Available online at www.ijpe-online.com

International Journal of Performability Engineering

vol. 14, no. 1, January 2018, pp. 26-36
DOI: 10.23940/ijpe.18.01.p4.2636

ICFLSB: An Improved Collaborative Filtering Algorithm based on Latent Semantic and Bayesian

Yun Wu[a,*], Ren Qian[a], Xiaofei Dong[b], Yiqiao Li[a], and Xinwei Niu[c]

[a]*College of Computer Science and Technology, Guizhou University, Guiyang, China*
[b]*School of Mechanical Engineering, Yancheng Institute of Technology, Yancheng, China*
[c]*School of Engineering, Penn State Behrend, Erie, Pennsylvania, United States*

Abstract

In the process of user-based collaborative filtering algorithm, finding similar users effectively plays a crucial role in obtaining a high recommendation accuracy. The original rating matrix is very sparse, resulting in similarity information loss during similarity calculating and degrading the efficiency of similar users extracting. To tackle this problem, we propose an improved collaborative filtering algorithm based on Latent Semantic and Bayesian (ICFLSB). ICFLSB first utilizes Latent Semantic to extract meaningful features in the original rating matrix. Then, we establish a Bayesian model based on these extracted features to predict items which users have not rated but may be interested. Further, we fill the sparse original rating matrix with these predicted items and find similar users. After that, we adopt the collaborative filtering algorithm to conduct recommendations. Experiments show that ICFLSB proposed in this paper has a better recommendation performance than the traditional collaborative filtering algorithm. In particular, the evaluation results demonstrate that our ICFLSB can achieve a 2.152% higher and 1.152% higher on recommendation accuracy and recall rate respectively when compared to the traditional collaborative filtering algorithm.

Keywords: latent semantic; Bayesian; sparse matrix; collaborative filtering; recommender systems

(Submitted on November 1, 2017; Revised on December 10, 2017; Accepted on December 20, 2017)

1. Introduction

Collaborative filtering algorithm is one of the most widely used recommendation algorithms today. It tries to discover users with similar interests based on the users' ratings on items and then recommend items to users with the same interests. Collaborative filtering algorithm is very easy to use and has a relatively satisfying efficiency. Thus, there is much research on the collaborative filtering algorithm. Banas et al. [3] integrated collaborating filtering algorithm with entropy to improve the recommendation performance by utilizing entropy on the original users rating matrix to strengthen users distinguish. Kaleli et al. [9] took the utmost of users' configuration information, which aggregated users by their configuration information and then searched for similar users in each user-cluster. Although the methods to improve collaborative filtering algorithm are very different, they have an identical ultimate goal: to find similar users with higher accuracy. Searching for similar users is the core factor that will affect the recommendation performance. Therefore, to search for similar users with higher accuracy and efficiency has become a hot research topic. Searching for similar users directly on the original users rating matrix is difficult and often impossible because the original users rating matrix is usually extremely sparse, as there are tens of thousands of items and most of them have no rating scores.

In order to mitigate the performance affection of matrix sparsity, we propose an improved collaborative filtering algorithm based on Latent Semantic and Bayesian (ICFLSB). Firstly, ICFLSB will extract the meaningful features and eliminate the redundant and invalid features by taking the advantage of Latent Semantic. Then, a Bayesian model will be built up on these extracted features and this model will be used to predict users' interests and favorites to fill up the sparse

* Corresponding author.
E-mail address: wuyun_v@126.com

matrix and facilitate the similar users' generation. Ultimately, the collaborative filtering algorithm will be applied to the similar users matrix in order to conduct recommendations.

2. Related Works

As the sparse problem has badly affected the performance of collaborative filtering algorithm, many strategies have been proposed to solve this thorny problem. Cui et al. [9] introduced the Jaccard similarity into the cosine similarity as an additional factor because authors believed that sparsity would cause great error deviation in the process of cosine similarity calculation. This method, however, did not essentially solve the sparse problem of the rating matrix and its effect was very limited. There are researches using clustering method to solve this problem [6]. Hong et al. [8] clustered the rows and columns of the rating matrix simultaneously to get a series of small matrixes with relatively high similarity and then used RBF to predict the rating. Liu et al. [11] utilized item attributes vector to cluster items and users interest vector to cluster users, reducing the sparsity of the rating matrix effectively. Similar works which used dimensional reduction to solve the sparse problem can also be seen in [7,4]. However, the dimensional reduction method not only reduces the sparsity of the rating matrix, but also loses important information during the process of dimensional reduction, resulting in more difficulty in determining similar users with little improvement in performance. Faced with the fact that the similarity calculation and the dimensional reduction cannot achieve the desired results, researchers proposed a completely new method named filling method to avoid the flaws in the previous methods [17]. Shi et al. [13] utilized jaccard similarity of information entropy to search for similar items and predicted each item for each user, then used these scores to fill the matrix. Liu et al. [11] established a time decay model of interest intensity and then filled the matrix with the predictive rate. Cetintas et al. [14] filled the matrix by using deep learning techniques for users' micro-blog content and pictures. Even though the filling methods listed above are able to solve the sparse problem to a certain degree, they are extremely complex and thus their efficiency is unsatisfied and their scalability is limited. To address this knotty problem, we propose a new filling method which utilizes the Bayesian model to predict the probability of users' favourite items.

The Bayesian model is widely used for its simple modelling and high efficiency. The recommendation algorithm based on the Bayesian model has been widely applied to various fields, such as the recommended baggage for travel customers [13] and personalized recommendation for manufacturing [21]. Guo et al. [5] proposed a method of Bayesian similarity measure which was based on the dirichlet distribution of the direction and length of the vector. Wang et al. [15] introduced a collaborative deep learning method which built the hierarchical Bayesian model to complete the deep learning for the content and simultaneously the results of collaborative filtering were used as feedback to train the model. Cong et al. [20] built the Bayesian model based on four attributes: Who, Where, When, and What, which were obtained by the semantic analysis of content. These works give us sufficient guidance. However, there is a disruptive problem existing in them: subjective will becomes a passive factor in choosing attributes to build Bayesian models. For example, the attributes of music can be defined as rock, classical, sad, cheerful, etc., anything we humans define it; the artificial subjectivity is obvious here. Therefore, it cannot reflect the real definition of the item in different users, and this deviation will without a doubt affect the recommendation performance.

In order to solve the subjectivity problem in choosing attributes, we propose to use the latent semantic model to define the attributes of items. The latent semantic model will map the items space and user space into the same space and the feature attributes of items in the same space can better represent the objective attributes of the items in different users. This avoids the subjectivity problem in choosing attributes. The latent semantic model plays a significant role in the recommendation system [18,19], and there are many research works on it in recent years. Liu et al. [16] proposed a method which combined the SVD algorithm and the collaborative filtering algorithm to solve the sparse problem. Ba et al. [2] combined the latent semantic with the collaborative filtering algorithm, solving the cold start problem and improving recommendation performance and scalability. Although these algorithms of the latent semantic model improved the recommendation performance to a certain extent, they failed to take the attributes generated in the latent semantic model into account. We thereby propose a method to use the features obtained from the latent semantic model to define the items' properties and further avoid humans' subjective will.

Consequently, in order to solve the sparse problem of the original rating matrix, we propose a Bayesian model to calculate users' preferences probability for each item and then use this probability to fill the original rating matrix. There is, however, a subjective will problem in the process of choosing features to establish the Bayesian model, which will degrade the result confidence dramatically. Faced with this fact, we further propose utilizing the latent semantic model to extract the meaningful features and using these features to build a Bayesian model to avoid the subjective will problem, increase the efficiency of finding similar users, strengthen the recommendation results confidence and improve the recommendation performance.

3. ICFLSB Model

3.1. Latent Semantic Model

In order to establish Bayesian model, appropriate attributes of items and users should be selected. Selecting these required attributes by using the latent semantic model can avoid the subjective will problem. Since its advent, the latent semantic model has caught researchers' attention and many models and algorithms have been developed, such as PLSA, LDA, latent class model, latent topic model, matrix factorization and so on. These technologies and methods are essentially similar and can be applied to the recommendation system. In this paper, we utilize the LFM model to extract features. The LFM formula used to calculate the degree of interest of user u to i can be described as below:

$$preference(u, i) = r_{ui} = p_u^T q_i = \sum_{f=1}^{F} p_{uk} q_{ki} \tag{1}$$

Where p_{uk} and q_{ki} in the formula are model parameters. The p_{uk} denotes the relationship between user U 's interest and feature K ; the q_{ki} denotes the relationship between feature K and item i. The purpose of LFM is to calculate these two parameters. By optimizing the following loss function to find the most appropriate parameters p and q :

$$c(p,q) = \sum_{(u,i)\in Train} (r_{ui} - \hat{r}_{ui}) = \sum_{(u,i)\in Train} \left(r_{ui} - \sum_{f=1}^{F} p_{uf} q_{if} \right)^2 + \lambda \| p_u \|^2 + \lambda \| q_i \|^2 \tag{2}$$

Where $\lambda \| p_u \|^2 + \lambda \| q_i \|^2$ in the formula is a regularization term used to prevent overfitting. To minimize the above loss function, it uses the random gradient descent method. This optimization algorithm derives p and q respectively:

$$\frac{\partial C}{\partial p_{uf}} = -2q_{ik} + 2\lambda p_{uk} \tag{3}$$

$$\frac{\partial C}{\partial p_{if}} = -2q_{uk} + 2\lambda p_{ik} \tag{4}$$

Then, according to the random gradient descent method, the parameters will move forward along the steepest direction, and then get the following recurrence formula:

$$p_{uf} = p_{uf} + \alpha(q_{ik} - \lambda p_{uk}) \tag{5}$$

$$q_{if} = q_{if} + \alpha(p_{uk} - \lambda q_{ik}) \tag{6}$$

Moreover, it defines a threshold t, and its value will be set as 10% of the sum of all features values of item according to empirical setting. Therefore, $t = 0.1\sum_{f=1}^{F} p_{uf}$. When the calculating result is greater than this value, it indicates that the item or user has this attribute; otherwise, the item or user does not contain this attribute. This methodology can help remove some noise attributes as well as reduce the computational complexity.

3.2. Bayesian Model

The Bayesian model is established on each user and is then used to predict the favourite or preference probability of items which users might be interested in but have not rated. The original rating matrix is then filled with these favourite or preference probabilities.

In this paper, we use the films which have been rated by users as a priori probability to train the model. The rated films

include users' favourites and users' dislikes. According to this fact, we divide the films into two categories: $Class_A$ includes the films that are users' favourites and $Class_B$ includes the films that are users' dislikes. Then, we calculate the numbers of attributes in each film in each class. For instance, if users' favourite $film_1$ has attribute a_1 and attribute a_2, then the number of attributes a1 and a2 in Class A should be $NumA(a_1) + 1$, $NumA(a_2) + 1$. The total number of attributes should be $total_a + 2$. Then the probability of an attribute in the film that the user likes can be described below:

$$P(a_i \mid A) = \frac{Num_A(a_i)}{total_a} \tag{7}$$

Similarly, the probability of an attribute in the film that the user dislikes can be described below:

$$P(a_i \mid B) = \frac{Num(a_i)}{total_b} \tag{8}$$

Under this context, we further create two dictionaries $Dict_A$ and $Dict_B$ to store these probabilities.

According to the Bayesian formula, when a movie contains the attribute a_i, the probability of the users' favourite can be described below:

$$P(A \mid a_i) = \frac{P(a_i \mid A) \times P(A)}{P(a_i \mid A) \times P(A) + P(a_i \mid B) \times P(B)} \tag{9}$$

In the real world, a film lasts longer than a song. Hence, it is highly unlikely that a user will spend many minutes on a film in which he/she shows little interest. According to this practical situation, we regard the films which users have not watched as negative samples and take the average of all these probabilities. Then, we obtain the probability $P(A) = 0.01$, which denotes the favourite or preference probability of a user for a randomly chosen film; and the dislike probability $P(B) = 0.99$, which denotes the dislike probability of a user for a randomly chosen film.

Based on the results we get from the Bayesian training model, we can further conduct users' preferences prediction. In particular, the probability that a user likes a film with attributes $(a_1, a_2, \cdots, a_n)$ is based on the composite probability formula:

$$P_m(A \mid a_1, a_2, \cdots, a_n) = \frac{P(A \mid a_1)P(A \mid a_2)\cdots P(A \mid a_n)}{P(A \mid a_1)P(A \mid a_2)\cdots P(A \mid a_n) + [1 - P(A \mid a_1)][1 - P(A \mid a_2)]\cdots[1 - P(A \mid a_n)]} \tag{10}$$

3.3. Collaborative Filtering Algorithm and Similarity Measurement

The principle of user-based collaborative filtering algorithm is to find similar users and then recommend items to them. We can get the perceptible knowledge of user-based collaborative filtering algorithm in Table 1 in which the integer 1 represents the users' preference for items.

Table 1. The Principle of User-based Collaborative Filtering Algorithm

user/item	Item A	Item B	Item C	Item D
User A	1		1	recommendation
User B		1		
User C	1		1	1

Diving a little deeper in Table 1, we can figure out that the $user_A$ is similar to the C, and deservedly the C's favourite $item_D$ will be recommended to $user_A$.

The core part of user-based collaborative filtering algorithm is to find similar users. The common methods for calculating the similarity of users include Cosine similarity, Jaccard distance, Euclidean distance, etc. The Cosine similarity calculates the vector cosine of user A and user B to determine their similarity and is one of the most widely used methods. Its formula can be described as below:

$$\text{sim}(\vec{A},\vec{B}) = \cos(\vec{A},\vec{B}) = \frac{\vec{A}\cdot\vec{B}}{\|\vec{A}\| * \|\vec{B}\|} \tag{11}$$

Where $vector_A$ and $vector_B$ represent the $user_A$ and $user_B$ multidimensional vectors respectively. These dimensions are items that have been evaluated by users.

The Jaccard distance regards the attributes of $user_A$ and $user_B$ as a collection. These attributes are the items that have been evaluated by users. The similarity formula is shown as below:

$$sim(A,B) = |A \cap B| / |A \cup B| \tag{12}$$

Euclidean distance is used to calculate the distance of every dimension between users. It calculates the sum of squares and arithmetic square root to define the similarity between users, as shown below:

$$sim(\vec{A},\vec{B}) = \sqrt{\sum_{i=1}^{n}(X_i - Y_i)^2} \tag{13}$$

However, the methods above present an inherent problem for the sparse problem in the original rating matrix. It will lose similar users and their information because it fails to consider the many items that have not been evaluated by users. Even though $user_A$ and $user_B$ are very similar, their similarity information will be lost if they rarely rate the same films.

3.4. Improved User-based Collaborative Filtering Algorithm

First and foremost, we utilize the LFM model to decompose the original rating matrix into two sub-matrixes A and B. We initialize a user matrix P and an item matrix Q and then assume there are F features and set their values as 1. Then, we iteratively optimize the values of these features by using the gradient descent method. Finally, we use the threshold t to filter features: we set those features whose values are less than t to zero, which means users or items do not contain this feature.

After obtaining the features of all users and items, set up two dictionaries Dict_like and Dict_dislike. The dictionary structure is {attribute name: number of occurrences}. The Dict_like dictionary records the features occurrences times in the users' preference class, while the Dict_dislike dictionary records the features occurrences times in the users' dislike class. Assume $film_1$ is in users' preference class and has two features a_1 and a_2, then the time of a_1 and a_1 in Dict_like should add 1 respectively. The preference and dislike statistics are finished according to historical records of users' ratings.

After the statistics of dictionary Dict_like and Dict_dislike, we further use the formula (7) and (8) to calculate $P(a_i \mid A)$ and $P(a_i \mid B)$ and store their values in two arrays named Array_A and Array_B. We apply formula (9) to Array_A and Array_B to calculate $P(A \mid a_m)$, which will be stored in another array named Array_P_A. For a user, getting the set of $P(A \mid a_m)$ means completing the Bayesian model training for this user. The next step is to calculate the favourite or preference probability of users for items that they have not rated.

Finally, according to the features (a_1,a_2,..a_n) of each film, we use the formula (10) to calculate the preference probability of users. The calculated preference probability will be directly filled into the sparse matrix. In this case, each user will have

a preference probability on each film which is not evaluated. Therefore, it transforms the sparse matrix into a dense matrix, which can help find similar users more efficiently. In this paper, we use the cosine similarity to calculate the similarity between users.

Therefore, the main steps of improved user-based collaborative filtering algorithm can be described through pseudocode as below:

(1) LFM Algorithm

Input: R , F , t .(R is the rank of base matrix, F is the number of feature, t is threshold).

Output: P , Q .(P is user matrix, Q is item matrix).

Step 1: Initialize the user matrix P and the item matrix Q ;

Step 2: Obtain the optimization feature values of P and Q by using $p_{uf} = p_{uf} + \alpha(q_{ik} - \lambda p_{uk})$ and $q_{if} = q_{if} + \alpha(p_{uk} - \lambda q_{ik})$;

Step 3: Use the threshold t to filter features: set those features whose values are less than t to zero;

Step 4: End.

(2) Bayesian Algorithm and User-based Collaborative Filtering Algorithm

Input: user matrix P , item matrix Q , rank of base matrix R , dictionaries Dict_like and Dict_dislike.

Output: Precision, and Recall.

Step 1: Set up two dictionaries Dict_like and Dict_dislike;

Step 2: Calculate the $P(a_i \mid A)$ and $P(a_i \mid B)$ after the statistics of dictionary Dict_like and Dict_dislike;

Step 3: Calculate the $P(A \mid a_m)$ based on the results of $P(a_i \mid A)$ and $P(a_i \mid B)$;

Step 4: Calculate the preference probability of users and fill them into the rank of base matrix R ;

Step 5: Repeat Step 1 to Step 3 until all the users are processed; get a dense matrix R ;

Step 6: Calculate the similarity of users on the dense matrix R ;

Step 7: Conduct recommendations by utilizing the user-based collaborative filtering algorithm and calculate the recommendation accuracy and recall rate.

Step 8: End.

4. Experiment

4.1. Experimental Environment and Procedures

The experimental computer has 2GB memory, 500GB hard disk, and Core i5 2.5GHZ CPU. The experimental dataset is MovieLens which contains users' rating of movies, the rating score ranging from 1 to 5. It includes 10,000 ratings data of 3,900 movies produced by 6040 individuals. It also includes all the features of the 3,900 movies. In this paper, however, we will not consider all the features in MovieLens; instead, we use LFM to extract the appropriate features to build the

Bayesian model.

(1) We randomly select part of these data as our experimental data;
(2) We choose 1/8 of these users as test users and the remaining 7/8 as the training users;
(3) We use cosine similarity to calculate the users' similarity and use the accuracy and recall rate as the evaluation metrics; we also subtract 0.1 as the offset from the similarity result;
(4) The experiment selects the similar neighbour users with 5 users as a gradient, and the number ranges from 10 to 70;
(5) We conduct recommendations and calculate the recommendation accuracy and recall rate.

4.2. Experimental Evaluation Metrics

In order to test the validity of the method proposed in this paper， we adopt two evaluation metrics: Accuracy and Recall Rate.

The Precision denotes the percentage ratio of items interested by users and the total recommendation items. If the recommendation list for user u contains N items (denoted R (U)) and the preference collection of items of this user is T (U), then the Accuracy can be described as below:

$$Accuracy = \frac{\sum_u |R(u) \cap T(u)|}{\sum_u |R(u)|} \tag{14}$$

The Recall Ratio is the percentage of the user's preference items in the recommendation list and the user's favourite items, as described below:

$$Recall = \frac{\sum_u |R(u) \cap T(u)|}{\sum_u |T(u)|} \tag{15}$$

4.3. Comparison and Analysis of Experimental Results

We randomly select 70 users to compare the recommendation performance of our improved collaborative filtering algorithm based on latent semantic and Bayesian (ICFLSB) to that of the traditional user-based collaborative filtering algorithm (CF). Dimensionality reduction collaborative filtering (DRCF) is proposed in [16] and collaborative filtering algorithm based on information entropy weighting (IEWCF) is proposed in [20]. The results are shown in Figure 1 and Figure 2 respectively:

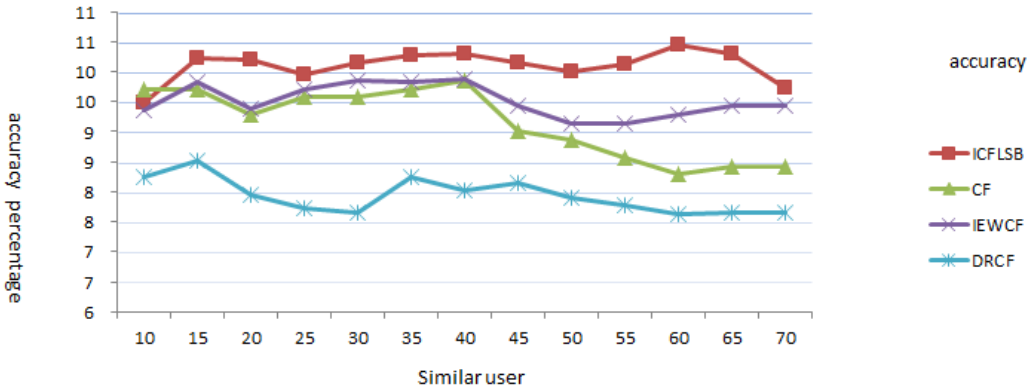

Figure 1. Comparison of Accuracy for 70 Users

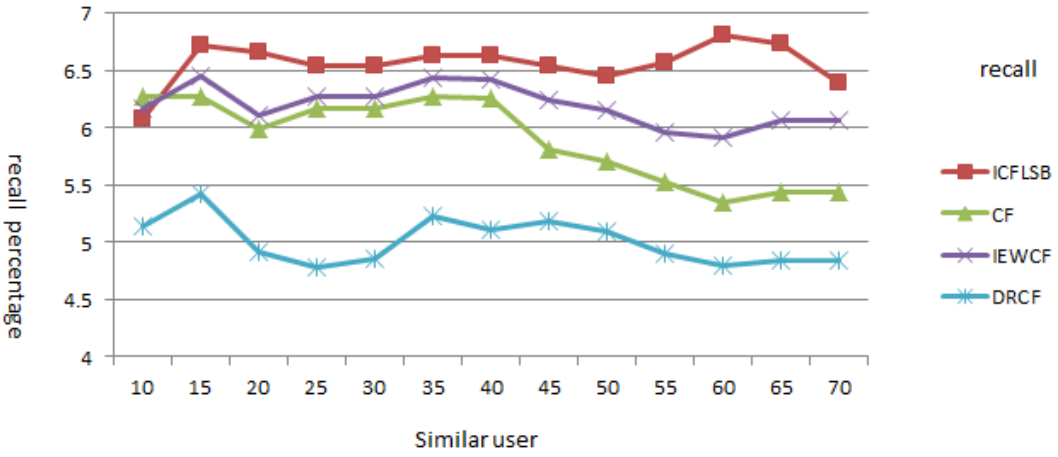

Figure 2. Comparison of Recall Rate of 70 Users

When we randomly select 70 users, compared with the traditional collaborative filtering algorithm and the IEWCF, the ICFLSB proposed in this paper does not show obvious advantages in the accuracy and recall rate when the number of similar user neighbours is set 10. However, when more similar users are selected, the ICFLSB has a dominant advantage. Compared to the traditional collaborative filtering algorithm, the accuracy and recall rate of ICFLSB have increased to the highest level on 60 similar users. Accuracy and recall rate have increased by 2.152% and by 1.152% respectively. Compared to the IEWCF, the accuracy and recall rate of ICFLSB have increased by 1.866% and by 0.896% respectively. Even though the accuracy rate and recall rate of all the algorithms have shown a decline when more similar users are selected, ICFLSB still has a better performance than traditional collaborative filtering algorithm and IEWCF.

Similarity, when we select all the users, compared with the traditional collaborative filtering algorithm, accuracy and recall rate of ICFLSB have increased by 2.152% and by 1.152% respectively. Compared with the IEWCF, accuracy and recall rate of ICFLSB have increased by 0.285% and by 0.206% respectively.

Furthermore, the results indicate that when the user base is too small, the algorithm performance will be degraded. This is because when the user base is too small, the dissimilar users will be added wrongly. Also, it is obvious that DRCF has a lower accuracy and recall rate as information loss in the process of dimensional reduction.

We further consider the impact of features generated by the latent semantic model on ICFLSB. According to the previous experiments, we get the optimization similar users 60. Then, we increase the numbers of features in F arithmetically. Then, we calculate the accuracy and recall rate again, as shown in Table 2.

Table 2. Accuracy and Recall Rate(ICFLSB) with the Number of features change in F

Number of Similar users K	**Number of features in F**	**Accuracy**	**Recall rate**
K=60	10	10.132	6.593
	20	10.276	6.686
	30	10.332	6.755
	40	10.488	6.818
	50	10.488	6.818
	60	10.488	6.818

Table 2 shows that the accuracy and recall rate of the ICFLSB increases as the number of features in F increases when the K is fixed as 60. This is because users and items in latent semantic space cannot be distinguished exactly when there are not enough features. Results indicate that when the number of features is 40, the ICFLSB performs the best. When the features number increases again, there is no obvious performance improvement. This fact means that 40 features can represent items and users perfectly in latent semantic space. The selection of more features will not help performance improvement but will instead increase redundancy and even degrade efficiency.

Based on this “discovery”, we set the number of features in F as 40 and re-do the first experiment again. In particular, we select 70 users randomly and compare the performances of ICFLSB, CF, DRCF, and IEWCF again. We still use the evaluation metrics: Accuracy and Recall Rate. The detail results can be seen in Figure 3 and Figure 4.

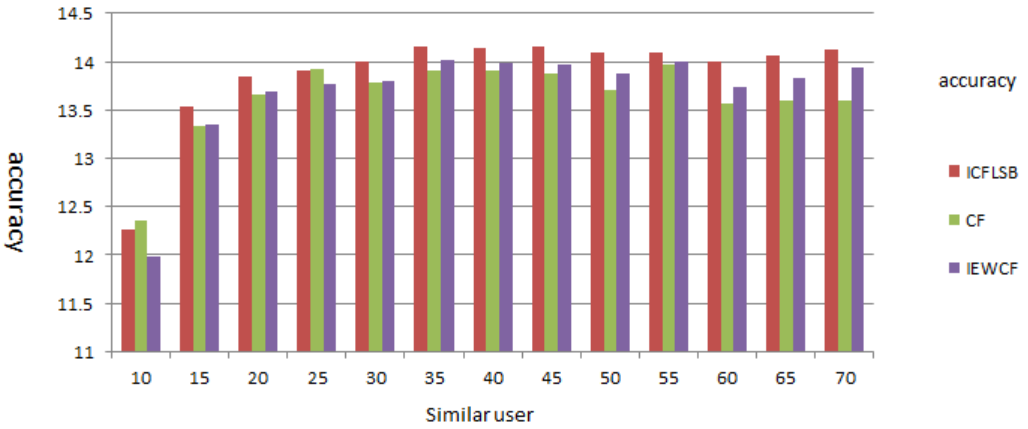

Figure 3. Comparison of accuracy for 70 users

From Figure 3 and Figure 4, we can see that the accuracy and recall rate increase as the user base increases. This is because the probability of searching for similar users will increase when the user base increases and therefore the algorithms performances will be improved. When the number of selected users increases, the performance of ICFLSB increases steadily. Specifically, compared with the CF, the accuracy and recall rate have increased to the highest level on 70 similar users. Accuracy and recall rate have increased by 0.531% and by 0.187% respectively. Compared with the performance of IEWCF, the accuracy and recall rate of ICFLSB have increased by 0.314% and by 0.157% respectively. The results indicate that ICFLSB can find similar users and avoid the sparse problem in original matrix effectively. Also, with the help of the latent semantic model, ICFLSB avoids the subjective will problem, resulting in a higher recommendation accuracy and recall rate and a more steady performance.

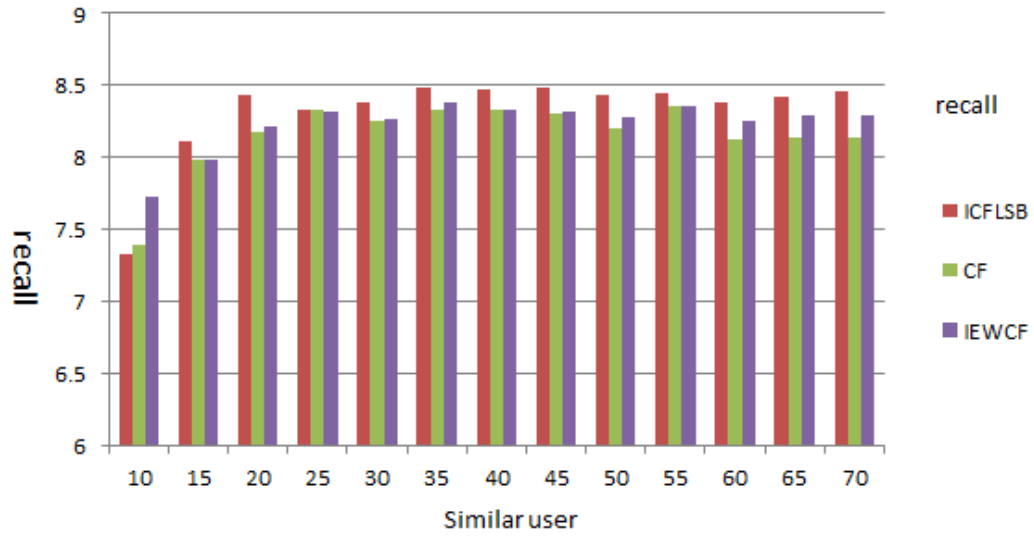

Figure 4. Comparison of recall rate of 70 users

5. Conclusions

The ICFLSB proposed in this paper avoids the sparse problem in the original rating matrix and subjective will problem in the process of extracting features, resulting in a better recommendation performance with a higher recommendation accuracy and recall rate. In this paper, we first utilize LFM model to generate the required features, avoiding the subjective will problem and maintaining the recommendation results' confidence. Then, we build a Bayesian model based on the generated features and we fill the original rating matrix with the results of the Bayesian model to solve the sparse problem in the original rating matrix, resulting in a higher efficiency of finding similar users. After we obtain all the information of similar users, we utilize the collaborative filtering algorithm to conduct the recommendations. Furthermore, we adopt Accuracy and Recall Rate as the evaluation metrics. In order to verify the validity and availability of ICFLSB, we compare the performance of ICFLSB with that of CF, DRCF, and IEWCF. The results show that ICFLSB can achieve a 2.152%

higher and 1.152% higher on recommendation accuracy and recall rate respectively when compared to CF, and it can achieve a 0.314% higher and 0.157% higher on recommendation accuracy and recall rate respectively when compared to IEWCF.

Acknowledgments

This work was supported by the Project of Jiangsu Provincial Policy Guidance Program (Enterprise-University-Research Institute Cooperation)-Perspective Joint Research Program under Grant No. BY2016065-52, Taicang Key Project of Research and Development Plan (Industry Perspective and Common Key Technologies) under Grant No. TC2016GY07, and the central special funds to guide the development of local science and technology project under Grant No. [2016] 4008.

References

1. F. Anishya and K. M. Suresh, "A Novel Approach for Travel Package Recommendation Using Bayesian Approach," International Conference on Computing and Communications Technologies. IEEE, pp.296-301, 2015
2. Q. Ba, X. Li and Z. Bai, "Clustering Collaborative Filtering Recommendation System Based on SVD Algorithm," *IEEE International Conference on Software Engineering and Service Science*. IEEE, pp. 963-967, 2013
3. D. Banas, C. Havrilova, and J. Paralic, "Combination of User Profile Information and Collaborative Filtering in Recommendations," *International Conference on Intelligent Engineering Systems.* IEEE, pp. 359-363, 2015
4. M. Ciesielczyk, A. Szwabe and M. Morzy, "Progressive Random Indexing: Dimensionality Reduction Preserving Local Network Dependencies," *Acm Transactions on Internet Technology*, vol. 17, no. 2, pp. 20, 2017
5. G. Guo, J. Zhang and N. Yorkesmith, "A Novel Evidence-Based Bayesian Similarity Measure for Recommender Systems," *Acm Transactions on the Web*, vol. 10, no. 2, pp. 1-30, 2016
6. L. Hui, H. Yun, and L. Cunhua, "Personalization Recommendation Algorithm Based on Nearest Neighbor Relation," *Computer Engineering and Applications*, vol. 48, no. 36, pp. 205-209, 2012
7. B. O. Ibrahim, N. Ithnin, and M. Nilashi, "A Multi-criteria Recommendation System Using Dimensionality Reduction and Neuro-Fuzzy Techniques," *Soft Computing*, vol. 19, no. 11, pp. 3173-3207, 2015
8. Y. Ji, W. Hong, and J. Qi, "Missing Value Prediction Using Co-clustering and RBF for Collaborative Filtering," *International Conference on Cloud Computing and Big Data.* IEEE, pp. 350-353,2015
9. C. Kaleli, "An Entropy-based Neighbor Selection Approach for Collaborative Filtering," *Knowledge-Based Systems*, vol. 56,pp. 273-280, 2014
10. X. Liu, "An Improved Clustering-based Collaborative Filtering Recommendation Algorithm," *Cluster Computing*, pp. 1-8, 2017
11. Z. Liu, "Collaborative Filtering Recommendation Algorithm Based on User Interests," *International Journal of u- and e-Service, Science and Technology*, vol. 8, 2015
12. J. Mao, Z. Cui, and P. Zhao, "An Improved Similarity Measure Method in Collaborative Filtering Recommendation Algorithm," *International Conference on Cloud Computing and Big Data,* pp. 297-303, 2013
13. S. Peng, Z. B. Zhou, and G. J. Wang, "Collaborative Filtering Algorithm Based on Rating Matrix Pre-filling," *Computer Engineering*, vol. 39, no. 1, pp. 175-178, 2013
14. D. Shin, S. Cetintas, and K. C. Lee, "Tumblr Blog Recommendation with Boosted Inductive Matrix Completion," *ACM International*, ACM, pp.203-212, 2015
15. H. Wang, N. Wang, and D. Y. Yeung, "Collaborative Deep Learning for Recommender Systems," *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* ACM, pp. 1235-1244, 2015
16. Q. Wang, X. Liu, and S. Zhang, "A Novel APP Recommendation Method Based on SVD and Social Influence," *Algorithms and Architectures for Parallel Processing*. 2015
17. X. Y. Wang and H. Liu, "Collaborative Filtering Recommendation Algorithm Integrated into Co-Rating Impact Factor," *Advanced Materials Research*, pp. 926-930, 2014
18. H. Wen, G. Ding, and C. Liu, "Matrix Factorization Meets Cosine Similarity: Addressing Sparsity Problem in Collaborative Filtering Recommender System, "*Web Technologies and Applications. Springer International Publishing*, pp. 306-317, 2014
19. M. Yan, W. Shang and Z. Li, "Application of SVD Technology in Video Recommendation System," *Ieee/acis, International Conference on Computer and Information Science*. IEEE, pp.1-5, 2016
20. Q. Yuan, G. Cong, and K. Zhao, "Who, Where, When, and What: A Nonparametric Bayesian Approach to Context-aware Recommendation and Search for Twitter Users," *Acm Transactions on Information Systems*, vol. 33, no. 1, 2015
21. W. Y. Zhang, S. S. Guo and S. Zhang, "Combining Hyperlink-induced Topic Search and Bayesian Approach for Personalised Manufacturing Service Recommendation," *Taylor and Francis Ltd*, pp. 1152-1163, 2016

Yun Wu received his Ph.D. degree from Guizhou University, Guizhou, China, in 2009. He is currently an associate professor, graduate supervisor, and the member of China Computer Society. His research interests include Distributed Computing, Game Theory, Recommender System, and Big Data and its Application.

Ren Qian is a master student in the College of Computer Science and Technology, Guizhou University. His current research interests include Distributed System and Recommender System.

Xiaofei Dong is a Ph.D. candidate of the School of Mechanical Engineering in Jiangsu University, China. He is a lecturer of School of Mechanical Engineering at Yancheng Institute of Technology, China. His current research interests include Sheet Metal Deforming, Manufacturing based on the Internet of Things, and Networked Manufacturing.

Yiqiao Li is a Masters student in the College of Computer Science and Technology, Guizhou University. Her current research interests include Recommender System, Distributed Computing and Data Mining.

Xinwei Niu received his Ph.D. from the Department of Electrical and Computer Engineering in Florida International University, USA. He is currently a visiting assistant professor of Electrical and Computer Engineering at Penn State Behrend. His current research interests include High Performance Computing, Hardware Acceleration, Reconfigurable computing, Hardware Security, and Power-/thermal-aware computing.