product-based neural collaborative filtering. In *IJCAI*, pages 2227–2233, 2018.

[Herlocker *et al.*, 1999] Jonathan L. Herlocker, Joseph A. Konstan, Al Borchers, and John Riedl. An algorithmic framework for performing collaborative filtering. SIGIR, pages 230–237, 1999.

[Hu *et al.*, 2016] Liang Hu, Longbing Cao, Jian Cao, Zhiping Gu, Guandong Xu, and Dingyu Yang. Learning informative priors from heterogeneous domains to improve recommendation in cold-start user domains. *ACM Trans. Inf. Syst.*, pages 13:1–13:37, 2016.

[Hu *et al.*, 2019] Qi-Ying Hu, Ling Huang, Chang-Dong Wang, and Hong-Yang Chao. Item orientated recommendation by multi-view intact space learning with overlapping. *Knowledge-Based Systems*, pages 358 – 370, 2019.

[Huang *et al.*, 2019] Ling Huang, Chang-Dong Wang, Hong-Yang Chao, Jian-Huang Lai, and Philip S. Yu. A score prediction approach for optional course recommendation via cross-user-domain collaborative filtering. *IEEE ACCESS*, 7:19550–19563, 2019.

[Jia *et al.*, 2010] Rongfei Jia, Maozhong Jin, and Chao Liu. Using temporal information to improve predictive accuracy of collaborative filtering algorithms. In *2010 12th International Asia-Pacific Web Conference*, pages 301–306, 2010.

[Jing *et al.*, 2015] Liping Jing, Peng Wang, and Liu Yang. Sparse probabilistic matrix factorization by laplace distribution for collaborative filtering. In *IJCAI*, pages 1771–1777, 2015.

[Li *et al.*, 2012] Jing Li, Ji-hang Cheng, Jing-yuan Shi, and Fei Huang. Brief introduction of back propagation (BP) neural network algorithm and its improvement. In *Advances in Computer Science and Information Engineering*, pages 553–558. Springer, 2012.

[Li *et al.*, 2017] Piji Li, Zihao Wang, Zhaochun Ren, Lidong Bing, and Wai Lam. Neural rating regression with abstractive tips generation for recommendation. In *SIGIR*, pages 345–354, 2017.

[Mnih and Salakhutdinov, 2008] Andriy Mnih and Ruslan R Salakhutdinov. Probabilistic matrix factorization. In *NIPS*, pages 1257–1264, 2008.

[Patra *et al.*, 2015] Bidyut Kr. Patra, Raimo Launonen, Ville Ollikainen, and Sukumar Nandi. A new similarity measure using Bhattacharyya coefficient for collaborative filtering in sparse data. *Knowledge-Based Systems*, pages 163–177, 2015.

[Sarwar *et al.*, 2001] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *WWW*, pages 285–295, 2001.

[van den Oord *et al.*, 2013] Aaron van den Oord, Sander Dieleman, and Benjamin Schrauwen. Deep content-based music recommendation. In *NIPS*, pages 2643–2651. 2013.

[Wang *et al.*, 2006] Jun Wang, Arjen P. de Vries, and Marcel J. T. Reinders. Unifying user-based and item-based collaborative filtering approaches by similarity fusion. In *SIGIR*, pages 501–508, 2006.

[Wang *et al.*, 2015] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. Collaborative deep learning for recommender systems. In *KDD*, pages 1235–1244, 2015.

[Wang *et al.*, 2016] Shaonan Wang, Jiajun Zhang, and Chengqing Zong. Learning sentence representation with guidance of human attention. *arXiv preprint arXiv:1609.09189*, 2016.

[Wang *et al.*, 2018a] Chang-Dong Wang, Zhi-Hong Deng, Jian-Huang Lai, and Philip S. Yu. Serendipitous recommendation in E-commerce using innovator-based collaborative filtering. *IEEE Transactions on Cybernetics*, pages 1–15, 2018.

[Wang *et al.*, 2018b] Shoujin Wang, Liang Hu, Longbing Cao, Xiaoshui Huang, Defu Lian, and Wei Liu. Attention-based transactional context embedding for next-item recommendation. In *AAAI*, 2018.

[Xue *et al.*, 2017] Hong-Jian Xue, Xinyu Dai, Jianbing Zhang, Shujian Huang, and Jiajun Chen. Deep matrix factorization models for recommender systems. In *IJCAI*, pages 3203–3209, 2017.

[Zhu *et al.*, 2017] Junxing Zhu, Jiawei Zhang, Lifang He, Quanyuan Wu, Bin Zhou, Chenwei Zhang, and Philip S. Yu. Broad learning based multi-source collaborative recommendation. In *CIKM*, pages 1409–1418, 2017.