A collaborative filtering recommendation algorithm based on user clustering with preference types

A collaborative filtering recommendation algorithm based on user clustering with preference types

Yan Yang*, Huaxiong Yao, Rong Li and Sai Wang

Computer School, Central China Normal University, Wuhan, Hubei, 430079, China

*Corresponding author's e-mail: ms_yangyan@mail.ccnu.edu.cn

Abstract. The traditional collaborative filtering algorithm is extensively applied in the field of personalized recommendation, but it still faces the challenges of data sparsity and scalability problems which result in low quality and efficiency of recommendation. To address the problems, a hybrid collaborative filtering recommendation algorithm is proposed based on user preference type clustering. First, by analyzing the relationship between users and item categories, we construct the user item category preference matrix. On this basis, user clustering is carried out and users with similar preference types are clustered into the same user groups. Then, to search for the nearest neighbors of target user, similarity of users is calculated in the cluster it belongs to by considering both the user ratings and their preferred item categories. Finally, an improved Slope One algorithm is proposed and applied to the nearest neighbor set to predict item ratings and then generate recommendations. The experiments indicate the proposed hybrid collaborative filtering algorithm can improve recommendation performance in the case of a great many users.

1. Introduction

With the fast development of the Internet, the explosive growth of data on the network leads to more and more serious information overload phenomenon. In order to help users quickly locate the information they are most likely to be interested in from the massive network data, recommender systems came into being. Recommender systems are one of the most efficacious approaches to mitigating the impact of information overload and realizing personalized services[1].

Nowadays, the most commonly used recommendation algorithms include content-based recommendations[2], collaborative filtering recommendations[3] and various hybrid recommendations[4]. As a kind of traditional recommendation technology, collaborative filtering algorithm has gained more favor in application because of its simple and efficient advantages. However, the collaborative filtering algorithm still faces such challenges as data sparsity and scalability in its actual application[5]. Many scholars and practitioners have proposed a variety of improvement methods for the algorithm. *Wu y. et al.*[6] proved that the clustering-based collaborative filtering algorithm has better recommendation effect in sparse data. *Chalco C. I. et al.*[7] introduced a collaborative filtering algorithm based on agglomerative hierarchical clustering for recommender systems. *Koohi h. et al.*[8] first applied fuzzy C-means to user-based collaborative filtering recommendation. *Li Taoying et al.*[9] proposed to improve the Slope One personalized recommendation algorithm by using the user interest forgetting function. *He Ming et al.*[10] proposed to combine item clustering and user preferences to collaborative filtering recommendation, which can raise recommendation accuracy to a certain extent.

Although the above works help reduce the adverse effect of data sparsity, there are also other problems, which are mainly manifested in: relying too much on the user-item rating matrix, some internal information among users is not fully utilized; searching nearest neighbors takes too much time,

many algorithms are too complex to meet the real-time requirements of recommendation. Therefore, a hybrid collaborative filtering recommendation algorithm is proposed based on user preference type clustering, trying to alleviate sparsity of rating matrix and maintain a certain degree of scalability.

2. Conventional collaborative filtering recommendation algorithm

2.1. Overview of collaborative filtering recommendation

Conventional collaborative filtering recommending methods mainly contains user-based and item-based collaborative filterings. Their key idea is to compute similarities of users or items according to user-item rating matrix, so as to make recommendations. Collaborative filtering algorithms generally include the following 4 steps.

1) *Building user-item rating matrix:* After collecting users' ratings or browsing information, and cleaning, transforming and entering the data, the user-item rating matrix can be obtained, which provides data basis for collaborative filtering algorithms.

2) *Searching for the nearest neighbor set of users (or items):* Similarities between target user (or item) and the others are calculated according to the user-item rating matrix, and the top k users (or items) are chosen to form the nearest neighborhood.

3) *Predicting item ratings:* The current user's rating of an item needs to be approximated according to the item rating of its nearest neighbor set. The product of the rating and its corresponding user (or item) similarity is calculated respectively for each neighbor involved, and their average is taken as the predicted rating of the item.

4) *Generating recommendation list:* After making predictions for every unrated item of target user, the top N items are chozen to form a recommendation list, which is fed back to users as the final recommendation result.

2.2. Main challenges for collaborative filtering recommendation

Due to the continuous expansion of system scale and user number, collaborative filtering algorithm has gradually exposed some challenging problems.

1) *Data sparsity problem:* Data sparsity is one primary reason for quality degradation of recommendation algorithms. A large quantity of rating data are necessary for the proper working of collaborative filtering algorithm. In practical application, although the overall amount of rating data seems large, but for a single specific user, it will be rare. Lack of rating data makes user-item rating matrix become a high-dimensional sparse matrix. If the items mutually rated by users are very few, the recommendation list will have strong contingency, which weakens the reliability of recommendation results.

2) *Scalability issues:* As system scale expands unceasingly, the number of users as well as items increases dramatically, which makes the dimension of user-item rating matrix be upgraded to thousands or even millions. The computation for user (or item) similarities needs more time, which cannot meet the real-time requirement of recommender systems.

This paper attempts to alleviate the above two problems in collaborative filtering recommendations.

3. A hybrid collaborative filtering recommendation algorithm based on user preference type clustering

3.1. Recommendation process

Our recommendation process can be summarized as following.

1) First, a user item category preference matrix is constructed through the combination of user-item ratings and item-category data sets.

2) K-means clustering algorithm is leveraged for clustering system users. Users with similar item category preferences are divided into the same cluster, and K user clusters are obtained.

3) The similarities between target user and the others within the cluster is calculated. M users with top highest similarities are sought out to form the nearest neighborhood of target user.

4) An improved Slope One algorithm introducing user similarities is applied to the target user's nearest neighborhood for predicting the ratings of his unrated items. Finally, the top-N recommendation list is derived on the basis of the ranking results of the predicted ratings.

The first two steps above can be completed offline; the second two steps need to be processed online.

3.2. Recommendation algorithm

3.2.1. Constructing user item category preference matrix.

An item in the recommender system can belong to multiple categories. For example, a movie may be a romantic movie or a comedy movie. Assume the set of users in the system is $U = \{u_1, u_2, ..., u_m\}$, the set of items is $I = \{i_1, i_2, ..., i_n\}$, and the item category set is $C = \{c_1, c_2, ..., c_t\}$. The item category matrix IC can be constructed from the relationship between the items and the categories. Rows of matrix IC represent the items, columns represent the categories, and IC_{ij} denotes whether the i^{th} item belongs to category j. When item i has category attribute j, IC_{ij} score is 1. When item i does not have category attribute j, $ICij$ score is 0.

By combining the user rated items with the category attributes of different items, the users' evaluation matrix on different item categories can be obtained. The calculation formula of user u's preference degree $p_{u,c}$ for item category c is as follows:

$$p_{u,c} = \frac{\sum_{i \in I_{u,c}} r_{u,i}}{card(I_{u,c})} \tag{1}$$

where $I_{u,c}$ denotes the set of items with the category attribute c among the items rated by user u; $r_{u,i}$ represents the real rating of user u on item i; *card()* represents the number of elements in a set.

After obtaining the user rating data and item category information, users' preference for item types can be extracted to build a user item category preference matrix P as is shown in Table 1. In matrix P, rows represent the users, columns represent the categories, and p_{ij} denotes user i's preference for category j, which is calculated by equation (1).

Table 1. User item category preference matrix.

	c_1	c_2	...	c_j	...	c_t
u_1	p_{11}	p_{12}	...	p_{1j}	...	p_{1t}
u_2	p_{21}	p_{22}	...	p_{2j}	...	p_{2t}
...	...	...	...	...	...	...
u_i	p_{i1}	p_{i2}	...	p_{ij}	...	p_{it}
...	...	...	...	...	...	...
u_m	p_{m1}	p_{m2}	...	p_{mj}	...	p_{mt}

Usually, users only evaluate the items in the category they are interested in, so it can be considered that there are some similarities between those users who are interested in the same category. Therefore, user similarities can be measured by the resemblance degree of users' preferences for different item categories. In this work, cosine similarity is adopted to compute the preference type similarities between users.

$$sim_p(u,v) = \frac{\sum_{i=1}^{t} p_{u,i} p_{v,i}}{\sqrt{\sum_{i=1}^{t} p_{u,i}^2} \sqrt{\sum_{i=1}^{t} p_{v,i}^2}} \tag{2}$$

In equation (2), $p_{u,i}$ and $p_{v,i}$ denote user u's and user v's preference for category i; t represents number of categories.

3.2.2. User clustering based on item category preferences.

K-means clustering algorithm is leveraged for clustering system users. Users with similar category preferences are divided into the same cluster. The similarity calculation uses equation (2) in the clustering algorithm.

After the preference type based user clustering, the number of users in each cluster is far less than the original user number. And we find a user space with similar category preferences for the target user. When the target user arrives, search his nearest neighbors only within this user space. Thus, we can reduce the interference caused by the users with too few common ratings and improve the performance of nearest neighbor searching.

3.2.3. Searching nearest neighbors.

Once the cluster to which the target user belongs is determined, the similarities can be calculated between the target user and the others in this user cluster respectively. M users that have top greatest similarities with target user are chosen to form the nearest neighbor set. As the user mutual rating based similarity and the category preference based similarity evaluate the similarity of users from different perspectives, we propose to combine these two similarity measurement methods through certain weight factors to give a comprehensive similarity degree.

The formula for computing comprehensive similarity of users is as follows:

$$sim(u,v) = \alpha sim_p(u,v) + (1-\alpha) sim_r(u,v) \tag{3}$$

where $sim(u,v)$ represents the comprehensive similarity of user u and v; $sim_p(u, v)$ denotes the similarity of user u and v based on their category preferences, which is computed by equation (2); $sim_r(u, v)$ denotes the cosine similarity of user u and v based on their mutual ratings, which is computed by equation (4); α is the weight factor. α and $1-\alpha$ represent respectively the weight of the preference type based similarity and the mutual rating based similarity in comprehensive similarity calculation. By choosing the appropriate value of α, we can combine the advantages of these two kinds of similarity measurements, which helps improve the effect of the recommendation algorithm.

$$sim_r(u,v) = \frac{\sum_{i \in I_{u,v}} r_{u,i} r_{v,i}}{\sqrt{\sum_{i \in I_{u,v}} r_{u,i}^2} \sqrt{\sum_{i \in I_{u,v}} r_{v,i}^2}} \tag{4}$$

In equation (4), Iu,v denotes the set of items user u and v rated, and ru,i denotes user u's rating on item i.

After obtaining the similarity matrix between users with the above calculation method, the top M users who have the largest comprehensive similarities with target user can be found out to be the nearest neighbor set.

3.2.4. Predicting ratings using improved Slope One algorithm and generating recommendation list.

After the nearest neighbor set is determined, we propose to apply an improved weighted Slope One algorithm which takes user similarities into consideration to this set for evaluating and predicting the user's unrated items. Thereafter, the top-N recommended item list is generated according to the ranking result of the predicted ratings.

The core idea of Slope One is to infer the rating difference between two items from the ratings of the group of users on each item, so as to approximately estimate the rating of target user towards target item according to the rating difference as well as his historical rating. Slope One exploits the computation method of linear regression analysis. And it has the merits of simple implementation and high efficiency. Conventional formula for computing user *u*'s predicted rating $P(u)_j$ for item *j* is as following:

$$dev_{j,i} = \frac{\sum_{t \in U_{j,i}} (r_{t,j} - r_{t,i})}{card(U_{j,i})} \tag{5}$$

$$P(u)_j = \frac{\sum_{i \in I_j} (dev_{j,i} + r_{u,i})}{card(I_j)} \tag{6}$$

Here, $dev_{j,i}$ refers to the rating difference of item *j* from item *i*; $U_{j,i}$ refers to the user set that rated both item *i* and *j*; $r_{t,j}$ and $r_{t,i}$ refer to the rating for item *j* and item *i* by user *t* respectively; I_j denotes the set of all the items that user *u* rated and meets the condition $(i \neq j \text{ and } U(j,i) \neq \Phi)$ as well.

Conventional Slope One algorithm ignores the weight of rated items with different number of users, and does not consider the impact of similarity differences between users on the evaluation results, which will have negative influence on the accuracy of the algorithm to a certain degree. Therefore, we propose an improved weighted Slope One algorithm which takes user similarities into consideration. It not only uses the number of the users jointly rated among items as weight when calculating the prediction rating (the weight of items with more users jointly rated is relatively large), but also adds the similarity degree between the neighbor and target user as the weight to the computation of the average rating deviation. Thus, the algorithm further fixes the prediction result of Slope One. The formula for that is as follows.

$$dev_{j,i} = \frac{\sum_{t \in U_{j,i}} (r_{t,j} - r_{t,i}) \cdot sim(t,u)}{\sum_{t \in U_{j,i}} sim(t,u)} \tag{7}$$

$$P(u)_j = \frac{\sum_{i \in I_j} (dev_{j,i} + r_{u,i}) \cdot card(U_{j,i})}{\sum_{i \in I_j} card(U_{j,i})} \tag{8}$$

Here $sim(t,u)$ represents the similarity degree of two users *t* and *u*, and its calculation method is shown in equation (3).

4. Experiments and analysis

4.1. Data set

The experiment data in this work are collected from the MovieLens data set, which is provided by the GroupLens research group of Minnesota University. MovieLens data set records users' rating data of movies, which is a common benchmark data set for personalized recommendation algorithm evaluation. The 1M data subset adopted in our work contains 1,000,209 rating records of about 3,900 movies from 6,040 users. Every user rated no less than 20 movies, with the scores ranging from 1 to 5. The higher the rating value, the greater the user's favor for the movie. All movies are tagged as one or more of 18 different categories, such as action, comedy, hero, romance, etc. In our experiments, 80% of the data set is randomly selected as the training set, and the other 20% is as the testing set.

4.2. Metrics

RMSE (root mean square error) and MAE (mean absolute error) are two most frequently used metrics for measuring the accuracy of rating predictions. In our work, MAE is used to measure the precision of

recommendation algorithms. The lower the MAE value, the greater the precision of the algorithm. The formula for computing MAE is as following:

$$MAE = \frac{\sum_{i=1}^{n} |p_i - q_i|}{n} \tag{9}$$

where $\{p_1, p_2, \ldots, p_n\}$ denotes the rating set of the user predicted by the recommendation algorithm, $\{q_1, q_2, \ldots, q_n\}$ represents the real ratings for the items, and n denotes the number of the items rated by user.

4.3. Experimental results

For the purpose of evaluating the accuracy of this recommendation algorithm, we compare it with traditional user based collaborative filtering algorithm (UBCF) and item based collaborative filtering algorithm (IBCF). In our experiments, user cluster number K is set to 6, because in the preprocessing of the data set it is found that when $K = 6$ the data of users in each cluster are evenly distributed. Weight factor α is set to 0.4 in user similarity calculation, because through observation, the value of MAE is the lowest when $\alpha = 0.4$. And the number of neighbors M in nearest neighbor searching is set to be 10-80, increasing by 10 each time. The influence of the change of M on the recommendation accuracy of each algorithm is observed in turn. The experimental results are presented in Figure 1.

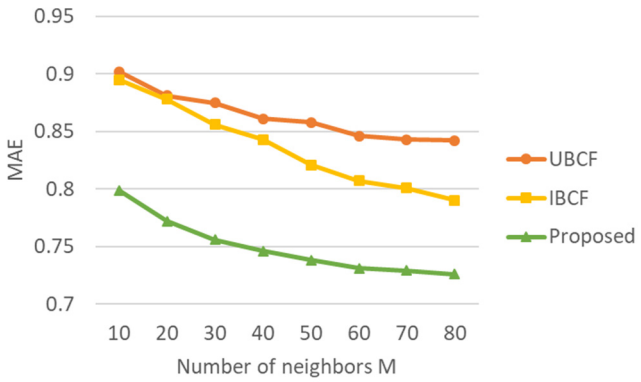

Figure 1. MAE values of different recommendation algorithms.

As indicated in Figure 1, with the increase of the number of nearest neighbor users M, the MAEs of the three algorithms decrease in a certain degree, which means that the recommendation accuracy is gradually improving. And the MAE value of the proposed algorithm is lower than those of UBCF and IBCF. When the number of neighbors is close to 60, the MAE value is gradually getting stable and tends to be optimal. Therefore, if the number of nearest neighbors as well as other parameters are set properly, our proposed algorithm has better performance than the other two algorithms in recommendation accuracy.

5. Conclusions

Targeting at the data sparsity and scalability problems of conventional collaborative filtering algorithm, this paper proposes a hybrid collaborative filtering recommendation algorithm which is based on user preference type clustering. First, by analyzing the relationship between users and item categories, we construct the user item category preference matrix. On this basis, user clustering is carried out and users with similar preference types are clustered into the same user groups. Then, to search for the nearest neighbors of target user, the comprehensive similarity of users is calculated in the cluster it belongs to by considering both the user ratings and their preference types. Finally, an improved Slope One algorithm is proposed and applied to the nearest neighbors to predict item ratings and then generate recommendations. The experiments indicate the proposed hybrid collaborative filtering algorithm can improve recommendation performance in the case of a great many users.

Acknowledgments
The work of this paper was supported by the National Natural Science Foundation of China under the grant 61672257.

References

[1] Chen R., Hua Q., Chang Y., Wang B., Zhang L., Kong X. (2018) A Survey of Collaborative Filtering-Based Recommender Systems: From Traditional Methods to Hybrid Methods Based on Social Networks. IEEE Access, 6: 64301-64320.
[2] Shu J., Shen X., Liu H., Yi B., Zhang Z. (2018) A Content-Based Recommendation Algorithm for Learning Resources. Multimedia Systems, 24: 163–173.
[3] Xue F., He X., Wang X., Xu J., Liu K., Hong R. (2018) Deep Item-based Collaborative Filtering for Top-N Recommendation. ACM Transactions on Information Systems, 37(3): 1-25.
[4] Logesh R., Subramaniyaswamy V. (2019) Exploring Hybrid Recommender Systems for Personalized Travel Applications. Cognitive Informatics and Soft Computing, Advances in Intelligent Systems and Computing, 768: 535-544.
[5] Ming X. (2017) Research on Collaborative Filtering Recommendation Algorithm Based on User Clustering. Beijing Jiaotong University, Beijing.
[6] Wu Y., Liu X., Xie M., Ester M., Yang Q. (2016) CCCF: Improving Collaborative Filtering via Scalable User-Item Co-clustering. In: Proceedings of the Ninth ACM International Conference on Web Search and Data Mining. San Francisco. pp. 73-82.
[7] Chalco C.I., Chasi R.B., Ortiz R.H. (2018) Hierarchical Clustering for Collaborative Filtering Recommender Systems. In: Proceedings of International Conference on Applied Human Factors and Ergonomics. Orlando. pp. 346-356.
[8] Koohi H., Kiani K. (2016) User Based Collaborative Filtering Using Fuzzy C-means. Measurement, 91: 134-139.
[9] Li T., Li M., Li P. (2017) Personalized Collaborative Filtering Recommendation Algorithm Based on Weighted Slope One. Application Research of Computers, 34(8): 2264-2268.
[10] He M., Sun W., Xiao R., Liu W. (2017) Collaborative Filtering Recommendation Algorithm Combining Clustering and User Preferences. Computer Science, 44(11A): 391-396.