A Group Recommendation Approach Based on Neural Network Collaborative Filtering

Jia Du
School of Computer Science and Technology
Wuhan University of Technology
Wuhan, China
jiayehello@gmail.com

Lin Li
School of Computer Science and Technology
Wuhan University of Technology
Wuhan, China
cathylilin@whut.edu.cn

Peng Gu
School of Computer Science and Technology
Wuhan University of Technology
Wuhan, China
380059082@qq.com

Qing Xie
School of Computer Science and Technology
Wuhan University of Technology
Wuhan, China
felixxq@whut.edu.cn

***Abstract*—At present, the most popular recommendation algorithms belong to the class of latent factor models(LFM). Compared with traditional user-based and item-based collaborative filtering methods, the latent factor model effectively improves recommendation accuracy. In recent years, deep neural networks have succeeded in many research fields, such as computer vision, speech recognition, and natural language processing. However, there are few studies combining recommendation systems and deep neural networks, especially for group recommendation. Some academic studies have adopted deep learning methods, but they mainly use it to process auxiliary information, such as acoustic features of sounds, and semantic analysis of texts, the inner product is still used to deal with latent features of users and items. In this paper, we first obtain the nonlinear interaction of latent feature vectors between users and projects through multi-layer perceptron(MLP), and use the combination of LFM and MLP to achieve collaborative filtering recommendation between users and items. Secondly, based on the individual's recommendation score, a fusion strategy based on Nash equilibrium is designed to ensure the average satisfaction of the group users. Our experiments are conducted on the Track 1 of KDD CUP 2012 public data set, taking the square root mean square error(RMSE) as the evaluation index. The experiment compares the traditional LFM optimization model, the MLP model and the LFM-MLP hybrid model in individual recommendation, and compares the strategy proposed in this paper with the traditional three single group strategies, the most pleasure, the average strategy and the least misery. The experimental results show that the proposed method can effectively improve the accuracy of group recommendation.**

***Keywords*—collaborative filtering, latent factor model, MLP, Group Recommendation, Nash equilibrium**

I. Introduction

The recommendation system can effectively help users filter irrelevant information, find information of interest to them, and improve the utilization of information by analyzing and mining relevant information about users and commodities [1], [2]. The recommended collaborative filtering recommendation (CF) [3] by analyzing existing user behaviors and information is widely used in the industry. Among them, the neighborhood-based recommendation method [4] and the matrix decomposition-based recommendation method (MF) [5] are the most widely used, and the MF is better able to deal with data sparsity than the former, so the industry mostly uses matrix-based The recommended method of decomposition is recommended.

At present, the main matrix decomposition method has singular value decomposition SVD [6] . In order to reduce the dimension, the original SVD decomposes the original matrix into the product of three matrices, which solves the problem of data sparsity to some extent, but it needs to be sparse. The matrix is densified, which increases the amount of calculation and storage. The LFM (Latent Factor Model) model directly uses the original matrix to be decomposed into two feature matrices, and has achieved good results in Netflix movie recommendations. Based on the traditional SVD decomposition model, we have built a matrix decomposition model (SVD++) [7] that incorporates user SNS relationships and social behavior parameters.

Deep learning is re-emerging, and the field of deep learning in the recommendation system is booming. Deep neural networks (DNNs) have been found to be effective in many areas, from computer vision, speech recognition to text processing [8]–[11]. Previous studies have applied neural networks to recommendations [12]–[14], but using DNN to model auxiliary information, such as the textual description of the project, the audio characteristics of the music, and the visual content of the image, combined with the use of MF, Use inner product to combine the potential features of users and projects. There is also a general framework that combines the deep neural network recommendation algorithm [15] with the abstracted matrix decomposition and the multi-layer perceptron MLP, which can capture the linear relationship between users and items, and also acquire their nonlinear relationship. Making more accurate scoring predictions [16] but their research focused primarily on implicit feedback and did not take advantage of explicit feedback from users (ie, ratings and reviews). User satisfaction has not been observed, and there is naturally a lack of negative feedback.

The group recommendation field has also achieved certain research results in recent years, and most of them use a neighborhood-based collaborative filtering algorithm. There are studies [17] using neighbor-based collaborative filtering recommendation for each group member recommendation, predicting group members to view the project's preferences, using the least misery strategy to fuse the recommendation results of each group member to the group's recommendation results, but no Compare with other integration strategies. There is also a research [18] on Spark proposed a matrix decomposition algorithm based on distributed stochastic gradient descent, which is better than ALS in MLlib, using the least misery strategy, which still belongs to a single fusion strategy.

The group-oriented collaborative filtering recommendation studied in this paper is mainly divided into two aspects. The first is to study the current mainstream collaborative filtering-based recommendation method, and analyze the mainstream singular value decomposition model and the related optimization algorithm hidden factor model LFM. Combining the neural network model MLP to achieve collaborative filtering of neural networks; the second research group fusion strategy, and using the Nash equilibrium method as a group fusion strategy to fuse the user prediction scores within the group, so as to achieve the appropriate group fusion strategy Recommend items for Weibo users.

II. Collaborative filtering recommendation

A. LFM-based recommendation algorithm

The large-scale users' rating matrix is usually sparse, and the MF algorithm can effectively improve the shortcomings of the traditional collaborative filtering recommendation algorithm on large-scale sparse data sets. The SVD algorithm [6] is a commonly used method in matrix factorization algorithms, but needs to be divided into three matrices, and the calculation amount is too large. LFM does not need to fill the original sparse matrix, and the original matrix is decomposed into the product of two low-rank feature matrices of **P** and **Q**, which reduces the amount of calculation and achieves good precision in explicit feedback data (score data). The matrix **P** is a User-Feature matrix and **Q** is an Item-Feature matrix. Reconstruct the predictive scoring matrix:

$$\mathbf{R} = \mathbf{P}^T\mathbf{Q} \tag{1}$$

MF associates each user and item with a real-valued vector of latent features. The number of latent factors is F, the subscripts i and u are the number of the item and the user respectively, and all the individual vector representations in the matrix are as shown in formula :

$$\hat{\mathbf{r}}_{u,i} = \sum_{f=1}^{F} \mathbf{p}_{u,f}\mathbf{q}_{i,f} \tag{2}$$

In actual application scene, the difference in user scoring habits and the differences between the items make the ratings difference in the data set. For example, some users are accustomed to playing high scores, and the user's ratings deviates from the ratings of most users. Differences in quality, appearance, style, etc. between items can also result in score bias. So add a deviation in LFM to eliminate the difference in ratings:

$$\hat{\mathbf{r}}_{u,i} = \sum_{f=1}^{F} \mathbf{p}_{u,f}\mathbf{q}_{i,f} + \mathbf{b}_u + \mathbf{b}_i + \mu \tag{3}$$

Where μ represents the average global score, $\mathbf{b}_u$ and $\mathbf{b}_i$ represent the user's deviation and item's deviation. In addition to the user's explicit rating of the item, the social behavior relationship is added, and the user in the data set is a microblog user, and the user usually has comments, forwarding, and likes. That is, the collections $\mathbf{S(u)}$ and $\mathbf{A(u)}$ are added, where $\mathbf{S(u)}$ is a collection of users of interest to the user, and $\mathbf{A(u)}$ is the user behavior collection:

$$\mathbf{p}_u \leftarrow \mathbf{p}_u + |\mathbf{S(u)}|^{\alpha 1} \sum_{k\in\mathbf{S(u)}} y_k + |\mathbf{A(u)}|^{\alpha 2} \sum_{k\in\mathbf{A(u)}} y_k \tag{4}$$

The parameters $\alpha 1$ and $\alpha 2$ both take the value of -0.5, and u is added to $\mathbf{S(u)}$, that is, each user pays attention to himself by default, and the final rating prediction formula:

$$\begin{aligned}\hat{\mathbf{r}}_{u,i} = \mathbf{b}_u + \mathbf{b}_i + \mu + \mathbf{q}_{i,f} \sum_{f=1}^{F}(\mathbf{p}_u + |\mathbf{S(u)}|^{\alpha 1} \sum_{k\in\mathbf{S(u)}} y_k \\ + |\mathbf{A(u)}|^{\alpha 2} \sum_{k\in\mathbf{A(u)}} y_k)\end{aligned} \tag{5}$$

The matrices **P** and **Q** are continually optimized by the global objective function. After multiple iterations, the local optimal solution is obtained, and the prediction rating matrix is approximated to the real rating matrix. The global objective function is:

$$C(\mathbf{p},\mathbf{q}) = \sum_{(u,i)\in Train} (\mathbf{r}_{u,i} - \hat{\mathbf{r}}_{u,i})^2 \tag{6}$$

To avoid over-fitting, add the regularization term constraint parameter λ to the above formula, and update to get the following formula:

$$C(\mathbf{p},\mathbf{q}) = \sum_{(u,i)\in Train} (\mathbf{r}_{u,i} - \hat{\mathbf{r}}_{u,i})^2 + \lambda(|\mathbf{p}_u|^2 + |\mathbf{q}_i|^2) \tag{7}$$

In order to make the final predicted rating matrix infinitely approximate to the real rating matrix, the objective function is minimized and the objective function is solved by the stochastic gradient descent algorithm (SGD).

B. the MLP model based on deep neural network

The neural network has been shown to approximate any continuous function [19], then we can use the neural network to simulate the calculation formula (5) at the latent factors pace, that is, the interaction function of a single user–item is f(ui). Drawing on the idea of matrix factorization and latent factor models, the user-item matrix is decomposed into two latent feature matrices. That is the input layer, which is located at the bottom of Fig. 1. The two matrices User and Item describe the feature representation of the user u and the item i in the latent factor space.

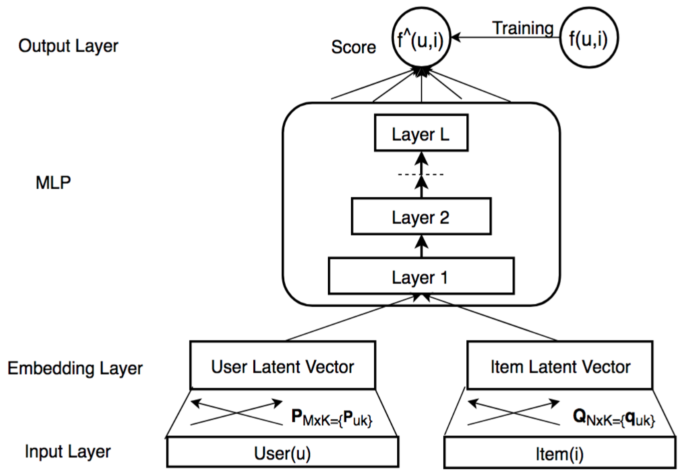

Fig. 1. MLP collaborative filtering framework

Above the input layer is the embedding layer, which is a fully connected layer that projects a sparse latent factor space matrix into multiple vectors. The obtained $\mathbf{P}$ and $\mathbf{Q}$ can be regarded as the latent feature vectors corresponding to the user u and the item i on the latent factor space. Drawing on multi-mode deep learning [20], this paper uses two paths to model users and items, combining the latent feature vectors of users and items.

The shallow neural network cannot resolve the interaction and association between the user and the item's late features, because simply a vector concatenationt network is not sufficient to model the collaborative filtering algorithm implementation.So adding a latent layer improves the ability of the model to model complex nonlinear relationships, and the use of multi-layer perceptron neural network MLP improves the level of abstraction. Then, the feature vectors of the user and item in the embedding layer are sent to the MLP network structure, which can be regarded as a neural network collaborative filtering layer. And the feature vectors $\mathbf{p}$ and $\mathbf{q}$ are mapped to the prediction score. Properly setting each layer of the CF layer, you can find some latent structures of user-item interactions. The dimensions of the last latent layer determine the capabilities of the model.

The network structure of the latent layer is in accordance with the common MLP structure, like a tower structure. The bottom layer is the widest, and its size is equal to the number of latent factors, which is the number of latent feature vectors in the embedding layer, and then the number of neurons in each successive layer is gradually reduced. It is necessary to obtain more abstract data relationships and features through a small number of latent units. In the experiment, we will contiguous layers in the latent layer, halved from bottom to top, so the number of latent factor vectors in the embedding layer is a multiple of 8.

The final is the output layer, which outputs the predicted score $\hat{r}_{u,i}$. The process of training the network can be seen as minimizing the pointwise loss function between $\hat{r}_{u,i}$ and its target value f(ui).This model has greater flexibility and non-linearity to learn the interaction between $\mathbf{p_u}$ and $\mathbf{q_i}$, rather than the linear LFM method, which uses only a fixed element-based model. To be precise, the MLP model used in this paper is defined as

$$\begin{aligned}
\mathbf{z}_1 = a_1(\mathbf{p}_u, \mathbf{q}_i) = \begin{bmatrix}\mathbf{p}_u \\ \mathbf{q}_i\end{bmatrix} \quad ,\\
\phi_2(\mathbf{z}_1) = a_2(\mathbf{W}_2^T\mathbf{z}_1 + \mathbf{b}_2),\\
\cdots\\
\phi_L(\mathbf{z}_{L-1}) = a_L(\mathbf{W}_L^T\mathbf{z}_{L-1} + \mathbf{b}_L),\\
\hat{r}_{u,i} = \sigma(\mathbf{h}^T\phi_L(\mathbf{z}_{L-1}))
\end{aligned} \tag{8}$$

where $\mathbf{W}_l$, a_l, and $\mathbf{b}_l$ represent the weight matrix, activation function, and bias vector for the l-th layer's perceptron, separately.

The activation function of the MLP layer can be selected from sigmoid, hyperbolic tangent (tanh), and Rectifier (ReLU), Exponential Linear Units (ELU), and so on. Among them, the Sigmoid function limits each neuron to (0,1); and the study confirms that it will saturate. When the output of the factor approaches 0 or 1, the neuron stops learning. The dataset used in this article has a score of 1, 0, and -1. Similarly, although ReLU proves to be unsaturated and avoids overfitting, ELU is further optimized on ReLU, the tanh function is more suitable for the microblog dataset used in this article. It has been confirmed by experiments that the performance of tanh is slightly better than that of ReLu and ELU, while the result of sigmoid is the worst. The individual parameters of the model can be calculated using standard back-propagation algorithms. In order to make the final predicted scoring matrix infinitely approximate to the real rating matrix, the objective function is minimized and the objective function is solved by Adam. Adaptive Moment Estimation (Adam) [21] calculates first-order and second-order momentum based on historical gradients, thereby adjusting the learning rate of different parameters, reducing the pain of adjusting learning speed, and thus the convergence speed is faster than SGD.

C. fusion of LFM and MLP(LFM-MLP)

So far, we have proposed two cases of LFM and MLP. Through the above analysis, the input of both models is to decompose the user-item rating matrix into two latent factor matrices on the latent factor space, and then input the latent feature vector ($\mathbf{p_u}$ and $\mathbf{q_i}$). In the middle, the parameters of the internal model are continuously optimized and optimized through machine learning, and the corresponding score value f(u, i) is output. The LFM captures the linear relationship of the latent feature interaction, while the neural network MLP uses a non-linear kernel to simulate the interactive data of the latent feature. So the key to the question is: How do we integrate the two models so that they can reinforce each other and get closer to the true rate set of the user on the item?

Firstly, there is a simple way to resemble a neural tensor network (NTN), where LFM and MLP share the same input layer and embedding layer, and then combine the ratings. But this will limit the performance of the fusion model. For example, when LFM and MLP are optimal solutions in different sizes of embedding layers, that is, in the dimension of the latent factor space, the optimal solution cannot be obtained. Especially when the two sizes differ greatly, the desired effect cannot be obtained.

Secondly, in order to make both models achieve the best results, and the model is more flexible. Allows both models to process the input matrix data independently, but combines the two models when the final output is obtained. That is, on the original output layer of the two models, a layer of fully connected layers is added. The model is named LFM-MLP, expressed as follows:

$$\begin{aligned} \phi^{LFM} &= \mathbf{p}_u^L \bullet \mathbf{q}_i^L, \\ \phi^{MLP} &= a_L(\mathbf{W}_L^T(\cdots a_2(\mathbf{W}_2^T \begin{bmatrix} \mathbf{p}_u^M \\ \mathbf{q}_i^M \end{bmatrix} + \mathbf{b}_2)\cdots) + \mathbf{b}_L) \\ \hat{r}_{u,i} &= \sigma(\mathbf{h}^T \begin{bmatrix} \phi^{LFM} \\ \phi^{MLP} \end{bmatrix}) \end{aligned} \tag{9}$$

where $\mathbf{p}_u^L$ and $\mathbf{p}_u^M$ represents the user embedding layer of the LFM and MLP models. Similarly, $\mathbf{q}_i^L$ and $\mathbf{q}_i^M$ represent the embedding layer of the item.

D. fusion model pre-training

Due to the non-convexity of the objective function, studies [22] have shown that the gradient-based optimization method can only find the local optimal solution, and the initialization has a great impact on the speed and effect of the deep learning model convergence [23]. So we first train the LFM and GLP models and initialize them with the tanh function until the objective function converges. Finally, the weights are trained at the output layer, and the learning rate is α:

$$\mathbf{r} \leftarrow \begin{bmatrix} \alpha \mathbf{r}^{LFM} \\ (1-\alpha)\mathbf{r}^{MLP} \end{bmatrix} \tag{10}$$

Where $\mathbf{r}^{LFM}$ and $\mathbf{r}^{MLP}$ represent the $\mathbf{r}$ vectors of the pre-training models LFM and MLP, respectively; α is the learning rate parameter for adjusting the two pre-training models. Before training the hybrid model, the parameter α has a value of 0.5, indicating that the LFM and MLP have the same degree of influence on the hybrid model in the initial state.

Training the LFM and MLP from the bottom, both use the Adam activation function, because Adam converges faster than SGD. However, paper [24] have suggested that the results of Adam's final convergence are not as good as SGD. Through further experiments, it is found that the learning rate of Adam is too low, which affects the effective convergence. It is proposed to use the advantage of Adam's fast convergence in the early stage for calculation; later use SGD to find the optimal solution. There are also studies [25] to explore the convergence of the Adam algorithm, through the counter-examples that Adam may not converge in some cases. And Adam needs to save the momentum information for each parameter to update the parameters. In order to achieve the best results and save memory space, after passing the pre-trained parameters into the hybrid model, we use SGD to optimize the objective function.

III. Group-oriented integration strategy

On the basis of obtaining the scores of the predictions of the recommended items by the individual users, the recommendation algorithm for the group users can achieve the group users' prediction scores by integrating the scores of the individual users, so that the recommended results meet the group requirements. The following describes the three common single integration strategies [26], and proposes a new recommendation strategy.

A. Single integration strategy

1)Least Misery: Refers to the evaluation of an item, the entire group's rating is affected by the least like members of the project. Similar to the barrel principle, the capacity of the barrel is determined by the shortest board in the barrel, as shown in equation:

$$\mathbf{G}_j = min_{m_i \in G}\ \mathbf{r}_{ij} \tag{11}$$

where $\mathbf{G}_j$ is the score of the group user to be predicted for item , and $\mathbf{r}_{ij}$ is the score of member user i in group G for item j.

2)Most Pleasure: Contrary to the Least Misery strategy, when an item is evaluated, the entire group's rating is affected by the member who likes the item most, as shown in equation :

$$\mathbf{G}_j = max_{m_i \in G}\ \mathbf{r}_{ij} \tag{12}$$

3)Average Strategy: For the balance between minimum tolerance and maximum satisfaction, when an item is evaluated, the score for the entire group is affected by the average of all predicted scores of the group members. As the formula says:

$$\mathbf{G}_j = average_{m_i \in G}\ \mathbf{r}_{ij} = \frac{1}{n}\sum_{1}^{n} \mathbf{r}_{ij} \tag{13}$$

B. Fusion strategy based on Nash equilibrium

Differences in preferences between users within a group require a certain trade-off, resulting in the highest level of satisfaction with the group. Game Theory is the study of the theory of multiple parties in the game under certain conditions, using the strategy of the relevant parties to implement the corresponding strategy. Therefore, this paper considers the method based on non-cooperative game theory to maximize the average satisfaction of members in the group, and regard the members in the group as game players. The prediction score of items is the strategy choice of game players. In the strategy combination, all players ensure that his strategy is the best when others do not change the strategy. This is the Nash equilibrium. Therefore, the problem of reaching the best group satisfaction can be regarded as the problem of solving the Nash equilibrium. Although users will have their own choices, more than one Nash equilibrium will always exist.

Let set $\mathbf{G}$ be a collection of all groups with at least two members, all from the user set $\mathbf{U}$, so $|\mathbf{G}| = 2^k - k - 1$. The members of the set $\mathbf{G}$ are g, and $|g|$ represents the number of members in the combination of g.

If $r(u, i)$=0 is that the user u does not score the item i, let $\hat{r}(u, i)$ be the estimated user u's score for the item i.

Items that may be recommended for a group belong to a collection $\mathbf{N}$ of items that have not been evaluated by members of any group. These items were obtained from a list of items that were not evaluated by members of the group:

$$\mathbf{N} = \bigcap_{u \in g} \{i | \hat{r}(u, i) = \varnothing, i \in \mathbf{I}\} \tag{14}$$

The group recommendation problem is modeled as a non-cooperative game, in which the items recommended to the group are items in the Nash equilibrium. Common games are tuples (m, A, f), where:

- m is the number of players (group members).
- $\mathbf{A}$ is a collection of players' actions (possible value for rating the item). Vector $a_i \in \mathbf{A}$ is the profile of the action.
- f is a function used to calculate the player's score deviation payoff.

We fill a set of possible actions into the action set $\mathbf{RA}$ for the player to use. These actions are all items with higher predicted values. Each player's payoff function considers the union of items selected as game actions for each player j:

$$\mathbf{RA} = \bigcup_{j=1}^{m} a_j \tag{15}$$

The payoff function calculates the predicted satisfaction of member u, calculated by the actions selected by all players in a given game strategy:

$$f(u) = \frac{\sum_{i \in \mathbf{RA}} \hat{r}(u, i)}{|\mathbf{RA}|} \tag{16}$$

Nash Equilibrium is a stable group strategy that considers that other players do not change their satisfaction, so the current player has no incentive to change their satisfaction. And when there are multiple Nash equilibriums in this group, the best way to get payoff is by reconciling the mean function:

$$ha(x) = \frac{n}{\sum_{i=1}^{n} \frac{1}{x_i}} \tag{17}$$

Where x_i represents the value of the group member's preference for all items of each Nash equilibrium. It preferentially selects the value of the smaller variance.

TABLE I
DATA STATICS TABLE

File name	Size
rec_log_train.txt	1.99G
user_profile.txt	55.8M
Item.txt	1.18M
user_action.txt	217M
user_sns.txt	740M
user_key_word.txt	182M

IV. EXPERIMENTS

A. Experimental data set

The dataset for all experiments in this paper is from the Tencent Weibo dataset in KDD CUP 2012 Track1.It is from the 30-day Weibo record from 2011.11.11 to 2011.12.11. The selected 1,392,973 users recorded 73,209,277 historical scores for 4,710 items. The size of the data set is approximately 3.8G. The data set content includes information such as social relationships and user item characteristics. All the personal information related to the user in the data is protected by encryption. For example, the user name and the product name are replaced by a number. The list of experimental data set files is shown in Table 1.

Among the six files shown in Table 1, the score value in rec_log_train.txt indicates the recommendation result. The main record is the user's explicit score on the item. The data format are UserId, ItemId, Result, Unix-timestamp, and the score is only three. Values, 1 for acceptance, -1 for rejection, and 0 for unknown. User-item-score 3-tuple information is taken based on the experimental data requirements.

In the scoring data set (rec_log_train.tx), the number of records accepted by the user is only 5,253,828, while the total number of records is 73,209,277. The negative samples are the majority, and there are a large number of items that are not discovered or accepted by the user.And quite a few users are not very active.Therefore, we introduce the user's social behavior into the prediction score to solve the sparseness of the data, thereby improving the accuracy of the recommendation. The user's behavior weights (user@, number of forwarding and comments) recorded in the user behavior data set (user_action.txt) are accumulated, and finally the user behavior weights are ranked from large to small, thereby filtering out active users. A total of 61083 people were active with a degree of activity greater than 180. Finally, in the original score data set, the scores of these active users were screened and tested, and a total of 6,741,799 records

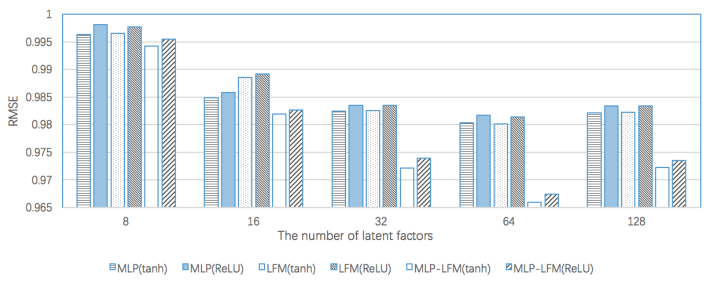

Fig. 2. Overall AVE_RMSE comparisons.

were recorded. All the experiments in this paper will use the records selected in the above rec_log_train.txt as the data set of the whole experiment, the size is 197M. Then, the data set is randomly selected and divided into a training set and a test set according to the ratio of the training set and the test set of 7:3. Finally, the final experimental results are given by means of cross-validation. The final result of each set of experiments is the average of 30 experimental results.

B. Recommended model evaluation index

This article uses the root mean square error(RMSE) as an indicator of evaluation. The RMSE formula is as follows

$$RMSE = \sqrt{\frac{\sum_{u,i \in TestSet} \left(\mathbf{r}_{u,i} - \hat{\mathbf{r}}_{u,i}\right)^2}{Test\_Set}} \tag{18}$$

where RMSE represents the square root of the ratio of the sum of squared differences between all predicted scores and true scores in the test set to the number of tests. In the above formula, $\mathbf{r}_{u,i}$ is the real score, $\hat{\mathbf{r}}_{u,i}$ is the prediction score, and Test_Set is the test set. So from the formula, the smaller the RMSE, the more accurate the recommendation.

C. Analysis of results

The experiment is divided into two parts, an experiment for multiple recommendation models for individuals, and a group recommendation experiment for groups using different fusion strategies. The individual recommendation models are: matrix factorization method LFM based on latent factor models, neural network based MLP, and a mixture of two models (labeled LFM-MLP). The experimental results obtained are shown in Fig. 2:

From Fig. 2, the previous hypothesis can be confirmed. The scores of the dataset used in this paper are only three values (-1, 0 and 1). The activation function tanh performs better than the activation function ReLU in each model.

Because the large abstraction factor leads to overfitting, the MLP model used for the experiment is a three-layer latent layer tower structure with a minimum size of 8. Under the same number of latent factors, the recommended accuracy of the hybrid recommendation model LFM_MLP is the highest, and the recommended accuracy is 2.83% higher than the traditional LFM model. And the improved LFM is slightly more accurate than the MLP, but the performance of the MLP can be improved by adding a deeper latent layer, which shows the performance of the three layers. When the latent factor is relatively small, the value of AVE_RMSE is relatively high, and the fitting effect is not ideal. However, as the number of latent factors increases, the accuracy becomes closer and closer, the fitting effect becomes better and better, and continues with the latent The factor increases and the curve gets closer and closer.

The experiment recommended by the group is to find the users in the same group through the user classification information in the item.txt file in the data set, thereby obtaining the set G of the group. However, regarding the selection of parameters for group recommendation, this paper adopts two methods for comparison experiments: one is to group the users after completing the recommendation prediction for all individuals, and use the global prediction score matrix; the other is to Grouping, the root then uses the hybrid model to solve the predicted scores of the user's collection of items in the group, and then performs group recommendation. Finally, K items are recommended to the group, and a recommendation list Top(K) is generated to calculate the root mean square error.

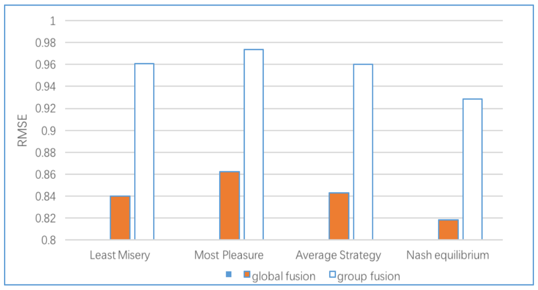

Fig. 3. The RMSE of global decomposition and group decomposition.

The experimental results are shown in Fig. 3. By analyzing the data of the group recommendation experiments using four fusion strategies, the Nash equilibrium performance is the best among the different schemes, and the highest satisfaction (Most Pleasure) performance is the most. Unsatisfactory, the prediction effect of the Average Strategy is slightly lower than the Least Misery in the group factorization, but the performance in the global factorization is better than the Least Misery, so in practical applications, the data needs to be analyzed first. Set and then select a single fusion strategy. In either case, the Nash equilibrium is more accurate than the recommendation of any single strategy: in an environment where the group is decomposed and merged, the performance of the prediction is 2.95% higher than the mean strategy; In the intra-group integration scenario, the predicted performance was 3.27% higher than the Average Strategy. This verifies the effectiveness and accuracy of the algorithm.

V. CONCLUSION AND FUTURE WORK

Recommendations for group-based recommendations and in combination with neural networks are receiving increasing attention from researchers.Based on the traditional LFM model, combined with multi-layer perceptron, a hybrid model is proposed, which uses the tanh function to initialize the matrix of latent factor space.First, the two sub-models are quickly pre-trained by Adam, and then the output layers of the two models are fully connected.The random gradient descent method is used to optimize the parameters, and the individual-oriented MF recommendation algorithm is improved.On this basis, using the Nash Equilibrium-based group fusion strategy, the three single fusion strategies are compared in the intra-group factorization and global factorization.Experiments have shown that this strategy recommends better performance on the data sets used in the text.

Further work on this article will be considered in the future.When the matrix factorization model is first established, the social behavior and social relationship between the users added in this paper occur within a certain period of time, and the influence of timestamps is not considered.Second, when using neural network collaborative filtering, a deeper MLP network structure is used to obtain a more abstract relationship, which may be closer to the true score by calculation.Finally, when calculating the error objective function, we use the point-by-point logarithmic loss function. If we use the pairwise loss function instead, it may achieve better results in the group recommendation.These are our further research content.

Acknowledgment

The authors thank the anonymous reviewers for their valuable comments, which are beneficial to the authors' thoughts on recommendation systems and the revision of the paper.

References

[1] R. Mishra, P. Kumar and B. Bhasker, ”A web recommendation system considering sequential information,” Decision Support Systems, vol. 75, pp. 1-10, 2015.

[2] M. Jiang et al, ”Social contextual recommendation,” in 2012, . DOI: 10.1145/2396761.2396771.

[3] M. Pham et al, ”A Clustering Approach for Collaborative Filtering Recommendation Using Social Network Analysis,” Journal of Universal Computer Science, vol. 17, (4), pp. 583-604, 2011.

[4] ZHOU Meng, ZHU Fu-xi.A. ”Crowd Orientation Algorithm Based on Neighbor Selection Strategy ”, Computer Research and Development,, vol. 54 (7): 1465-1476, 2017

[5] C. Fang et al, ”Diversified recommendation method combining topic model and random walk,” Multimedia Tools and Applications, vol. 77, (4), pp. 4355-4378, 2018.

[6] Z. Drmač, ”Algorithm 977: A QR–Preconditioned QR SVD Method for Computing the SVD with High Accuracy,” ACM Transactions on Mathematical Software (TOMS), vol. 44, (1), pp. 1-30, 2017;2018;.

[7] Liu J, Yang J. Social recommendation based on SVD + + and behavior analysis[J]. Journal of Computer Applications, 2013.

[8] R. Collobert and J. Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In ICML, pages 160–167, 2008.

[9] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In CVPR, 2016.

[10] R. Hong, Z. Hu, L. Liu, M. Wang, S. Yan, and Q. Tian.Understanding blooming human groups in social networks.IEEE Transactions on Multimedia, 17(11):1980–1988, 2015.

[11] H. Zhang, Y. Yang, H. Luan, S. Yang, and T.-S. Chua. Start from scratch: Towards automatically identifying, modeling, and naming visual attributes. In MM, pages 187–196, 2014.

[12] A. Van den Oord, S. Dieleman, and B. Schrauwen. Deep content-based music recommendation. In NIPS, pages 2643–2651, 2013.

[13] H. Wang, N. Wang, and D.-Y. Yeung. Collaborative deep learning for recommender systems. In KDD, pages 1235–1244, 2015.

[14] F. Zhang, N. J. Yuan, D. Lian, X. Xie, and W.-Y. Ma. Collaborative knowledge base embedding for recommender systems. In KDD, pages 353–362, 2016.

[15] He, Xiangnan, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. ”Neural collaborative filtering.” In Proceedings of the 26th International Conference on World Wide Web, pp. 173-182. International World Wide Web Conferences Steering Committee, 2017.

[16] Zhang S , Yao L , Sun A , et al. Deep Learning based Recommender System: A Survey and New Perspectives[J]. 2017.

[17] Y. Song et al, ”A novel group recommendation algorithm with collaborative filtering,” in 2013, . DOI: 10.1109/SocialCom.2013.138.

[18] X. Zeng et al, ”Parallelization of latent group model for group recommendation algorithm,” in 2016, . DOI: 10.1109/DSC.2016.54.

[19] K. Hornik, M. Stinchcombe, and H. White, ”Multilayer feedforward networks are universal approximators,” Neural Networks, vol.2, pp.359–366, 1989.

[20] N. Srivastava and R. R, Salakhutdinov, ”Multimodal learning with deep boltzmann machines,” In NIPS, pp. 2222–2230, 2012.

[21] Kingma D, Ba J , ”Adam: A Method for Stochastic Optimization”, Computer Science, pp.126-131, 2014.

[22] V. Srinivasan, A. Sankar and V. Balasubramanian, ”ADINE: An adaptive momentum method for stochastic gradient descent,” in 2018, . DOI: 10.1145/3152494.3152515.

[23] D. Erhan et al, ”Why Does Unsupervised Pre-training Help Deep Learning?” Journal of Machine Learning Research, vol. 11, pp. 625-660, 2010.

[24] Keskar N S , Socher R , ”Improving Generalization Performance by Switching from Adam to SGD”, in press, 2017.

[25] Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar, ”On the convergence of adam and beyond”, International Conference on Learning Representations, in press, 2018.

[26] L. Quijano-Sanchez et al, ”Social Factors in Group Recommender Systems,” Acm Transactions on Intelligent Systems and Technology, vol. 4, (1), pp. 1-30, 2013.

[27] L. Xu et al, ”User Participation in Collaborative Filtering-Based Recommendation Systems: A Game Theoretic Approach,” IEEE Transactions on Cybernetics, pp. 1-14, 2018.