Available online at www.ijpe-online.com

International Journal of Performability Engineering

vol. 15, no. 8, August 2019, pp. 2267-2276
DOI: 10.23940/ijpe.19.08.p27.22672276

Collaborative Filtering Algorithm based on Data Mixing and Filtering

Xiaohui Cheng[a,b], Li Feng[a,b], and Qiong Gui [a,b,*]

[a]*College of Information Science and Engineering, Guilin University of Technology, Guilin, 541000, China*
[b]*Guangxi Key Laboratory of Embedded Technology and Intelligent System, Guilin, 541000, China*

Abstract

Personalized recommendation systems based on the collaborative filtering algorithm are faced with an excessive user rating data sparseness problem. In order to solve this problem, an improved collaborative filtering algorithm is proposed, which gathers a variety of single numerical filling methods and selects a more appropriate filling method according to the filling rules to fill the vacant positions in the user-item scoring matrix filling. The recommendations are then made on the populated user-item score matrix through a user-based collaborative filtering approach. The method of data mixed filling can effectively reduce the recommended error and numerical singularity caused by fixed filling values such as the mean and median. The improved collaborative filtering algorithm is tested on the Movie Lens data set. The results show that the method of data mixing is adopted to fill the empty positions in the scoring matrix, which effectively alleviates the data sparsity problem in the collaborative filtering algorithm and improves the accuracy of recommendation systems for target users.

Keywords: collaborative filtering; sparse data; multivariate value filling; fill rule

(Submitted on June 11, 2019; Revised on July 13, 2019; Accepted on August 14, 2019)

1. Introduction

In order to solve the problem of how to quickly and accurately recommend products of interest to users and solve information overload problems, personalized recommendation systems have emerged and are widely used in various industries as a convenient way to find interesting and valuable information bands for users [1-2].

At present, researchers have introduced a variety of recommendation algorithms based on collaborative filtering. Compared with other algorithms, the collaborative filtering recommendation algorithm is scalable and easy to implement, and it is widely used in e-commerce and other fields. However, collaborative filtering technology still faces some challenges. For example, on some large websites and e-commerce platforms, the number of projects that users have evaluated is much less than the total number of projects on the website. As a result, the data in the user project scoring matrix is extremely sparse [3]. When calculating the similarity matrix, the accuracy of similarity is relatively low, and it will be difficult to find the nearest neighbor set of the target user or project with an inaccurate similarity value, which leads to the degradation of the recommendation system quality [4-5]. For this problem, the common method used by researchers to improve the collaborative filtering algorithm and effectively alleviate the sparse problem of the scoring matrix is empty value filling, which uses a single numerical value to increase the data density of the scoring matrix [6].

2. Related Concept

Collaborative filtering algorithms have become a mainstream technology in information filtering and systems. Their main functions are prediction and recommendation. They discover users' preferences by mining data about projects' historical behaviors, such as clicking and browsing evaluation records, classify users based on different interests, and recommend similar products [7-8]. Collaborative filtering can identify potential target user interests [9]. Collaborative filtering recommendation algorithms are divided into two categories: the user's collaborative filtering algorithms and the item-based

* Corresponding author.
E-mail address: cxiaohui@glut.edu.cn

collaborative filtering algorithms. The implementation process of user-based and project-based collaborative filtering algorithms is divided into three phases, namely model establishment, nearest neighbor finding, and recommendation generation [10-11]. Figure 1 shows the basic operation process of the collaborative filtering algorithm.

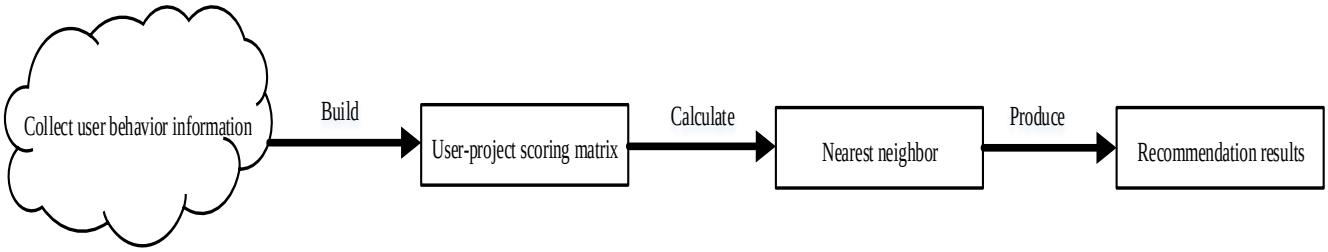

Figure 1. Basic steps of the collaboration algorithm

- Firstly, collect user ratings or other behavioral information to build the original user-item scoring matrix.
- Secondly, by calculating the similarity between target users and all other users, the user with the highest similarity is selected as the nearest neighbor set of target users.
- Thirdly, comprehensive nearest neighbor scores for score prediction and finally produces recommendation results.

At present, there are three main methods of similarity: Pearson correlation similarity, cosine similarity, and modified cosine similarity [12]. Since the modified cosine similarity principle is the same as the cosine similarity, with only a slight change in the formula, only the calculation formulas of the first two methods are introduced.

- Pearson coefficient correlation is shown in Equation (1). Let U_{ij} be the common user set of items i and j, and then the similarity sim (i, j) between item i and item j is measured by the Pearson correlation coefficient [13].

$$sim(i,j) = \frac{\sum_{c\in U_{ij}}(R_{ic} - \overline{R_i})(R_{jc} - \overline{R_j})}{\sqrt{\sum_{c\in U_{ij}}(R_{ic} - \overline{R_i})^2}\sqrt{\sum_{c\in U_{ij}}(R_{jc} - \overline{R_j})^2}} \tag{1}$$

Where R_{ic} and R_{jc} are the scores of user c on item i and item j, respectively, and $\overline{R_\iota}$ and $\overline{R_J}$ are the average scores of item i and item j, respectively.

- Cosine similarity is shown in Equation (2). The score obtained by the project is regarded as the vector in the n-dimensional user space [14]. If the user does not score the item, the user's rating on the item is set to 0, and the similarity between the items is passed through the cosine clip between the vectors.

$$sim(i,j) = \frac{\vec{\imath} \times \vec{\jmath}}{\|\vec{\imath}\| \times \|\vec{\jmath}\|} \tag{2}$$

Among them, the vector $\vec{\imath}$, $\vec{\jmath}$ is the score of the item i, j in the n-dimensional space. These two similarity methods can effectively calculate the similarity between users (projects). If a user is a new user or has too few items to score, there is no common item between the user and other users, resulting in an inability to calculate the similarity between the user and other users [15]. Therefore, in response to the problem caused by the extremely sparse user-item scoring matrix, researchers have proposed a variety of methods for pre-processing sparse data sets.

3. Single Value Fill Scoring Matrix

3.1. Single Value Data Filling Method

The filling method is a method of solving the sparsity problem by filling all the unevaluated items in the scoring matrix with a fixed value. Commonly used fixed values are selected by default, average, median, and mode. The user-item scoring matrix is shown in Table 1. Suppose there are n projects and m users, R_{ij} is the score of j for the first i project, and so on [15]. The average, median, and mode can be calculated respectively, and these three values are represented by users F_A, F_P, and F_M, respectively. It is assumed here that the user u does not overestimate the item v, I_u represents the item that has been evaluated by the user u, and U_v represents the set that has been evaluated excessively, as shown in Table 1.

Table 1. User-project scoring matrix

	Item 1	Item 2	……	Item *i*	.....	Item *n*
User1	R_{11}	R_{12}	……	R_{1i}	……	R_{1n}
User2	R_{21}	R_{22}	……	R_{2i}	……	R_{2n}
……	……	……	……	……	……	……
User *j*	R_{j1}	R_{j2}	……	R_{ji}	……	R_{jn}
……	……	……	……	……	……	……
User *m*	R_{m1}	R_{m2}	……	R_{mi}	……	R_{mn}

The fill calculation of the average is shown in Equation (3).

$$F_A = \begin{cases} \dfrac{^{x\in I_x}r_{ux} + ^{x\in U_V}r_{xv}}{\|I_u\| + \|U_v\|}, & \|I_u\| + \|U_v\| \neq 0 \\ 0, & \|I_u\| + \|U_v\| = 0 \end{cases} \tag{3}$$

Where $^{x\in I_x}r_{ux}$ is the sum of the scores of all items of user u on the line, $^{x\in U_V}r_{xv}$ is the sum of the scores of all users of the item *v*, $\|I_u\|$ the number of items evaluated by the line of user *u*, and $\|U_v\|$ is the number of users who have been scored for the items in a column.

The fill calculation of the median is shown in Equation (4).

$$F_P = \begin{cases} \dfrac{Mod_{x\in I_u}r_{ux} + Mod_{x\in U_v}r_{xv}}{2}, & \|I_u\| \neq 0, \|U_v\| \neq 0 \\ Mod_{x\in I_u}r_{ux}, & \|U_v\| = 0 \\ Mod_{x\in U_v}r_{xv}, & \|I_u\| \neq 0 \\ 0, & \text{Other} \end{cases} \tag{4}$$

Where $Mod_{x\in I_u}r_{ux}$ is the mode of the score matrix row, $Mod_{x\in U_v}r_{xv}$ is the mode of the score matrix column, $\|I_u\|$ is the number of items evaluated by user *u*'s row, and $\|U_v\|$ is the number of users who have been scored for the item in a column.

The fill calculation of the mode is shown in Equation (5).

$$F_M = \begin{cases} \dfrac{Median_{x\in I_u}r_{ux} + Median_{x\in U_v}r_{xv}}{\|I_u\| + \|U_v\|}, & \|I_u\| \neq 0, \|U_v\| \neq 0 \\ Median_{x\in I_u}r_{ux}, & \|U_v\| = 0 \\ Median_{x\in U_v}r_{xv}, & \|I_u\| \neq 0 \\ 0, & \text{Other} \end{cases} \tag{5}$$

Where $Median_{x\in I_u}r_{ux}$ is the median of the score matrix rows, $Median_{x\in U_v}r_{xv}$ is the median of the score matrix columns, $\|I_u\|$ is the number of items evaluated by user *u*'s row, and $\|U_v\|$ is the number of users who have been scored for the items in a column.

3.2. Single Value Filling Implementation

Single value filling can be calculated by using various values such as the average, median, and mode. The user-item scoring matrix can be filled by any one of the calculation methods, and the process is the same. In the case of mode padding, the steps are as follows:

- Step 1: Enter the user-item score matrix R_1.
- Step 2: Arbitrarily select an item I_{iy} in R_1 to determine whether the item I_{iy} is scored. If yes, continue to select any item I_{iy}, and if not, proceed to the next step.
- Step 3: Calculate the population filling of the project I_{iy}, and proceed to the next step.
- Step 4: Determine whether all ungraded matrix items are filled. If so, get the filled matrix R_2, and if not, continue to calculate the mode fill ungraded matrix item to form the filled matrix R_2.

- Step 5: Calculate the similarity based on the populated user-item scoring matrix R_2 to find a more realistic set of user neighbors.
- Step 6: Generate recommendation lists.

The specific filling flow is shown in Figure 2.

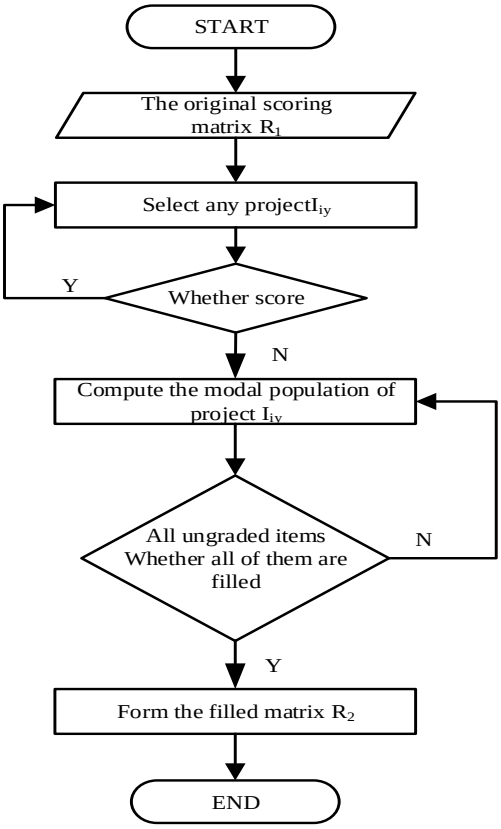

Figure 2. Single value (mode) fill flow chart

Although the single value filling method can alleviate the sparse problem to a certain extent, in the case of a large number of users and projects, all the single values are filled and the recommended calculation amount is also large, so it is suitable for a small-scale database. In real life, users will have some differences in the scoring of unevaluated items. This method is filled with uniform values, without taking into account differences in users' interests, and obliterates the user's personality. For example, when filling with the population fill method, the user may have multiple votes or no mode. Therefore, the application of the method of majority is more limited. In order to solve this problem, a filling rule is proposed, and the unrated items in each scoring matrix can have accurate values.

4. Filling the Mixed Data Scoring Matrix

At present, the fixed value is used to filled the empty position in the score matrix in order to alleviate the problem of data sparsity, which is the simple filling method. There are three methods for selecting fixed values, which are mean value, median, and mode, and each has its own advantages and disadvantages. They alleviate the problem of data sparsity to an extent but have limitations; for example, mean value filling is only suitable for small databases and eliminates the user's personalization when it is populated.

Mixed data filling fills the filling value of multiple fixed values (mean value, median, mode) through the appropriate filling rules to select a more suitable value for each vacant position in the scoring matrix.

4.1. Algorithm Flow

According to the architecture of the user- and item-based collaborative filtering algorithm and the existing data sparsity problem, a collaborative filtering algorithm based on data hybrid filling is proposed. It mainly completes the user and item

matrix building module and the nearest neighbor search module according to the filling rules. A new scoring matrix is generated, as shown in Figure 3.

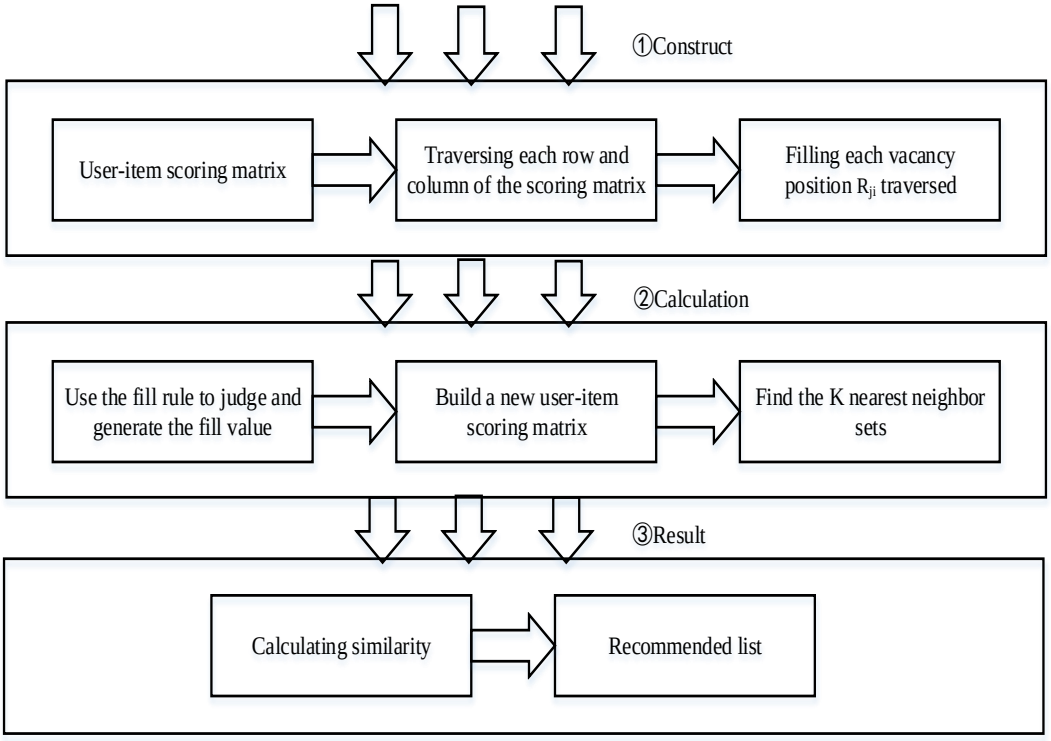

Figure 3. Based on the data fill algorithm flowchart mixing collaborative filtering

Combined with the above process, the algorithm considers all the existing filling methods, which alleviates the data sparsity problem to an extent. The original scoring matrix designs a new user item scoring matrix according to the filling rules to better find the nearest neighbor set and improve the accuracy of the recommendation.

4.2. Data Mixed Filling Rules

There are many methods to solve the data sparsity problem in the user-project scoring matrix, but the degree of mitigation is quite different. It is necessary to select appropriate and accurate values in various numerical filling methods to fill the vacancy in the scoring matrix. A new filling rule is proposed, and the padding value involved in the rule can at least be greater than 1. Calculate the weighted average of all the filling methods that can be considered, perform the range calculation for each filled value and the obtained weighted average value, and fill in the filling value corresponding to the minimum extreme difference value to the user-item score. The size of the weighted average value depends not only on the size of the filling value involved, but also on the frequency of each filling value. When multiple filling values are used in a user-item ungraded matrix, the weight will play a corresponding role. This rule takes three types of filling methods as the mean, median, and value.

The three filling methods of mean, median, and mode are respectively represented by F_A, F_P, and F_M, and their specific calculation processes are respectively shown in Equations (3), (4), and (5). The weighted average is represented by M, G_n ($n \in 1, 2, \cdots$) represents any one of the filling values, and W_n ($n \in 1, 2, \cdots$) is the number of times the filling value appears. Equation (6) and S_{ij} are expressed as a scoring item.

$$M = \frac{G_1 W_1 + G_2 W_2 + \cdots + G_n W_n}{W_1 + W_2 + \cdots + W_n} \tag{6}$$

The range between the mean F_A and the weighted mean M is represented by X, the range between the median F_P and the weighted mean M is represented by Y, and the range between the mode F_M and the weighted mean M is represented by Z. Equations (7) to (9) are as follows:

$$X = |F_A - M| \tag{7}$$

$$Y = |F_P - M| \tag{8}$$

$$Z = |F_M - M| \tag{9}$$

The median mode of the mean value is compared with the range value after completing the range calculation of the weighted average value, and the minimum value is selected. Equations (10) to (12) are as follows, and the filling value corresponding to its extreme value is filled into the vacancy position in the user-item score matrix.

$$if\ (X \leq Y \& X \leq Z) S_{ij} = F_A \tag{10}$$

$$if\ (Y \leq X \& Y \leq Z) S_{ij} = F_P \tag{11}$$

$$if\ (Z \leq X \& Z \leq Y) S_{ij} = F_M \tag{12}$$

4.3. Fill Rule Steps

The basic idea of the rule is the following: first, select an item in the user-item scoring matrix to determine whether it is not scored, calculate the mean, median, mode, and other filling methods for the unrated items, and then use Equation (6) to find the weighted average. The two perform the difference calculation to select the minimum number and fill in the unrated item. The specific filling steps are as follows:

- Step 1: Enter the user-item score matrix R_1.
- Step 2: Arbitrarily select an item I_{iy} in R_1 to determine whether the item I_{iy} is scored. If yes, continue to select any item I_{iy}, and if not, proceed to the next step.
- Step 3: Calculate the values of all the filling methods of the item I_{iy} to form the set Q, and proceed to the next step.
- Step 4: The formed set Q is subjected to the weighted average evaluation M.
- Step 5: Select the value Q_x of any filling method, $Q_x \in Q$, and calculate the difference between Q_x and the weighted mean M, $P = |Q_x\text{-}M|$.
- Step 6: Determine whether all values in Q have been calculated as the difference P from the weighted average M. If so, the padding value x corresponding to the value of the smallest difference P is selected to fill the user item matrix vacancy position. If not, it continues to calculate whether all values in Q have been calculated from the range P of the weighted average M.
- Step 7: Form the filled matrix R_2.
- Step 8: Calculate the similarity based on the populated user-item scoring matrix R_2 to find a more realistic set of user neighbors.
- Step 9: Generate recommendation lists.

The specific filling process is shown in Figure 4.

When the single value of the calculation is filled in the position of each vacancy in the scoring matrix, the filling method can be selected according to the filling rule calculation method, which not only solves the problem of data sparseness, but also can improve the accuracy of the vacancy position to find a more realistic set of user neighbors.

5. Experimental Results and Analysis

5.1. Data Set

The experimental environment of this paper is based on the cloud platform Hadoop1.2.1 pseudo-distributed cluster equipped with a Centos 6.4 operating system, one of which is the master node and two are the slave nodes. This experiment uses the Movie Lens data set. The score information table includes 943 users, 1,682 movies, and 100,000 ratings. The user's rating values for the project range from 1 to 5 as integers, and the test data is divided into training sets (80%). For the test set (20%), the training set is used to perform algorithm experiments and predictive estimation, and the test set is used to compare the predicted estimation results, so the sparsity can be calculated as: (943×1682-100000)/(943×1682) = 93.695% [16].

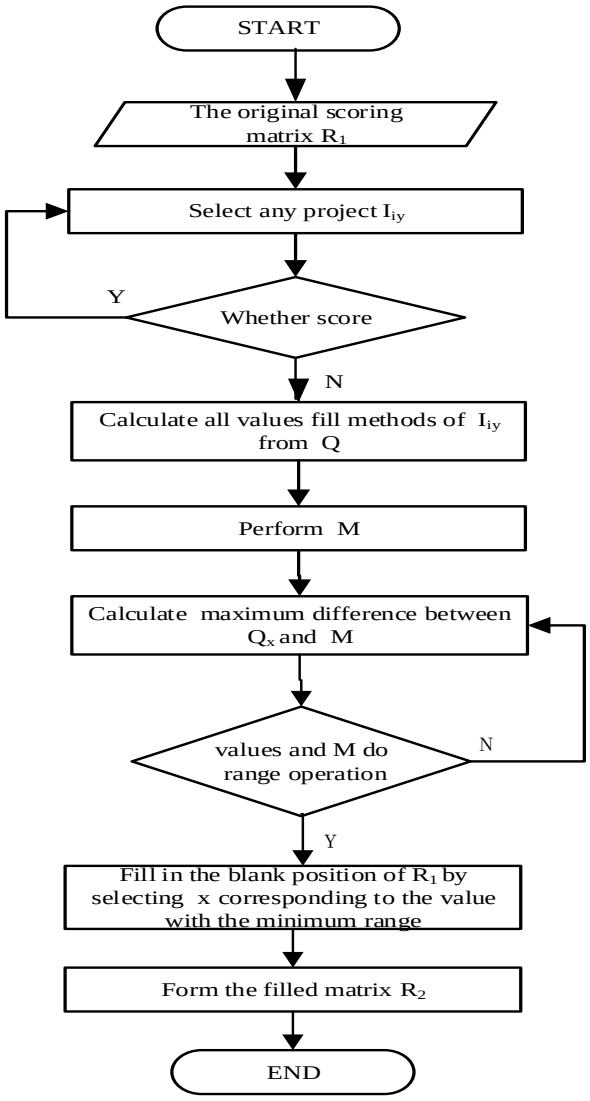

Figure 4. Multivariate value mixing fill flow chart

5.2. Evaluation Criteria

The criteria for evaluating the recommendation system recommendation quality are mainly statistical accuracy measurement methods and decision support accuracy measurement methods [17]. The recommended absolute precision is measured by the mean absolute error (MAE) and root mean square error (RMSE). MAE and RMSE represent the accuracy of the recommendation by calculating the difference between the predicted score and the actual score of the movie. The smaller the values of MAE and RMSE, the more accurate the recommendation. This paper uses the mean absolute error (MAE) to measure the recommendation accuracy. This method of measuring recommendation accuracy is easier to understand, and its essence is to calculate the average deviation between the predicted value and the true value. N is used to represent the number of predicted scores, p_i is used to represent the predicted value, and q_i is used to represent the real value. The expression of *MAE* is shown in Equations (13) and (14). The smaller the average absolute error value, the more accurate the recommended result, and the better the recommended algorithm performance [18].

$$MAE = \frac{\sum_{i=1}^{N} |p_i - q_i|}{N} \tag{13}$$

$$MAE = \sum_{i=1}^{M} \frac{MAE_i}{M} \tag{14}$$

In the formula, *M* is the total number of users and *MAE* is the absolute deviation of all users.

$$RMSE = \sqrt{\sum_{j=1}^{N_i} \frac{(p_{ij} - q_{ij})^2}{n}} \tag{15}$$

It can be seen from Equations (13) to (15) that if the calculated values of *MAE* and *RMSE* are smaller, the closer the theoretical result predicted by the algorithm is to the actual score, and the higher the accuracy of the algorithm recommendation.

5.3. Experimental Result

In order to make the evaluation conclusion more accurate, the filled data is used as the prediction score for recommendation, and the largest *N* of these predicted score values is taken as the recommendation values of the target users. The value range of *N* is [5, 20], with 5 as the step in this paper, and the comparison test uses the four kinds of padding data (mean, median, mode, multivariate mixed padding) as the recommended efficiency of recommendation and traditional collaborative filtering recommendation algorithm. The test results using the mean fill data as the predictive score are shown in Figure 5.

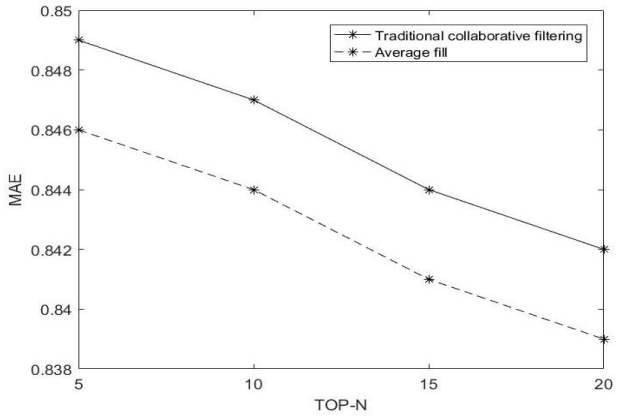

Figure 5. Mean filling and MAE comparison with traditional collaborative filtering algorithm

It can be seen from Figures 5 and 6 that the MAE value of the populating method of mean median mode is smaller than that of the traditional collaborative filtering recommendation algorithm, indicating that the recommendation effect of the three single-value fillers is good. Among the three filling methods, the recommended results are the best when using the median of the row and column of the grading matrix. They are the worst when using the average of the row and column of the scoring matrix. According to the comparison among the MAE values of the three filling methods and the multivariate value mixed filling method in Figure 5, the recommendation effect of the multivariate value mixed filling method is even better than that of the median filling method, which can alleviate the problem of data sparsity and at the same time guarantee better recommendation quality.

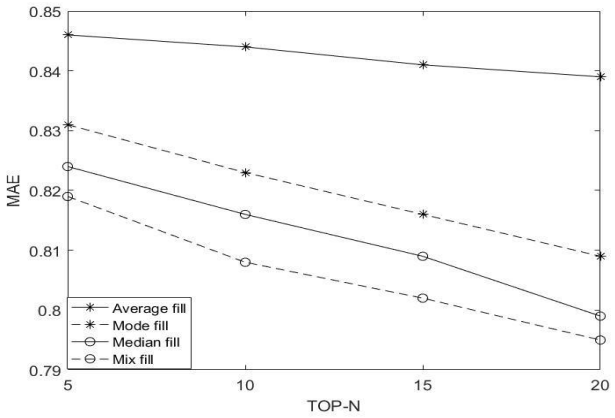

Figure 6. Comparison of MAE in four filling modes

6. Conclusions

This paper mainly studies the data sparse problem in traditional collaborative filtering recommendation. Considering that the solution of the data sparse problem generally adopts the method of filling in the value, the value of too single cannot truly represent the user's preference, and the problem of sparse scoring data cannot be fundamentally solved. Therefore, the method of multivariate value filling in the mitigation of user project data sparseness enriches the diversity of filling values and improves the original filling rules. The calculated weighted average is used to obtain the relatively concentrated value of the filling value, and the filling value is compared with the weighted average to find the filling value corresponding to the minimum extreme difference. The method is verified by experiments and effectively mitigates sparsity. In the future, we will conduct research on how to find the optimal weights in reasonable weighting values and the influence of user interests on many factors.

Acknowledgements

This work is sponsored by the National Natural Science Foundation of China (No. 61662017, 61862019, and 61262075,), and Guilin Science and Technology Project Fund (No. 2016010408).

References

1. B. J. Tian, P. P. Hu, and X. J. Du, "Optimization of Clustering based Collaborative Filtering Recommendation Algorithm under Hadoop," *Computer Engineering and Science*, Vol. 38, No. 8, pp. 1615-1624, 2016
2. C. Z. Xing and Y. Jin, "Collaborative Filtering Algorithm Combining Filling Method and Improved Similarity," Computer Application Research, 2019
3. G. L. Li, J. C. Ni, and P. P. Yu, "Weighted Slope One Algorithm based on Clustering and Spark Framework," *Computer Applications*, Vol. 37, No. 5, pp. 1287-1291, 2017
4. H. Yu and J. H. Li, "A Recommendation Algorithm to Solve the Cold Start Problem of New Projects," *Journal of Software*, Vol. 26, No. 6, pp. 1395-1408, 2015
5. J. F. Wang, Y. L. Miao, and P. F. Han, "A Probabilistic Matrix Decomposition Collaborative Filtering Recommendation Algorithm based on Trust Mechanism," *Miniature Microcomputer System*, Vol. 40, No. 1, pp. 31-35, 2019
6. J. X. Yu, L. M. Zang, and J. F. Wang, "A Collaborative Filtering Algorithm for Improving Similarity," *Journal of Henan University of Technology*, Vol. 38, No. 2, pp. 116-121, 2019
7. L. Li, Y. X. Dong, and C. H. Zhao, "Collaborative Filtering Recommendation Algorithm Combined with User Trust," *Journal of Chinese Computer Systems*, Vol. 38, No. 5, pp. 951-955, 2017
8. M. Win, M. B. Hynes, and A. Cater, "Algorithmic Acceleration of Parallel ALS for Collaborative Filtering: Speeding up Distributed Big Data Recommendation in Spark," in *Proceedings of International Conference on Parallel and Distributed Systems*, 2016
9. W. H. Li and S. R. Xu, "Design and Implementation of E-Commerce Recommendation System based on Hadoop," *Computer Engineering and Design*, Vol. 35, No. 1, pp. 130-136, 2014
10. W. Zan, X. Yu, and F. Nan, "An Improved Collaborative Movie Recommendation System using Computational Intelligence," *Journal of Visual Languages and Computing*, Vol. 25, No. 6, pp. 667-675, 2014
11. X. B. Guo, S. L. Zhao, and D. P. Niu, "A Combined Recommendation Method for Solving Sparse Data and Cold Start Problem," *Journal of University of Science and Technology of China*, Vol. 45, No. 10, pp. 804-812, 2015
12. X. D. Xiang and Z. X. Qi, "Research on Collaborative Filtering Algorithm based on Slope One Algorithm to Improve Scoring Matrix Filling," Computer Application Research, 2019
13. X. Zhong, G. Yang, and L. Li, "Clustering and Correlation based Collaborative Filtering Algorithm for Cloud Platform," *International Journal of Computer Science*, 2016
14. X. Y. Yang and J. Yu, "Collaborative Filtering Recommendation Model based on Trust Model Filling," *Computer Engineering*, Vol. 41, No. 5, pp. 6-13, 2015
15. Y. Gong and Q. Zhang, "Hashtag Recommendation using Attention-based Convolutional Neural Network," in *Proceedings of International Joint Conference on Artificial Intelligence*, AAAI Press, 2016
16. Q. Q. Shi, H. Z. Wang, D. Li, X. F. Shi, C. Ye, and H. Gao, "Maximal Influence Spread for Social Network based on MapReduce," *ICYCSEE*, pp. 128-136, 2015
17. Y. Song, H. Z. Wang, J. Z. Li, and H. Gao, "MapReduce for Big Data Analysis: Benefits, Limitations and Extensions," *ICYCSEE*, pp. 453-457, 2016
18. Y. Wang, H. Z. Wang, J. Z. Li, and H. Gao, "Efficient Graph Similarity Join for Information Integration on Graphs," *Frontiers of Computer Science*, Vol. 10, No. 2, pp. 317-329, 2016

Xiaohui Cheng received his bachelor's degree from Shanghai University of Technology in 1982. He is currently the dean of the School of Information Science and Engineering at Guilin University of Technology. He is also the director of the Key Laboratory of Guangxi University of Information and Manufacturing, the executive director of the Guangxi Computer Society, the director of the Embedded Systems Branch of the China Computer Software Industry Association, the executive

director of the Guangxi Computer Software Industry Association, a member of the Guangxi Natural Science Foundation Expert Committee, and the vice chairman of the Guangxi Measuring Instruments Industry Alliance. His current research interests include embedded systems and Internet of things technology.

Li Feng is a graduate student in the School of Computer Science and Technology at Guilin University of Technology. Her research interests include embedded systems and Internet of things technology.

Qiong Gui received her master's degree from Guilin University of Technology and her Ph.D. from the School of Information Engineering at Wuhan University of Technology. She is currently an associate professor and graduate instructor in the College of Information Science and Engineering. Her research interests include big data analysis and information security.