A Hybrid Collaborative Filtering Recommendation Algorithm Based on User Attributes and Matrix Completion

Jie Yi, Maosheng Zhong*, Yinfen Chen and Anquan Jie
School of Computer Information Engineering, Jiangxi Normal University, China

*Corresponding author

Abstract. Collaborative filtering is a popular strategy in recommendation system. Traditional collaborative filtering relies on the user-item rating matrix that encodes the individual ratings of users for items to make recommendations. However, in the real-world, the rating matrix is highly sparse, and many new users do not have rating records, thus traditional collaborative filtering could not provide satisfactory recommendations. To alleviate this issue, we propose a hybrid algorithm that utilizes LMaFit to complete rating matrix, reducing the degree of sparsity, and provides a hybrid user-similarity to supply a good support for recommending to new users in the condition of cold start. Extensive experiment results on real-world datasets show the proposed algorithm has a better performance than other methods.

Keywords: Collaborative filtering; Data sparsity; Recommender system.

1. Introduction

Nowadays, collaborative filtering[1,2] is the most successful and widely used personalized recommendation technology in e-commerce recommendation system. The basic idea of collaborative filtering is that the target user will be attracted with the items that the neighbors with the same hobbies. That is, similarity measurement algorithm is used to search for the neighbors who has the same hobby and the recommendation system selects a series of candidate recommendation items to the target users according to the information of similar users, and then predicts the ratings of the list of candidate recommendation items and produces recommendation results.

However, with the continuous increase of users and items on the e-commerce platform, data sparsity has become a bottleneck restricting the application of traditional collaborative filtering algorithms. This leads to low recommendation accuracy and poor real-time performance and so on. Since users rated items only account for a small percentage of all items, it is hard to judge whether the users have similar preferences where there is no overlapping between the items users have rated, thus decreasing the accuracy of recommendation. At the same time, surging users and items make the user-item matrix excessive, which makes the nearest neighbor range of the search target users larger, and the real-time recommendation performance is seriously affected. In essence, the main reason for these two problems is data sparsity. Cold start[3] is also a constant concern for recommender system. When a new items comes along, it is hard to give recommendations because of the lack of user ratings on item. In addition, when a new user joins in system, the system cannot recommend anything because of a lack of rating record. Therefore, it is necessary to deeply research on these two issues.

Based on the above phenomena, this paper proposes a hybrid collaborative filtering personalized recommendation algorithm. Details are as follows:

1. To alleviate the data sparsity problem, we utilize LMaFit algorithm to decrease the sparsity of the user-item rating matrix.
2. We propose a hybrid user-similarity based on users’ attributes and users’ ratings, which effectively solves the cold start problem and improves the accuracy of recommendation.
3. We perform substantial experiments using real datasets to evaluate our proposed algorithm and show it has a good performance in both precision and cold start.

2. Related Work

In order to solve the problem of sparsity and improve the accuracy of the recommender, many researchers have proposed various methods. Currently, several common methods are as follows:
(1) Clustering technology: clustering-based approaches provide an alternative to model-based methods, reducing the search space by clustering similar users or items together. [4] cluster the users by fuzzy C-means, and [5] cluster the items by k-means. [6] proposed Category Preferred Canopy-K-means which clusters users to build user-item Category Preferred Ratio matrix. [7] propose the subspace clustering approach, tree structures of neighbor users are drawn for the target user based on subspaces. Although clustering can narrow down the search for the nearest neighbor of the target user by reducing the dimension, different choices of initial clustering center could produce different results.
(2) Matrix factorization: by using matrix factorization, the potential factors of users and items from the original user-item matrix can be mined to estimate missing rating values. SVD[8], RSVD[9], PMF[10], APG[11], GSMF[12] are proposed as matrix factorization model to process the user-item matrix, and obtain latent factors of users and items. These models effectively alleviate the problem of data sparsity and improve the accuracy of recommendation. But matrix factorization applies to solving large-scale matrix problems, and looks pale in handling new users who don’t have rating record, and could not make recommendation.
(3) Matrix Completion: filling the unknown ratings is probably the most straightforward way to densify the sparse data. [13] utilized Slope One predictor to alleviate the sparsity problem and improve its scalability and accuracy. [14] filled the rating matrix by using a median average of rows and columns, and its fill-in method was divided into four cases to improve the effect of similarity calculation. Since the number of item class is much smaller the number of items, [15] proposed user-class score matrix, which increased the matrix data density and improve the cold start problem. For cold start issue, some researchers solve it by utilizing users’ attributes. Calculating the similarity through friend relationships and user tags is proposed to apply in social application[16]. The ISC algorithm based on user characteristics and item attributes preference was proposed to compute similarity[17]. The ACF algorithm based on user attributes was proposed to predict and adjust the rating matrix[18]. To solve the complete cold start and incomplete cold start problems for new items, [19] proposed two recommendation models, which are based on a framework of tightly coupled collaborative filtering approach and deep learning neural network. To solve the new-user cold start problem, a distributed hybrid recommendation framework based on user classification[20] is proposed. Those methods not only good for cold start, but also can alleviate data sparsity.

3. A Hybrid Collaborative Filtering Recommendation Algorithm Based on User Attributes and Matrix Completion

This section introduces traditional collaborative filtering algorithm and the proposed algorithm, and illustrates the procedure of the proposed algorithm.

3.1. Traditional Collaborative Filtering Algorithm

The goal of the recommendation system is to predict the current user’ ratings for items that have not been rated, and then recommend the recommendation list to the current user. The ratings of items can be obtained by some implicit or explicit methods. For example, the ratings can be expressed in different ways by analyzing the users’ behaviors in the user set (such as like and dislike using integers 0 to 5, or binary values 0 and 1). In the collaborative filtering algorithm, the rating matrix and the similarity function are first used to calculate similarity between users, then form a similar user set and predict the score of the unrated items, and finally recommend the list with N items to the current user.
3.1.1. The calculation of user-similarity. The user-item ratings matrix $R = [r_{ij}]_{n \times m}$ contains the rating records of $n$ users for $m$ items, where $1 \leq i \leq n$, $1 \leq j \leq m$.

The user-based collaborative filtering recommendation algorithm always measures the similarity between users according to the user’s historical rating of the items. There are several methods to calculate users’ similarity: Pearson correlation coefficient, cosine vector and modified cosine vector. The Pearson correlation coefficient method is used to generate the similarity $\text{sim}_u(u, v)$ between user $u$ and user $v$ in this paper, and can be expressed as below:

$$\text{sim}_u(u, v) = \frac{\sum_{i \in I_u} (r_{ui} - \bar{r}_u)(r_{vi} - \bar{r}_v)}{\sqrt{\sum_{i \in I_u} (r_{ui} - \bar{r}_u)^2 \sum_{i \in I_v} (r_{vi} - \bar{r}_v)^2}}$$  \hspace{1cm} (1)

3.1.2. Finding the $K$ nearest neighbor set. Assuming that the user’s interest is relatively stable in the recommendation system, and the more similar the interest between two users, the more likely these two users like the same items. The users in the user set $U$ are sorted in descending order according to the similarity with the target user. The larger the value of $\text{sim}_u(u, v)$ is, the more similarities between the users, the more likely the users are interested in the same items. Then, find the top $K$ users with the highest $\text{sim}_u(u, v)$ value, that is, the $K$ neighbors most similar or closest to the target user. The set of $K$ nearest neighbors of target user $u$ can be represented as $N_u$. The value of $K$ is adjustable, and different value can affect the accuracy of the recommendation.

3.1.3. Ratings Prediction. According to $\text{sim}_u(u, v)$ and $N_u$, the ratings of target user $u$ on item $i$ can be obtained. The formula is as follows:

$$\hat{r}_{ui} = \bar{r}_u + \frac{\sum_{n \in N_u} \text{sim}_u(u, n) \times (r_{ni} - \bar{r}_n)}{\sum_{n \in N_u} |\text{sim}_u(u, n)|}$$  \hspace{1cm} (2)

Where $\bar{r}_n$ is the average rating of items target user $u$ has rated, $\bar{R}_u$ is the average rating of items neighbor $n$ has rated; $r_{ni}$ is the rating from neighbor $n$ on item $i$.

3.2. Traditional Collaborative Filtering Algorithm

3.2.1. Similarity computing based on users’ attributes. In the real application, the user's ratings of the items lack of self-consciousness, and it has great limitations to merely rely on ratings. For the cold start issue, there is a solution to recommend users based on the similarity of users’ attributes when the number of users’ ratings is few, and gradually transit to utilize users’ ratings for recommendation with the increase of ratings. In this paper, a formula is used to merge users’ attributes recommendation and users’ ratings recommendation, achieving a smooth transition between two recommendations. The user’s attributes are generally divided into: numeric (ratio) attributes, such as the user’s age; binary attributes, such as the user’s gender; nominal attributes, such as the user’s occupation. The impacts of these factors on the recommendation in the calculation of similarity are considered according to the use of the datasets.

Since the user's age value distribution is a continuous integer, the age set is expressed as $\text{age} = \{\text{age}_1, \text{age}_2, ..., \text{age}_8\}$, where $\text{age}_i$ represents the corresponding age. The occupations set is represented as: $\text{occu} = \{\text{occu}_1, \text{occu}_2, ..., \text{occu}_n\}$, where $\text{occu}_n$ represents a particular occupation. The similarity between user $u$ and $v$ can be calculated by the following formula.

(1) Sex attributes

Different gender users have different needs for the commodity, and the similarity formula based on gender is:
sex(u, v) = \begin{cases} 
1, & u.sex = v.sex \\
0, & u.sex \neq v.sex 
\end{cases}

(2) Age attributes

The similarity formula based on age is shown as:

\[ \text{age}(u, v) = \begin{cases} 
1, & |u.age - v.age| \leq 10 \\
0, & |u.age - v.age| > 10 
\end{cases} \]

(3) Job attributes

The similarity formula based on job is listed as:

\[ \text{job}(u, v) = \begin{cases} 
1, & u.job = v.job \\
0, & u.job \neq v.job 
\end{cases} \]

So the formula of the similarity based on user’s attributes are showed as Eq.3.

\[ \text{sim}(u, v) = \frac{1}{2} \left( \exp(\text{Attr}(u, v)) - 1 \right) \]
\[ \text{Attr}(u, v) = a \cdot \text{sex}(u, v) + b \cdot \text{age}(u, v) + c \cdot \text{job}(u, v) \]  

(3)

where a, b, c is the percent of sex attributes, age attributes and job attributes, and a + b + c = 1.

3.2.2. A hybrid user-similarity generated.

After Eq.1 and Eq.3 calculating similarity respectively, the final similarity between two users is calculated by the weighted method, and can be expressed by Eq.4.

\[ \text{sim}(u, v) = \alpha \text{sim}_{sex}(u, v) + (1 - \alpha) \text{sim}_{age}(u, v) \]  

(4)

3.2.3. Low-rank Matrix Completion.

LMaFit (Low-rank Matrix Fit) algorithm can be used to complete user-item matrix, which is simple and effective. In addition, LMaFit need not to calculate the nuclear norm, avoiding massive calculation from SVD. LMaFit algorithm can be represented as:

\[ \min_{X,Y,Z} \frac{1}{2} \| X Y - Z \|_F^2, s.t. P_{\Omega} = P_{\Omega} (M) \]  

(5)

where \( X \in R^{m \times K}; Y \in R^{K \times n}; Z \in R^{m \times n}; K \) represents the rank of matrix, which is an adjustable parameter. By using LMaFit algorithm to process the original user-item matrix, most of the zero elements in the matrix are filled, which effectively alleviates the data sparsity problem.

3.3. Traditional Collaborative Filtering Algorithm

We have introduced a hybrid similarity based on user attributes and matrix completion to the traditional collaborative filtering recommendation algorithm; Besides, we utilize the k-means++ clustering to find the better neighbors of target users, thus we propose a hybrid collaborative filtering recommendation algorithm based on user attributes and matrix completion.

The algorithm process is as follows:

Algorithm1: proposed algorithm

Input: user-item matrix \( R \), users’ attributes tables

Output: user-item matrix \( \hat{R} \).

Begin
Step1: Calculate the \( \text{sim}(u, v) \) by users’ attributes tables and the Eq. 4;
Step2: Cluster the users by k-means++, obtaining the users’ class;
Step3: Complete the matrix \( R \) by LMaFit, obtaining completed matrix \( \hat{R} \);
Step4: Find the class to which target user belongs;
Step5: Select the top K users with high \( \text{sim}(u, v) \) values as the neighbors of target user;
Step6: Predict the ratings of target user on items by \( R \) and the Eq. 2, obtaining matrix \( \hat{R} \).

End
4. Experiments
In this section, we describe our experiments and results in detail. Firstly, the datasets and evaluation metrics are introduced, and then the experiment setting is introduced. Finally, the experiment results and analysis are presented. And in all experiments, we set the parameter of the similarity calculation formula as: $a=0.6$, $b=0.3$, $c=0.1$, $\alpha=0.7$.

4.1. Datasets
The dataset we have used in our experiments is MovieLens-100k which is from the Movielens site (https://movielens.org/). The dataset contains 100k ratings from 943 users on 1682 movies. Every user has rated at least 20 movies on a rating scale of 1-5, with the higher the rating, the more interested the user was in the film. In addition, if the user does not rate a certain movie, the rating is 0. Here is the formula that can calculate the sparsity of data:

$$S = \left(1 - \frac{N}{m \times n}\right) \times 100\% \quad (6)$$

Where $S$ is sparsity of data, $m$ is the number of users, $n$ is the number of items, and $N$ is the number of ratings.

And we use the following two datasets on the data source of movielens-100k:
- Datasets 1 (u1-u3): we form three experimental datasets to test the performance of the algorithms under different sparsity. The sparsity degrees of three datasets can be calculated as 93.70%, 94.96% and 96.22%.
- Datasets 2 (ucold): These datasets are generated by sampling the original dataset to simulate the scene of the new users. There are 4 sub-datasets. The specific sampling methods are as follows: 50 users are randomly selected from 943 users as new users, and the new users randomly select 0, 2, 8, and 16 historical scores to join the training set, which are used to represent new users in turn. The training set also includes historical scores for the remaining 894 old users. The test set includes the remaining scores for the new user. The four sub-datasets are recorded as ut0, ut2, ut8, and ut16.

4.2. Evaluation Metric
We employ the mean absolute error (MAE) and root mean squared error (RMSE) as the evaluation metrics,

$$MAE = \frac{1}{N} \sum_{ij} |\hat{R}_{ij} - R_{ij}| \quad (7)$$

$$RMSE = \sqrt{\frac{1}{N} \sum_{ij} (R_{ij} - \hat{R}_{ij})^2} \quad (8)$$

where $R_{ij}$ is the rating of user $i$ on item $j$, $\hat{R}_{ij}$ denotes the corresponding predicted rating, $T$ is the set of ratings in the data and $|T|$ is the number of ratings.

4.3. Baselines
The proposed algorithm is compared against six popular collaborative filtering algorithms, which are described as follows:
- ACF ACF(Adaptive Collaborative Filtering) is based on user attributes to solve the cold start problem.
- APG APG(Accelerated Proximal Gradient) calculate the approximation by solving a nuclear norm regularized linear least squares problem.
- GSMF GSMF(Group-Sparse Matrix Factorization) factorizes the rating matrices for multiple behaviors into the user and item latent factor space with group sparsity regularization.
- ISC ISC(Improved Similarity Computing) is based on user characteristics and item attributes preference.
- PMF PMF(Probabilistic Matrix Factorization) is a model to factorize the user-item matrix to user and item factors, which exists Gaussian observation noise and Gaussian priors on the latent factor vectors.
RSVD (Regularized SVD) is a standard matrix factorization method for estimating user/project characteristic by minimizing the sum-squared error using L2 regularization.

4.4. Experiment results

4.4.1. The performance between Traditional Collaborative Filtering and proposed algorithm. In order to learn the impact of data sparsity on recommendation accuracy, we measure the performance between Traditional Collaborative Filtering and proposed algorithm with three sparsity degree. The sparsity of the three sub-datasets is 93.70%, 94.96% and 96.22%, respectively. After completing the matrix, the sparsity of the three sub-datasets is 3.23%, 3.88% and 6.09% respectively, which greatly reduces the sparsity of the rating matrix. It can be seen from Table 1, the difference of MAE value is 0.0743 when the sparsity degree of dataset is 93.70%; however, the difference of MAE value is 0.1908 which is 2.5 times more than the former when the sparsity degree is 96.22%. It is because that the sparser the data, the bigger the MAE value. This shows that completing matrix before predicting ratings can increase the accuracy of recommendation. In addition, the sparsity degree varies from 93.70% to 96.22%, the MAE value tends to be stability.

Table 1. MAE of Traditional Collaborative Filtering and proposed algorithm with different sparsity.

| Sparsity   | Traditional Collaborative Filtering | proposed algorithm |
|------------|-------------------------------------|-------------------|
| 93.70%     | 0.8554                              | 0.7811            |
| 94.96%     | 0.9347                              | 0.8096            |
| 96.22%     | 1.0244                              | 0.8436            |

4.4.2. Recommendation accuracy comparisons. Table 2 shows the recommendation results that compare APG, RSVD, PMF, GSMF and our hybrid algorithm using two evaluation metrics. We can see that APG has the worst accuracy. The proposed algorithm has the minimum RMSE value and MAE value, which fully demonstrates that the proposed algorithm can effectively improve the quality of recommendation.

Table 2. RMSE and MAE of compared models.

| Model  | RMSE  | MAE   |
|--------|-------|-------|
| APG    | 1.1065| 0.9312|
| RSVD   | 1.0382| 0.8066|
| PMF    | 1.0461| 0.8016|
| GSMF   | 1.0202| 0.7887|
| proposed | **0.9738** | **0.7811** |

4.4.3. The performance with different neighbor number K. The choice of neighbor number could affect the recommendation accuracy. We compare our algorithm with ISC and ACF, and obtain the MAE value of three methods with different neighbor number K. In Fig. 1, the MAE value of ACF is lower than other methods when the K=10, K=20 and K=30, but the performance of proposed algorithm is the best when K=40 and K=50, and the MAE value is 0.7811 when K =40.
4.4.4. The performance in the condition of cold start. In order to test the cold-start performance of algorithms, we select 2 datasets which have been deleted the relevant rating information to simulate the cold-start scenario, and test the cold-start ability. Fig. 2 shows the performance of three methods under some different circumstances. When new users have no ratings, all three methods can predict ratings according to users’ attributes. ACF and proposed method complete the user-item matrix before predicting, thus the MAE values of ACF and proposed method are better than the MAE value of ISC. In the case of few ratings, the performance of proposed method is significantly better than the other two methods. With the increase of users’ ratings, the performance of three methods become close. And Fig. 2 shows that the proposed method can smoothly transit to the warm start of the old users while it can be suitable for the cold start of new users.

5. Conclusion
This paper proposes a hybrid recommendation algorithm integrating matrix completion and users’ attributes, which not only can make user-item matrix dense, but also can be applicable when faced with cold start scenarios. Experiments demonstrate proposed algorithm does not have the best performance with RMSE and MAE as 0.9738, 0.7811 respectively, but it can achieve a smooth transition with taking account of new users and old users. LMaFit algorithm is not the best method to complete matrix due to a slight change in the original rating. In the future work, we will focus on better method to complete matrix, and proposed algorithm will be transplanted into the cloud computing environment to improve the parallelism and extensibility of the algorithm.

Acknowledgements
This work was supported by National Natural Science Foundation of China (Grant No.61877031, 61462027 and 61462045), and Science and Technology Project Founded by the Education Department of Jiangxi Province (Grant No.GJJ160277).
References

[1] Bobadilla J., Ortega F., Hernando A., Gutiérrez A.: Recommender systems survey. Knowledge-based systems, 46, 109–132 (2013).
[2] Jannach D., Zanker M., Felfernig A., Friedrich G.: Recommender Systems: An Introduction, Cambridge University Press, New York, USA (2010).
[3] Jesús Bobadilla, Ortega F., Hernando A., Jesús Bernal: A collaborative filtering approach to mitigate the new user cold start problem. Knowledge-Based Systems26, 225–238 (2012).
[4] Koohi H., Kiani K.: User Based Collaborative Filtering using Fuzzy C-Means. Measurement 91, 134–139 (2016).
[5] Himel M., Uddin M., Hossain M., Jang Y.: Weight based movie recommendation system using K-means algorithm. In: Proceedings of ICTC, pp. 1302–1306, IEEE (2017).
[6] Jianjiang L., Kai Z., Xiaolei Y., Peng W., Jie W., Karan M.: Category Preferred Canopy-K-means based Collaborative Filtering algorithm. Future Generation Computer Systems93, 1046–1054 (2019).
[7] Koohi H., Kiani K.: A new method to find neighbor users that improves the performance of collaborative filtering. Expert Systems with Applications. 83: 30–39 (2017).
[8] Vozalis M., Margaritis K.: Applying SVD on Item-based Filtering. In: Proceedings of ISDA, pp. 464–469 (2005).
[9] Paterek A.: Improving regularized singular value decomposition for collaborative filtering. In: Proceedings of KDD, pp. 5–8 (2007).
[10] Salakhutdinov R., Mnih A.: Probabilistic Matrix Factorization. In: Proceedings of NPSCA. pp. 1257–1264 (2007).
[11] Kimchuan Toh SY: An accelerated proximal gradient algorithm for nuclear norm regularized least squares problems. Pacific Journal of Optimization6(3), 615–640 (2010).
[12] Yuan T., Cheng J., Zhang X., Qiu S., Lu H.: Recommendation by Mining Multiple User Behaviors with Group Sparsity. In: Proceedings of AAAI, pp. 222–228 (2014).
[13] Liang T., Fan J., Zhao J., Liang Y. Li Y.: Improved slope one collaborative filtering predictor using fuzzy clustering. Advanced Data Mining and Applications8346, 181–192 (2013).
[14] Wang X., Liu Y., Xiong F.: Improved personalized recommendation based on a similarity network. Physica A: Statistical Mechanics and its Applications456, 271–280 (2016).
[15] Zeng C., Xing C., Zhou L.: Similarity measure and instance selection for collaborative filtering. In: Proceedings of WWW, pp. 652–658, ACM (2003).
[16] Ma T., Zhou J., Tang M., Tian Y., Al-Dhelaan A., Al-Rodhaan M., Lee S.: Social Network and Tag Sources Based Augmenting Collaborative Recommender System. IEICE Transactions on Information & Systems98(4), 902–910 (2015).
[17] Liu A., Li B.: Collaborative Filtering Recommendation Algorithm Based on Improved Similarity Computing. In: Proceedings of MMCCE, pp. 1375–1379 (2015).
[18] Yang S., Wang L., Cheng L.: Adaptive Collaborative Filtering Recommendation Algorithm Based on User Attributes. In: Proceedings of AITA, pp. 73–79 (2017).
[19] Wei J., He J., Chen K., Zhou Y., Tang Z.: Collaborative filtering and deep learning based recommendation system for cold start items. Expert Systems with Applications69, 29–39 (2017).
[20] Wang J., Chen Y.: A Distributed Hybrid Recommendation Framework to Address the New-User Cold-Start Problem. Ubiquitous Intelligence & Computing & IEEE Intl Conf on Autonomic & Trusted Computing & IEEE Intl Conf on Scalable Computing & Communications & Its Associated Workshops, 1686–1691 (2016).