IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received December 4, 2018, accepted December 21, 2018, date of publication March 5, 2019, date of current version April 2, 2019.

Digital Object Identifier 10.1109/ACCESS.2018.2890553

A BP Neural Network Recommendation Algorithm Based on Cloud Model

HONG TANG[1,2], MAN LEI[1,2], QIN GONG[1], AND JICHAO WANG[2]
[1]School of Communication and Information Engineering, Chongqing University of Posts and Telecommunications, Chongqing 400065, China
[2]Chongqing Key Lab of Mobile Communications Technology, Chongqing University of Posts and Telecommunications, Chongqing 400065, China

Corresponding author: Man Lei (leim12@163.com)

This work was supported by the Changjiang Scholars and Innovative Research Team in University of China under Grant IRT_16R72.

ABSTRACT Rating prediction is one of the key studies in the recommendation system. The traditional rating prediction algorithms only utilize user's rating data to predict unknown ratings. In fact, however, we need to deal with the randomness and fuzziness of user ratings. At the same time, the sparsity of the data limits the performance of these algorithms. Therefore, a backpropagation neural network recommendation algorithm based on cloud model is proposed. First, the algorithm uses cloud model qualitative and quantitative transformation method to deal with the user ratings, which generates multiple cloud prediction values for users, and these values constitute the cloud layer. Then, the cloud layer joins the neural network, which can improve the accuracy of rating prediction. The missing value of rating matrix is filled in and recommendation list for the target user is generated. Finally, we perform experiments on the real-world data set, finding out that the proposed method achieves better results compared with the traditional recommendation methods in term of recall, precision, and F1.

INDEX TERMS Recommendation system, cloud model, neural network, rating prediction.

I. INTRODUCTION

In recent years, with the rapid development of technologies such as the Internet of Things, cloud computing and social networks, it is difficult for users to effectively mine information in massive data. The emergence of recommendation system can efficiently alleviate the problem of information overload [1]–[3].

The most common solution of recommendation system is the collaborative filtering (CF) recommendation method based on rating prediction [4], [5]. However, the randomness and fuzziness of user ratings will produce deviation to the prediction results to some extent. The randomness of the user ratings is mainly reflected in user's rating preference, which is regarded as constant during a period of time. The user is used to rating high, but sometimes the lower ratings appears. The fuzziness of user ratings is reflected in most of the social platforms based on ratings. Users can only use 1-5 ratings to represent their preference degree, which include very dissatisfied, dissatisfied, average, satisfied, and very satisfied. Discrete expressions are more ambiguous, and no continuous expression is more accurate. Finally, the sparsity of data makes the system unable to efficiently analyze user behavior and accurately predict user ratings, so as it is difficult to further improve the quality of the recommendation results [6], [7].

In this paper, we propose a backpropagation neural network recommendation algorithm based on cloud model (CM-BPNN) to overcome the problems.

The contributions of this paper are summarized as follows.

1) We utilize the method of qualitative and quantitative conversion of cloud model to predict different cloud ratings for unrated items. The method deals with the randomness and fuzziness of user ratings from the perspective of the user and item.
2) A backpropagation neural network recommendation algorithm based on cloud model is designed. According to the characteristics of the nonlinear structure and fault tolerance of multi-layer networks, the cloud layer composed of multi-dimensional rating cloud model is added to the neural network to improve the rating prediction accuracy and then fill the missing value of rating matrix.
3) Compared with the traditional recommendation algorithm for solving data sparsity problem, our proposed algorithm makes full use of user rating data and improves recommendation quality.

The remainder of this paper is organized as follows. In the section II, we describe the definition of relevant cloud model and propose the design of cloud model score transformation algorithm. In section III, we propose a BP neural network score prediction model based on cloud model and algorithm designing. In section V, we analyze the experimental results, and section VI concludes our work.

II. RELATED WORK

In the recommendation system, the commonly algorithm is to use the ''user-item'' historical ratings as a data source to predict the item's rating by analyzing the user behavior model. Collaborative filtering (CF) is a typical and the most popular information filtering technology in recommender system [8]–[10]. Collaborative filtering has two classical models: nearest neighbor model and latent factor model. The nearest neighbor model includes user-based collaborative filtering (UBCF) and item-based collaborative filtering (IBCF), which the first problem to be solved is the processing of user historical rating data. However, the cloud model is an important research method for user behavior analysis.

Cloud model is a qualitative and quantitative transformation model proposed by academician Li [11] on the basis of probability theory and fuzzy theory. In the uncertain artificial intelligence, the cloud model can achieve the uncertain transformation between qualitative concepts and quantitative values [12], [13]. Cloud model is widely used in data mining, decision analysis, image processing, artificial intelligence and other fields. Now cloud model is widely used for experimental exploration, which is a hot topic for scholars. In 2007, aiming at the deficiencies of traditional similarity measure to strictly match the object attributes. According to the comprehensive analysis of traditional methods, Zhang *et al.* [14] introduced the cloud model into the idea of CF algorithm and proposed a collaborative filtering algorithm to compare user similarity at the knowledge level. Finally, experiments showed that this algorithm could obtain a smaller MAE value. In 2008, Zhang and Liu [15] proposed an improved traditional IBCF recommendation algorithm based on cloud model. This algorithm first uses the back-cloud algorithm to calculate the cloud feature vectors of each item, and then uses the cloud similarity calculation method to calculate the item similarity and find the item neighbor. By using the weighted average strategy for neighbors of the item to predict unrated items, the top N similar items with the highest rating are recommended for target users. In 2009, Wu and Zheng [16] used cloud model to obtain user profiles and then calculated item or user similarity by cloud model similarity method to achieve recommendation. Sa [17] proposed an improved CF algorithm, which is a multi-index evaluation algorithm based on cloud model clustering. Based on the multi-index weight of the item, this algorithm realizes the soft clustering of the item to reduce the impact of data sparsity. Li *et al.* [18] proposed a similarity calculation method of two normal cloud models, which overcomes the problem of high time complexity of random selection of cloud drop similarity calculation and improves the efficiency of the recommendation algorithm. Sui and Qiao [19] constituted an emotional tendency vector through a cloud model, and the recommendation relies on the similarity calculation between the nearest neighbor cloud model vectors. Yang *et al.* [20] designed the normal cloud model-based algorithm for multi-attribute trusted cloud service selection strategy. According to the basis of the central limit theorem, the distribution of the user experience data is an approximate normal distribution, so the normal cloud model is used to describe the user experience data.

Aiming at the shortcomings of traditional similarity measurement methods, cloud-based model is a key hub for linking qualitative knowledge and quantitative knowledge. The cloud model is successfully applied to personalized recommendation, and a cloud model similarity calculation method is proposed. Then, on this basis, some researchers have proposed an item rating prediction algorithm based on cloud model, which uses similar items to fill in the user's scores on unrated items. To make up for data sparsity and improve the quality of the recommendation system. Although this method overcomes the drawbacks of strictly matching object attributes in the traditional similarity measurement method, the researchers fail to consider the randomness and fuzziness of user ratings to reduce the recommendation accuracy. Moreover, these methods simply consider the ratings without considering the time, and the time impart factor has an influence on the timeliness of the user rating data.

Finally, the sparsity of data is still the main limitation and weakness of the collaborative filtering recommendation method [21], [22]. The development of machine learning techniques such as neural networks provides a new research method for recommendation algorithms. For example, Zhang *et al.* [23] used feature representation methods based on quadratic polynomial regression models, and then used these potential features as input data for deep neural network models for rating prediction. Chen [24] used BP neural network to fill the blank rating, then used collaborative filtering to form the nearest neighborhood, and finally generated the recommendation collection. Neural networks can be used to improve the performance of recommendation algorithms, and the sparsity of data is a key issue for recommendation systems. Therefore, choosing an efficient and compatible recommendation algorithm to overcome this problem is the focus of research.

III. PROPOSED METHOD

A. RELATED DEFINITION OF CLOUD MODEL

Cloud model, which mainly reflects the randomness and fuzziness of most events in the real world, has been widely used in artificial intelligence of uncertainty [25]. The related definitions of cloud model are given below.

Definition 1: Cloud and cloud drops. Suppose D is a quantitative domain comprising accurate values, and B is the corresponding qualitative concept on D. If the quantitative value $x(x\in D)$ is a random realization of the qualitative concept B, and the certainty degree of quantitative value x belong to qualitative concept B is $\mu(\mu\in[0,1])$, it is a random number with stable tendency. The distribution of x on domain D is called cloud model, referred to as cloud. x is called a cloud drop.

The cloud is formed by a number of randomly generated cloud drops. The overall shape of the cloud is represented by three digital features: Expectation Ex, Entropy En and Hyper entropy He, which reflects the quantitative features in the qualitative concept.The overall characteristics of cloud is expressed by three digital features $C(Ex,En,He)$. Ex can be expressed as the center of gravity position of all cloud drops in the domain, which best reflects the coordinates of the qualitative concept in the number domain. En represents the measurable granularity of qualitative concept. In the domain space, it can be accepted by qualitative concept, that is, fuzzy degree. He is the dispersion degree of En, which reflects the cohesion of each precise value belong to the certainty degree of the linguistic value, namely the thickness of cloud drops in the domain. In the cloud model, the expectation curve of the qualitative concept cloud is approximately subject to normal distribution.

Definition 2: Forward cloud transformation is a mapping from the overall feature of the qualitative concept to the quantitative value, which can transform the qualitative cloud feature value (Ex,En,He) into the quantitative values. Its generation is as follows:

$$En' = NORM(En, He^2) \tag{1}$$

$$x = NORM(Ex, En'^2) \tag{2}$$

$$\mu = exp^{\frac{-(x-Ex)^2}{2(En')^2}} \tag{3}$$

where $NORM(x,\delta^2)$ is a normal gaussian random variable with expectation of x and variance of δ.

Definition 3: Reverse cloud transformation is to realize the transformation from quantitative values to qualitative concepts, and convert a set of accurate values into qualitative concepts represented by numerical features (Ex,En,He). Its generation is as follows:

$$Ex = \frac{1}{N}\sum_{j=1}^{N} x_j \tag{4}$$

$$En = \sqrt{\frac{\pi}{2}} \times \frac{1}{N}\sum_{j=1}^{N} |x_j - Ex| \tag{5}$$

$$S^2 = \frac{1}{N-1}\sum_{j=1}^{N}(x_j - Ex)^2 \tag{6}$$

$$He = \sqrt{S^2 - En^2} \tag{7}$$

B. RATING HANDING

On the e-commerce website, users are generally satisfied with items through ratings, and the rating range is 1-5. The higher the user rating, the higher the user satisfaction. Our work is to establish a model that can accurately predict user's ratings on the test set.

Considering the timeliness of user rating data, inspired by Chen *et al.* [26], this paper selects the exponential function of the controllable variable as the historical rating time weight. Compared with all the information in the time domain, the recent data more accurately reflects the current user rating preference, and the rating data with close time interval has a higher contribution. And to reduce the time complexity, the user is given a time interval to set a threshold. When the rating time interval is greater than or equal to the threshold L, the rating data is processed. The user set is defined as $U = \{u_1,u_2,\ldots,u_n\}$, the item set is defined as $I = \{i_1,i_2,\ldots,i_m\}$, where n and m are the numbers of users and items in the recommender system. The user-item history rating matrix is $R = [r_{u,i}]^{n\times m}$, where $r_{u,i}$ indicates the history rating of the user u for item i. The updated rating is as shown in (8).

$$rt_{u,i} = \begin{cases} e^{-\lambda|t_{now}-t_{u,i}|} \times r_{u,i} & |t_{now} - t_{u,i}| \geq L \\ r_{u,i} & |t_{now} - t_{u,i}| < L \end{cases} \tag{8}$$

$|t_{now} - t_{u,i}|$ indicates the interval value between the current time t_{now} and the rating time $t_{u,i}$ of user u for item i. Then the user-item rating matrix is updated as follows:

$$RT = \begin{array}{c c} & \begin{array}{ccccc} I_{T,1} & . & . & . & I_{T,m} \end{array} \\ \begin{array}{c} U_{T,1} \\ . \\ . \\ . \\ U_{T,n} \end{array} & \begin{bmatrix} rt_{1,1} & . & . & . & rt_{1,m} \\ . & . & & & . \\ . & & . & & . \\ . & & & . & . \\ rt_{n,1} & . & . & . & rt_{n,m} \end{bmatrix} \end{array} \tag{9}$$

where $U_{T,u} = \{rt_{u,1},rt_{u,2},\ldots,rt_{u,i}\}(u{=}1,2,\ldots,n)$ is the score vector of user u, $I_{T,i} = \{rt_{1,i},rt_{2,i},\ldots,rt_{u,i}\}(i{=}1,2,\ldots,m)$ is the score vector of item i. If user u does not rate item i, then $rt_{u,i} = 0$.

C. CLOUD MODEL RATING CONVERSION ALGORITHM

Aiming at the complexity of user's subjective behavior, this paper proposes a qualitative and quantitative rating transformation method based on cloud model. This section mainly analyzes the behavior of user ratings, including the processing of randomness and fuzziness of user rating data. For the rating randomness, the cloud model can ignore the influence of some random values to some extent. For the rating fuzziness, the cloud model predicts the user rating as a continuous value, and more accurately describe the user's rating preference.

According to the reverse cloud transformation method, each user's rating vector $U_{T,u} = \{rt_{u,1},rt_{u,2},\ldots,rt_{u,i}\}$ and each item's rating vector $I_{T,i} = \{rt_{1,i},rt_{2,i},\ldots,rt_{u,i}\}$ were transformed into qualitative concepts represented by numerical features $C_u(Ex_u,En_u,He_u)$ and $C_i(Ex_i,En_i,He_i)$ respectively. In the process of transformation, which will ignore the effect of partial random values. If u tends to have a high score and sometimes a few low scores. In the calculation process, the low score has a low contribution to the overall user rating preference, which will not affect the overall trend of the user preference expectation curve represented by digital features $C_u(Ex_u,En_u,He_u)$.

In the practical application, the certainty degree $\mu_{u,i}$ of quantitative value $rt_{u,i}$ belong to qualitative concept is difficult to obtain, and the user-based and item-based certainty degrees of the rated item should be calculated according to the forward cloud transformation method, then the calculation equation is as follows:

$$\mu^{\tau}_{u,i} = exp^{\frac{-(x-Ex_{\tau})^2}{2(En'_{\tau})^2}} (\tau = u, i) \tag{10}$$

Ex_u represents the score expectation of a single user u, $En'_u = NORM(En_u, He_u^2)$ represents a gaussian random implementation of the expectation and hyper-entropy of a single user, where $rt_{u,i}$ represents updated rating of the user u rated item i. Similarly, Ex_i represents the score expectation of single item i, and $En'_i = NORM(En_i, He_i^2)$ represents a gaussian random implementation of expectation and hyper-entropy of single item.

In the recommendation system, due to the rapid increase in the size of the data, the number of users and items increase sharply, resulting in user rating data having different degrees of sparsity. Data information of unrated items can not be obtained, resulting in the inability to directly calculate the corresponding rating certainty degree. Therefore, the user-based and item-based certainty degrees of unrated items are calculated as follows:

$$p\mu^{u}_{u,i} = \frac{\sum \mu^{u}_{u,i} \times Sim_{mk}}{\sum Sim_{mk}} \tag{11}$$

$$p\mu^{i}_{u,i} = \frac{\sum \mu^{i}_{u,i} \times Sim_{nK}}{\sum Sim_{nK}} \tag{12}$$

We utilize Jaccard similarity method to calculate the similarity between users, as well as items. Sim_{mk} indicates the similarity between the item i_m and the item i_k. At the same time, Sim_{nK} indicates the similarity between the user u_n and the user u_K, Sim_{mk} and Sim_{nK} are calculated as follows:

$$Sim_{mk} = \frac{i_m \cap i_k}{i_m \cup i_k} \quad (k = 1, 2, ...m) \tag{13}$$

$$Sim_{nK} = \frac{u_n \cap u_K}{u_n \cup u_K} \quad (K = 1, 2, ..., n) \tag{14}$$

$i_m \cap i_k$ indicates the number of users who have a rating behavior for both item i_m and item i_k, and $i_m \cup i_k$ indicates the total number of users who have a rating behavior for item i_m or item i_k. $u_n \cap u_K$ indicates the number of items that user u_n and user u_K have rated, and $u_n \cup u_K$ is the total number of items that user u_n or user u_K has rated. It is assumed that the user's rating prediction corresponding to $p\mu^{\tau}_{u,i}$ is $pr^{\tau}_{u,i}$, $pr^{\tau}_{u,i}$ can be used as the rating prediction value of the single inference prediction model, and the user-based rating prediction $pr^{u}_{u,i}$ and the item-based rating prediction $pr^{i}_{u,i}$ are calculated according to the certainty degree of the unrated item. as follows:

$$pr^{\tau}_{u,i} = Ex_{\tau} \pm \sqrt{-2lnp\mu^{\tau}_{u,i}} \times En'_{\tau} (\tau = u, i) \tag{15}$$

The cloud model score transformation method is used to generate multiple score prediction values for the user to the item, which are user-based score prediction and item-based score prediction. In addition, we address the fuzziness of user rating and predict continuous rating value, which more accurately describes the user rating preference. The algorithm for cloud model rating conversion algorithm is shown in Algorithm 1.

Algorithm 1 Cloud Model Rating Conversion Algorithm

Input: Each user's score vector $U_{T,1}$, ..., $U_{T,u}$
Each item's score vector $I_{T,1}$, ..., $I_{T,i}$
Output: $pr^{u}_{u,i} = Ex_u \pm \sqrt{-2lnp\mu^{u}_{u,i}} \times En'_u$
$pr^{i}_{u,i} = Ex_i \pm \sqrt{-2lnp\mu^{i}_{u,i}} \times En'_i$

1: **for** $u = 1$ to n **do**
2: Construct u's cloud model $C_u(Ex_u, En_u, He_u)$ according to $U_{T,u} = \{rt_{u,1}, rt_{u,2}, \ldots, rt_{u,i}\}$ by (4-7)
3: Calculate $\mu^{u}_{u,i} = exp^{\frac{-(x-Ex_u)^2}{2(En'_u)^2}}$ according to (10)
4: Calculate the similarity between users
5: Calculate $p\mu^{u}_{u,i}$ according to (11)
6: Predict $pr^{u}_{u,i}$ according to (15)
7: **end for**
8: **for** $i = 1$ **to** m **do**
9: Construct i's cloud model $C_i(Ex_i, En_i, He_i)$ according to $I_{T,i} = \{rt_{1,i}, rt_{2,i}, \ldots, rt_{u,i}\}$ by (4-7)
10: Calculate $\mu^{i}_{u,i} = exp^{\frac{-(x-Ex_i)^2}{2(En'_i)^2}}$ according to (10)
11: Calculate the similarity between items
12: Calculate $p\mu^{i}_{u,i}$ according to (12)
13: Predict $pr^{i}_{u,i}$ according to (15)
14: **end for**

D. CLOUD MODEL-BP NEURAL NETWORK RATING PREDICTION ALGORITHM

BP neural network is a typical multi-layer feedforward neural network and adopts a backpropagation training network [27]. Since the network structure of multiple layers has better representation ability for data, in the process of fitting historical data, the model will express data more abstractly through layer-by-layer feature extraction, thereby improving prediction accuracy. Based on the above proposed cloud model conversion rating method, this paper aims to improve rating prediction accuracy and alleviate the recommendation efficiency problem caused by the sparsity of user rating data, and forms a cloud layer by constructing multiple cloud prediction values. Then this paper constructs a CM-BPNN rating prediction model to obtain a reasonable prediction value and realize the missing value filling of rating matrix. The overall structure of our model is shown in Fig.1:

In the system framework of Fig.1, BP network is a 4-layer network. The main input parameters include the initial weight vector for W_{ic}, W_{ch}, W_{ho}, the threshold for α_h and β_o, learning rate η(0-1) and error control parameter ε. In our model, the weight matric between the input layer and the cloud layer W_{ic} sets $\{1,1\}$.

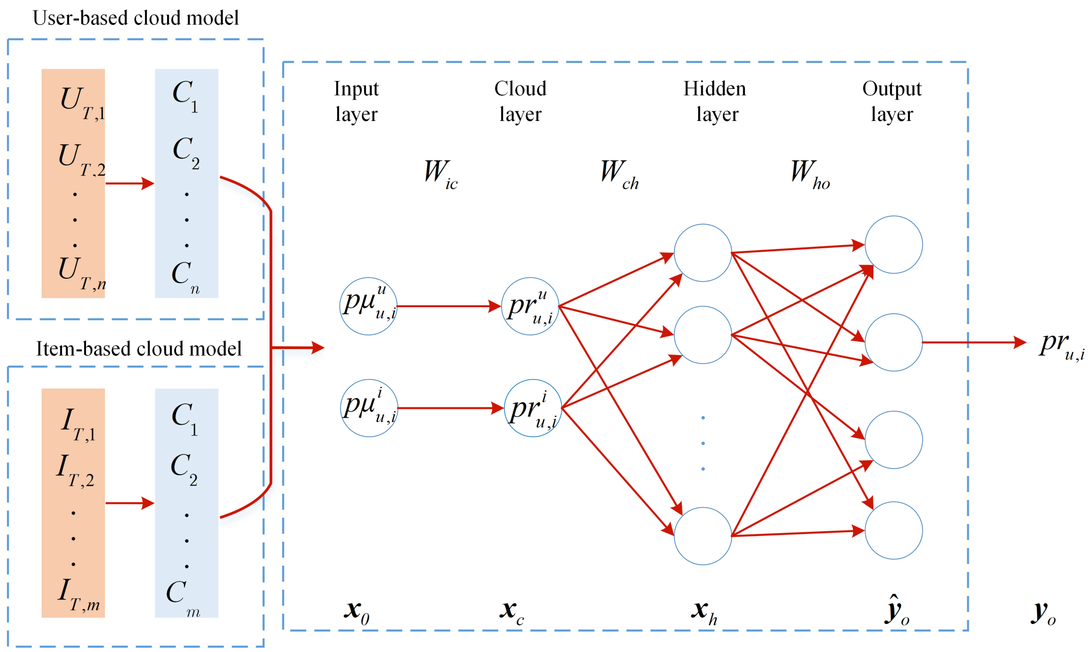


FIGURE 1. BP neural network structure based on cloud model.

According to our model, in the input layer, the input vector x_o is user-based and item-based the certainty degrees of unrated items.

$$x_o = (p\mu^u_{u,i}, p\mu^i_{u,i}) \tag{16}$$

When x_o passes through the input layer, the output of the cloud layer is obtained by the following equation:

$$x_c = cmrp(p\mu^c_{u,i}) \tag{17}$$

where $c = 1$ is the user-based rating prediction, and $c = 2$ is the item-based rating prediction, and the $cmrp()$ indicates the activation function. In this paper, we can obtain the activation function of the cloud neurons.

$$cmrp(z) = Ex_c \pm \sqrt{-2lnz} \times En'_c \tag{18}$$

where z is the weighted input of the input neuron.

The output of the h-th neuron in the hidden layer is expressed by (19):

$$x_h = activation(\alpha_h + \sum_{c=1}^{2} w_{ch}x_c) \tag{19}$$

where w_{ch} is the weight matric between the c-th neuron of the cloud layer and the h-th neuron of the hidden layer. α_h is the threshold of the h-th neuron in the hidden layer. In this paper, we choose sigmoid as the activation function for the hidden layer and the output layer.

In the output layer, our training goal is to predict the rating $pr_{u,i}$ of the comprehensive score for user u to item i. We can obtain the output of the o-th neuron in the output layer $\hat{y}_o$:

$$\hat{y}_o = activation(\beta_o + \sum_{h} w_{ho}x_h) \tag{20}$$

where w_{ho} represents the connection weight between the h-th neuron of the hidden layer and the o-th neuron of the output layer. β_o is the threshold of the o-th neuron in the output layer. Finally, we use a mean square error (MSE) method to evaluate the difference between the prediction result $\hat{y}_o$ and the supervised value y_o.

$$E = \frac{1}{2}\sum_{l=1}^{M}(\hat{y}_o - y_o)^2 \tag{21}$$

where M is the number of neurons in the output layer. Next, we use the gradient descent method to update w_{ho},

$$w_{ho} = w_{ho} - \eta\frac{\partial E}{\partial w_{ho}} \tag{22}$$

We use inp to represent the weighted input of the output layer:

$$inp = \sum_{h} w_{ho}x_h \tag{23}$$

Then:

$$\frac{\partial E}{\partial w_{ho}} = \frac{\partial E}{\partial \hat{y}_o}\frac{\partial \hat{y}_o}{\partial inp}\frac{\partial inp}{\partial w_{ho}} = \frac{\partial E}{\partial \hat{y}_o}\frac{\partial \hat{y}_o}{\partial inp}x_h \tag{24}$$

According to (21).

$$\frac{\partial E}{\partial \hat{y}_o} = (\hat{y}_o - y_o) \tag{25}$$

$$\begin{aligned}\frac{\partial \hat{y}_o}{\partial inp} &= \frac{\partial activation(inp + \beta_o)}{\partial inp} \\ &= activation'(inp + \beta_o) \\ &= \hat{y}_o(1 - \hat{y}_o)\end{aligned} \tag{26}$$

where we use the nature of the sigmoid function $activation'() = activation()(1 - activation())$.

We analyze the update rules for w_{ch} in the same way:

$$w_{ch} = w_{ch} - \eta \frac{\partial E}{\partial w_{ch}} \tag{27}$$

We use $hinp$ to represent the weighted input of the hidden layer:

$$hinp = \sum_{c} w_{ch} x_c \tag{28}$$

Then:

$$\frac{\partial E}{\partial w_{ch}} = \frac{\partial E}{\partial x_h} \frac{\partial x_h}{\partial hinp} \frac{\partial hinp}{\partial w_{ch}} = \frac{\partial E}{\partial x_h} \frac{\partial x_h}{\partial hinp} x_c \tag{29}$$

According to (25-26).

$$\begin{aligned} \frac{\partial E}{\partial x_h} &= \sum_{o} \frac{\partial E}{\partial inp} \frac{\partial inp}{\partial x_h} \\ &= \sum_{o} \frac{\partial E}{\partial \hat{y}_o} \frac{\partial \hat{y}_o}{\partial inp} \frac{\partial inp}{\partial x_h} \\ &= \sum_{o} (\hat{y}_o - y_o)\hat{y}_o(1 - \hat{y}_o) w_{ho} \end{aligned} \tag{30}$$

We record $(\hat{y}_o - y_o)\hat{y}_o(1 - \hat{y}_o)$ as g_o.

$$\begin{aligned} \frac{\partial x_h}{\partial hinp} &= \frac{\partial activation(hinp + \alpha_h)}{\partial hinp} \\ &= activation'(hinp + \alpha_h) \\ &= x_h(1 - x_h) \end{aligned} \tag{31}$$

Next, we analyze the update rules for β_o and α_h in the same way, based on the above discussion, we obtain the following rules for updating the parameters in the CM-BPNN model:

$$\beta_o = \beta_o - \eta \frac{\partial E}{\beta_o} = \beta_o - \eta g_o \tag{32}$$

$$\alpha_h = \alpha_h - \eta \frac{\partial E}{\alpha_h} = \alpha_h - \eta x_h(1 - x_h) \sum_{o} g_o w_{ho} \tag{33}$$

$$w_{ho} = w_{ho} - \eta \frac{\partial E}{\partial w_{ho}} = w_{ho} - \eta g_o x_h \tag{34}$$

$$w_{ch} = w_{ch} - \eta \frac{\partial E}{\partial w_{ch}} = w_{ch} - \eta x_c x_h(1 - x_h) \sum_{o} g_o w_{ho} \tag{35}$$

During the iterative process, the error E will stop when the error is reduced to the set value or the number of fits is greater than the preset maximum number of times, otherwise the next round of fitting is performed. In the process of learning, each set of samples is fully utilized, and a reasonable predictive value can be obtained by calculating the data layer by layer. The rating prediction is filled with a ''user-item'' rating matrix. A Top-N recommendation list is generated for the user according to the rating prediction. The CM-BPNN rating prediction algorithm is shown in Algorithm 2.

Algorithm 2 CM-BPNN Rating Prediction Algorithm

```
Input: x_o = (pμ^u_{u,i}, pμ^i_{u,i})
Output: pr_{u,i}(u = 1, 2, ..., n.i = 1, 2, ..., m)
Random initialize W_ch, W_ho, α_h, β_O, η(0 − 1), ε
repeat
for all (pμ^u_{u,i}, pμ^i_{u,i}) ∈ x_o do
    Predict pr^u_{u,i}, pr^i_{u,i} according to (17)
    Calculate each neuron output of the hidden layer
    according to (19)
    Calculate each neuron output of the output layer
    according to (20)
    Update α_h and β_o according to (32-33)
    Update w_ch and w_ho according to (34-35)
If E ≫ ε
break
end for
```

IV. EXPERIMENTAL COMPARISON AND ANALYSIS OF RESULTS

In this section, we will compare our algorithm with some mainstream algorithms on the Douban data set. Then, we perform a detailed analysis of our experimental results from three aspects.

A. DATASET PREPARING

To illustrate the performance of the proposed algorithm, we use real network data to carry out experiments. we chose a real-world dataset to perform the experimental research. The experimental data is collected from Douban site (http://www.Douban.com), which provides the evaluation on movie, books and music. On the Douban site, users can rate to movies, books and music, and Douban also provides users with social services. The initial data set contains 339 ratings for 2096 movies. Among them, each user has at least 20 ratings, each movie is evaluated by at least 20 users, the rating ranges from 1 to 5. The experimental data set contains a time range of 2007-2018.

Table 1 shows the statistics of the experimental data. In order to verify the accuracy of the prediction algorithm, the data set is divided into training set and test set. The training set is used to study or train the parameters in the prediction model, and the test set is used to evaluate the performance of the model. To control the sparsity of data, we introduce variables that denotes the percentage of the training set to the entire data set.

B. EVALUATION METRICS

To estimate the quality of each algorithm and compare these algorithms, we adopt three evaluation metrics in our experiments: recall, precision, and $F1$. The expressions are as follows:

$$recall = \frac{\sum_u |R(u) \cap T(u)|}{\sum_u T(u)} \tag{36}$$

where $R(u)$ represents a collection of recommended items for the user u, $T(u)$ represents a collection of items that the user

TABLE 1. Statistics of dataset.

Statistic	Quantity
Number of users	399
Number of items	2096
Number of ratings	356613
Range of ratings	1 to 5
Minimum number of ratings peer user	20
Minimum number of ratings peer item	20
Average number of ratings by users	241.25
Average number of ratings for items	51.37

u actually participated in on the test set. This metric reflects how much of the user-item score record is included in the final recommendation list. The definition of precision is as follows:

$$precision = \frac{\sum_u |R(u) \cap T(u)|}{\sum_u R(u)} \tag{37}$$

$R(u)$ and $T(u)$ are consistent with the recall metric description, which reflects how much of the final recommendation list is an actual user-item score record. The definition of $F1$ is as follows:

$$F1 = \frac{precision \times recall \times 2}{precision + recall} \tag{38}$$

$F1$ is a comprehensive metric reflecting the precision and recall. In the recommendation system, there may be inconsistent precision and recall. Generally, the higher the precision and recall, the better recommendation performance, but sometimes there will be high precision and low recall; Or low precision and high recall. And $F1$ is a common metric that considers this problem comprehensively.

C. COMPARED METHODS

This section introduces the baselines that will be used for comparing with our method. we intend to compare the performance of our proposed method with some classic baselines to verify the superiority.

The wide application of collaborative filtering recommendation algorithm is due to accurate rating prediction, mainly including neighbor-based collaborative filtering and model-based collaborative filtering. Neighbor-based collaborative filtering is divided into user-based uollaborative filtering (UBCF) and item-based collaborative filtering (IBCF).

UBCF (User-based Collaborative Filtering) uses items that have been rated by similar users to predict the unknown score of the target user. At present, all the successful collaborative filtering algorithms are improved on the basis of UBCF algorithm. UBCF algorithm is selected as the basic algorithm.

Slope One algorithm is an Item-based collaborative filtering recommendation algorithm proposed by professor Daniel Lemire in 2005. Compared with other similar algorithms, Slope One algorithm is simple, easy to implement, high efficiency of execution, and relatively high accuracy of recommendation. Slope One is used to calculate the score of unrated items in the way of weighted average according to the ratings of related users on the same item.

In this paper, UBCF and Slope One in collaborative filtering algorithm are selected as the most basic comparison algorithm to verify the performance of the recommended system.

Model-based collaborative filtering includes latent factor model, matrix decomposition model, potential factor model, regression model, etc. The user behavior data set is divided into training set and test set, and the training set is to use machine learning method to fit model parameters, and the test set is to measure the accuracy of the model. In the training process, the user interest model is built for the target user by iterative method, and then the recommended item data is input into the trained model to screen out the top N items that are of the most interest to the target user.

Latent factor model (LFM) is one of the hottest research topics in the recommendation system in recent years, it was first appeared in the field of text mining, is used to find the content of the implied meaning in the text. LFM is an improvement of SVD decomposition algorithm, which uses semantic analysis technology to mine potential topics or categories among users and uses iterative optimization for calculation. LFM has the best recommendation effect in the current recommendation algorithm. Therefore, choosing LFM as the baseline algorithm for comparison is sufficient to prove our proposed algorithm.

D. THE IMPACT OF PARAMETER L

Our method will be compared with other rating prediction methods which include UBCF, Slope One and Latent Factor Model LFM. And our proposed method is CM-BPNN(Tw). Where Tw (Time-weight) means based on the time weight. CM-BPNN(Tw) represents the CM-BPNN algorithm based on the time weight. The paper weights the user's rating information. In order to better reflect how the time factor affects the recommendation result, this part of the experiment compares the impact on the recommendation result under different time interval threshold settings. The size of the time interval determines the contribution of the user's rating data, that is, the weight value of the data information, which in turn affects the recommendation result of the recommendation algorithm. The range of the time interval is $0 \leq L \leq 11$ in this experiment, and the unit is year. The experiment used 80% of the data as a training set and 20% of the data as a test set for comparison. The experimental results are shown in Fig.2:

As we can see from this figure, CM-BPNN(Tw) method achieves much larger values of recall, precision, and $F1$ than Baseline. It shows that as the value of L is increasing, the number of user ratings that need to be calculated is decreasing, and the recall, precision, and $F1$ of the algorithm show a steady downward trend. Comprehensive analysis of the reasons for the above results. When $L = 0$, indicating that the user rating data is the rated in the current year. the rating contribution is 100%, the time weight is 1, and the remaining years data is not processed by time weight, the results of its three recommended metrics are the best. When the user's rating time interval is large, for instance $L = 11$, it means that

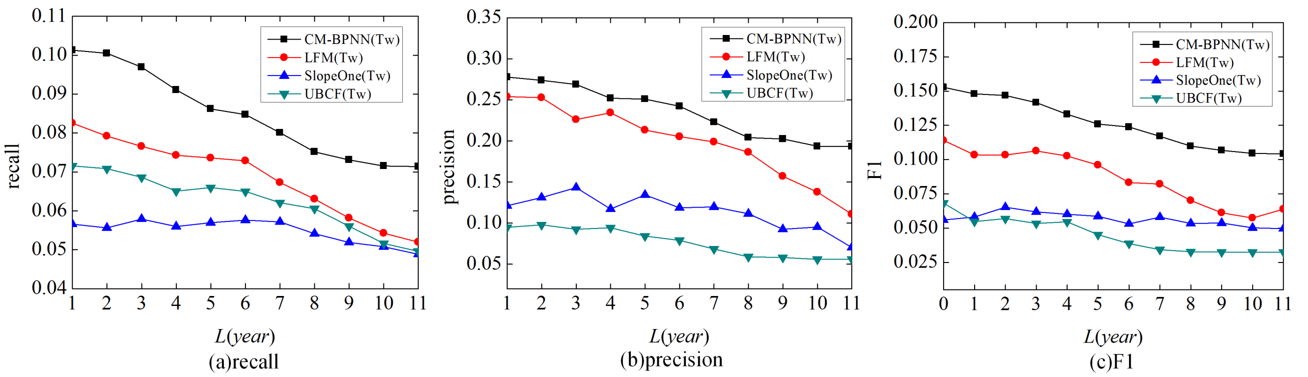


FIGURE 2. Different time interval values *L* comparison.

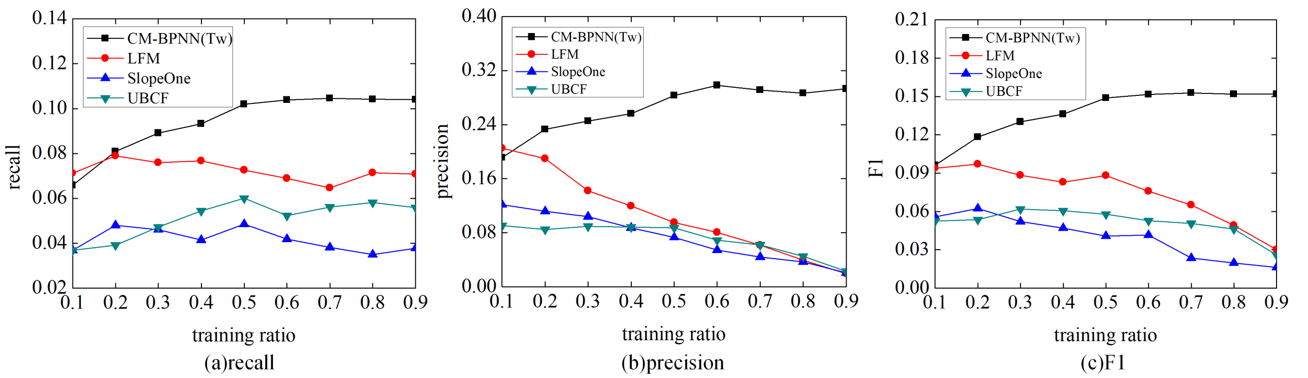


FIGURE 3. Comparison of recommended results for each algorithm under different sparsity.

user ratings are time weighted when the user's rating time is only greater than 11 years. The experimental data period is 2007-2018, $L = 11$ is equivalent to the user's rating data and is not time-weighted. The results of the three recommended metrics are the lowest. This paper also compares LFM, Slope One, and UBCF based on time weight experiments, and improved recommendation results. The above experiments prove that our method effectively proves the influence of time factor on the recommendation result and improves the problem of insufficient recommendation performance formed by the timeliness of data.

E. EXPERIMENT ABOUT DATA SPARSITY

The training set is used to study or train the parameters in the prediction model, and the test set is used to evaluate the performance of the model. To control the sparsity of data, we introduce variables that denotes the percentage of the training set to the entire data set. In this experiment, the sparsity is represented by different proportions of the training set. The lower the proportion of the training set, the more sparsity of data, and vice versa. The experimental results are as follows:

Fig.3 shows the effect of recommendation results at different sparsity levels. As can be seen from the figure, when the training ratio is equal to 0.1, LFM method achieves large values of recall, precision, and $F1$ than our proposed algorithm. When the training ratio is greater than 0.1, we observe that our method gives better results with a considerable superiority of three measures. Therefore, in general, the CM-BPNN(Tw) method is superior to Baseline. In the comparison between the precision and the $F1$, the effects of the Baseline method decrease with the decrease of the sparsity, and CM-BPNN(Tw) has been steadily increasing. The experiment proves that the proposed method can show better recommendation results at lower sparsity and can effectively improve the recommendation quality in data sparsity environment.

The above experiments compare the proposed method with Baseline method in terms of the recommendation performance. This paper also compares the recommended effects of each method based on time weight. The experimental results are shown in the following Fig.4:

Fig.4 the experimental results show that under the different levels of sparsity, each method based on time weight have a good performance, and the proposed approach is still better than the Baseline based on time weight.

Table 2 shows that under different sparsity levels, the recommended effects of each method based on time weight are improved to some extent, and our method is still better than the Baseline method based on time weight. It can be seen from the table that this method has better recommendation effects on recall, precision, and $F1$.

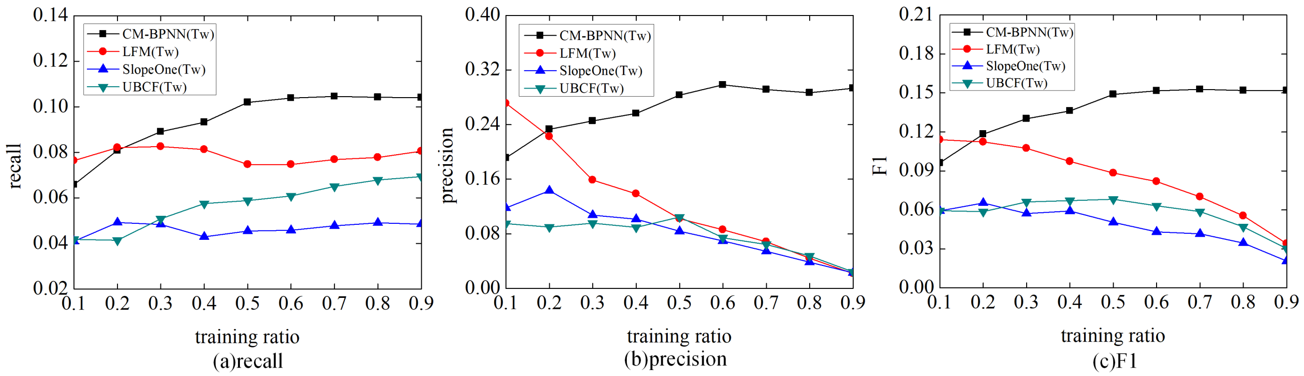


FIGURE 4. Comparison of recommended results for each method (based on time weight) under different sparsity.

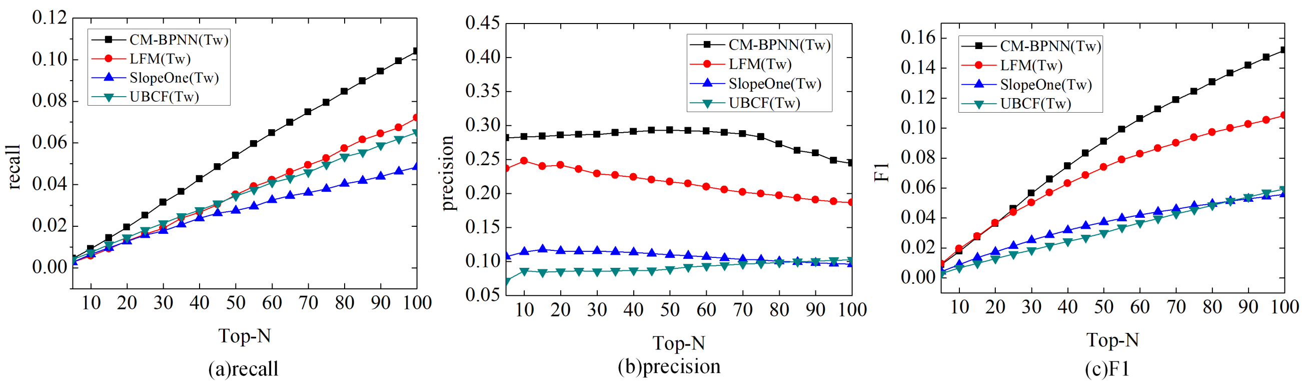


FIGURE 5. Comparison of Top-N recommendation results of each method based on time weight.

TABLE 2. Comparison of recommended performance of different methods.

Algorithm name	Recall	Precision	F1
CMBPNN(Tw)	0.10459	0.29823	0.15276
LFM	0.07875	0.20534	0.09712
Slope one	0.04925	0.12156	0.06235
UBCF	0.06590	0.09068	0.06466
LFM (Tw)	0.08253	0.27122	0.11402
Slope One(Tw)	0.04907	0.14328	0.06528
UBCF(Tw)	0.06938	0.10452	0.06824

F. EXPERIMENT ABOUT TOP-N

In the recommendation system, generating a personalized Top-N recommendation list for the user based on the rating prediction, indicating that the first N items of the recommendation list are recommended to the target user. The experiment controls the number of N, and the N value is regularly increased. Comparing recommendation results in different N value and analyze the difference between algorithms. In the above experiments, the processing of the time weight of the historical rating has been confirmed, which can improve the recommendation performance. Therefore, the following figure shows a comparison of Top-N recommendations for our method and other algorithms based on time weight:

Fig.5 shows that as the value of N is increasing, CM-BPNN achieves larger values of recall, precision, and $F1$. On the $F1$ metric, when N is less than 30, LFM is close to the method of this paper; while N is larger than 30, the effect is lower than we proposed method, Slope One and UBCF can not achieve the effect of our method. In the comparison of recall and precision, the proposed method has better recommendation effect than other algorithms based on time weight. Therefore, in general, the proposed algorithm is superior to other algorithms based on time weight in Top-N recommendation comparison, and has better recommendation effect.

V. CONCLUSIONS

In this paper, we have proposed a BP neural network based on cloud model recommendation algorithm. The recommendation research is therefore conducted based on rating prediction in social networks. In order to improve the accuracy of prediction caused by the randomness and fuzziness of user ratings, and to improve the recommendation quality caused by data sparsity. We first consider the timeliness of user rating information, weakening the historical rating time weight. Then, we use the cloud model to address the randomness and fuzziness of the user ratings, and predict multiple cloud rating prediction for user, and then combines the BP neural network to obtain the user's comprehensive rating prediction. The experimental results show that the proposed method achieves good performance, prove that each method based on time weight is successful attempt. In different sparsity environments, the experimental results show that the proposed method is very effective in data sparsity environment. And the experiment is also compared under different Top-N recommendations, with a result showing that the proposed method improves the recommendation performance compared with the baseline method to some extent.

In this paper, we solve the problem of inaccurate recommendation caused by data sparsity. However, the limitation of the work is that we do not take into account user social relationships. In the recommendation system, the user interest model can be established through the user's friend relationship, and the integration of social relationship can improve the recommendation quality.

For future work, we plan to increase the scalability and portability of the proposed method in this paper. For instance, social relationship between users could be considered in our proposed method.

REFERENCES

[1] B. Sun and L. Dong, "Dynamic model adaptive to user interest drift based on cluster and nearest neighbors," *IEEE Access*, vol. 5, pp. 1682–1691, 2017.

[2] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez, "Recommender systems survey," *Knowl.-Based Syst.*, vol. 46, pp. 109–132, Jul. 2013.

[3] M. D. Ekstrand, J. T. Riedl, and J. A. Konstan, "Collaborative filtering recommender systems," *Found. Trends Hum.-Comput. Interact.*, vol. 4, no. 2, pp. 81–173, Feb. 2010.

[4] N. Polatidis and C. K. Georgiadis, "A multi-level collaborative filtering method that improves recommendations," *Expert Syst. Appl.*, vol. 48, pp. 100–110, Apr. 2016.

[5] X. Yang, Y. Guo, Y. Liu, and H. Steck, "A survey of collaborative filtering based social recommender systems," *Comput. Commun.*, vol. 41, pp. 1–10, Mar. 2014.

[6] G. Guo, H. Qiu, Z. Tan, Y. Liu, J. Ma, and X. Wang, "Resolving data sparsity by multi-type auxiliary implicit feedback for recommender systems," *Knowl.-Based Syst.*, vol. 138, pp. 202–207, Dec. 2017.

[7] L. Wang, Z. Yu, B. Guo, T. Ku, and F. Yi, "Moving destination prediction using sparse dataset: A mobility gradient descent approach," *ACM Trans. Knowl. Discovery Data*, vol. 11, no. 3, p. 37, Apr. 2017.

[8] W. Lu and F. L. Chung, "Computational creativity based video recommendation," in *Proc. 39th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2016, pp. 793–796.

[9] M.-S. Shang, Z.-K. Zhang, T. Zhou, and Y.-C. Zhang, "Collaborative filtering with diffusion-based similarity on tripartite graphs," *Phys. A, Stat. Mech. Appl.*, vol. 389, no. 6, pp. 1259–1264, Mar. 2010.

[10] X. Wang, Y. Liu, G. Zhang, Y. Zhang, H. Chen, and J. Lu, "Mixed similarity diffusion for recommendation on bipartite networks," *IEEE Access*, vol. 5, pp. 21029–21038, 2017.

[11] D. Li, C. Liu, and W. Gan, "A new cognitive model: Cloud model," *Int. J. Intell. Syst.*, vol. 24, no. 3, pp. 357–375, Mar. 2009.

[12] G. Wang, C. Xu, and D. Li, "Generic normal cloud model," *Inf. Sci.*, vol. 280, pp. 1–15, Oct. 2014.

[13] W. S. Li, Y. Wang, J. Du, and J. Lai, "Synergistic integration of graph-cut and cloud model strategies for image segmentation," *Neurocomputing*,Vol. 257, pp. 37–46, Sep. 2017.

[14] W. G. Zhang, D. Y. Li, P. Li, and J.-C. Kang, "A collaborative filtering recommendation algorithm based on cloud model," *J. Softw.*, vol. 18, no. 10, pp. 2403–2411, Oct. 2007.

[15] X. X. Zhang and T. H. Liu, "Applying cloud model to improve Item-based collaborative filtering recommendation algorithm," *Library Intell.*, vol. 53, no. 1, pp. 117–120, 2009.

[16] Y. Wu and J. Zheng, "A research of recommendation algorithm based on cloud model," in *Proc. IEEE Int. Conf. Intell. Comput. Intell. Syst.*, Oct. 2010, pp. 469–473.

[17] L. Sa, "Collaborative filtering recommendation algorithm based on cloud model clustering of multi-indicators item evaluation," in *Proc. Int. Conf. Bus. Comput. Global Informatization*, Jul. 2011, pp. 645–648.

[18] H. L. Li, C. H. Guo, and W. R. Qiu, "Similarity measurement between normal cloud models," *Acta Electron. Sinica*, vol. 39, no. 11, pp. 2561–2567, Nov. 2011.

[19] G. Sui and H. Qiao, "Emotional tendency contrast recommendation algorithm based on cloud model," *J. Netw.*, vol. 9, no. 2, pp. 437–442, Feb. 2014.

[20] Y. Yang, R. Liu, Y. Chen, T. Li, and Y. Tang, "Normal cloud model-based algorithm for multi-attribute trusted cloud service selection," *IEEE Access*, vol. 6, pp. 37644–37652, 2018.

[21] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proc. 26th Int. Conf. World Wide Web.*, Apr. 2017, pp. 173–182.

[22] Y. Koren, "Factor in the neighbors: Scalable and accurate collaborative filtering," *ACM Trans. Knowl. Discovery Data*, vol. 4, no. 1, Jan. 2010, Art. no. 1.

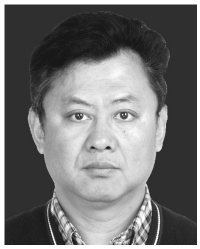
[23] L. Zhang, T. Luo, F. Zhang, and Y. Wu, "A recommendation model based on deep neural network," *IEEE Access*, vol. 6, pp. 9454–9463, 2018.

[24] D. E. Chen, "The collaborative filtering recommendation algorithm based on BP neural networks," in *Proc. Int. Symp. Intell. Ubiquitous Comput. Educ.* Chengdu, China, May 2009, pp. 234–236.

[25] C. Xu, G. Wang, and Q. Zhang, "A new multi-step backward cloud transformation algorithm based on normal cloud model," *Fundam. Inf.*, vol. 133, no. 1, pp. 55–85, 2014.

[26] J. Chen, X. J. Liu, B. Li, and W. Zhang, "Personalized microblogging recommendation based on dynamic interests and social networking of users," *Acta Electron. Sinica*, vol. 45, no. 4, pp. 898–905, 2017.

[27] J. Wang, Y. Wen, Y. Gou, Z. Ye, and H. Chan, "Fractional-order gradient descent learning of BP neural networks with Caputo derivative," *Neural Netw.*, vol. 89, pp. 19–30, May 2017.



HONG TANG received the master's degree in photoelectric signal process from Sichuan University and the Ph.D. degree in computer software and theory from the University of Chongqing, in 2003. He is currently a Professor with the Chongqing University of Posts and Telecommunications, China. His current research interests include social networks, computer networks, and telecommunication technologies.

MAN LEI received the B.S. degree in communication and information engineering from the Chongqing University of Posts and Telecommunications, in 2016, where she is currently pursuing the M.S. degree. Since 2017, she has been involved in data mining and machine learning. Her research interests include recommender systems, social networks, and neutral networks.

QIN GONG received the B.S. degree from Chengdu University, in 2016. She is currently pursuing the master's degree in information and communication engineering with the Chongqing University of Posts and Telecommunications. Her research interests include recommender systems, natural language processing, and deep learning techniques.

JICHAO WANG received the B.S. degree from the College of Mobile Telecommunications, Chongqing University of Posts and Telecommunications, in 2016, where he is currently pursuing the master's degree in electronics and communication engineering. His research interests include machine learning and data mining.