Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 2479314, 8 pages
https://doi.org/10.1155/2022/2479314

Research Article

Hybrid Collaborative Filtering Algorithm Based on Sparse Rating Matrix and User Preference

Hengtao Wang, Hongman Wang, Fangchun Yang, and Jinglin Li

Engineering Research Center of Information Network, Ministry of Education,
School of Computer Science (National Pilot Software Engineering School), Beijing University of Posts and Telecommunications, Beijing, China

Correspondence should be addressed to Hongman Wang; wanghm@bupt.edu.cn

Received 12 October 2021; Accepted 12 January 2022; Published 31 January 2022

Academic Editor: Yingjie Wang

This study presents a hybrid collaborative filtering recommendation algorithm for sparse data (HCFDS) to increase the recommendation impact by addressing the problem of data sparsity in standard collaborative filtering methods. To begin, the similarity calculation divergence is evident in a data sparse environment due to the difference in user scoring standards and the rise in weight of the same score in the overall score. The user similarity algorithm IU-CS and item similarity algorithm II-CS are suggested in this work by incorporating the score difference threshold and the same score penalty factor, in order to address the deviation of similarity computation caused by the excessive dilation. Second, this work offers a filling optimization technique for score prediction to address the issue of missing score matrix data. The II-CS algorithm presented in this work is used to forecast the missing items in the scoring matrix first, and then, the user's preference score in the item category dimension is utilized to correct the score prediction value and fill the matrix. Finally, the IU-CS method presented in this work is used in this study to provide recommendations on the filled score matrix. Experiments indicate that, when compared to the preoptimization method and other algorithms, the optimized algorithm successfully solves the problem of data sparsity and the recommendation accuracy is considerably increased.

1. Introduction

The fast advancement of Internet technology has resulted in a quick increase in data on the network, reducing the efficiency of information gathering significantly. The recommendation system [1] is a good way to tackle the difficulties mentioned above. It can suggest interesting material for users based on their previous information attributes and activities.

The algorithm at the heart of a recommendation system involves collaborative filtering, content, association rules, and hybrid recommendation [1]. Among these, the collaborative filtering algorithm [2] is the most widespread and is well-known, consisting mostly of two collaborative filtering algorithms based on user [3] and item [4]. The most significant distinction between the two algorithms is their "similarity" metric, which supports the recommendations' proposal. In both algorithms, "similarity" takes under consideration users' and items' similarities. However, the user-based collaborative filtering algorithm is to recommend according to the similarity between users, while the project-based collaborative filtering algorithm is to recommend according to the similarity between projects.

The calculation of similarity is the heart of the collaborative filtering algorithm [3], and data sparsity and similarity calculation techniques have an impact on the correctness of the output. The problem of data sparsity arises from the vast number of users and items in the recommendation system, and users are unable to rate all things, resulting in a substantial amount of data missing in the user-item scoring matrix, which has a direct impact on recommendation accuracy. When the score matrix data is sparse, the difference in user scoring standards and the weight of the same score in the overall score rises, and the calculation deviation of user similarity rises. Due to a lack of data on user ratings, each user's rating has a significant impact on the computation of item similarity, resulting in a divergence of results.

Domestic and international academics have been conducting extensive studies in order to address the data sparsity and similarity calculation accuracy issues.

To solve the problem of data sparsity, C. Li and Ma. [5] created the score matrix comprising the user's average weighted score and the item's average weighted score. Deng et al. [6] used a user-based collaborative filtering recommendation algorithm to make predictions, filling in the score gaps with the predicted values as the intermediate results, and then made recommendations based on item similarity. Yue Xi et al. [2] proposed first using the cosine similarity to calculate the similarity between items and then predicting the score to fill the data. The above method uses cosine similarity to predict the score when the data is sparse, and the accuracy of the predicted value is low.

The problem of data sparsity causes the cosine similarity computation to deviate. Ruonan Ji et al. [7] improved the modified cosine similarity of users by combining implicit feedback and time characteristics of users, and Wang and Zheng [3] weighted similarity to improve cosine similarity by considering the items of users' common score and the proportion in the sum of users' total score. X. Gao et al. [8] utilize cosine similarity to determine user similarity based on the modified score data and compensate for the score deviation caused by user status.

As far as the degree of similarity is concerned, Qu et al. [1] analyzes the degree of similarity of every characteristic of an item individually and then utilizes the neural network of BP (back-propagation) to obtain the final degree of similarity. Huo et al. [9] used the attenuation feature of the Logistic Equation for the purpose of adding the user's time component of interest to the calculation of similarity. Zhao et al. [10] and others use similarity analysis and weight calculation to identify the element's similarity to extract video features based on usage behavior, item name, and other factors.

The aforementioned research has used various techniques to enhance similarity calculation methods, the majority of which include criteria other than scores, and while the accuracy has increased to some extent, the data sparsity problem has not been addressed.

To address the problem of data sparsity, this paper offers a hybrid collaborative filtering recommendation method for sparse data (HCFDS). Firstly, in order to solve the problem of obvious deviation of similarity calculation due to the difference of user scoring standards and the increase of the weight of the same score in the total score in the data sparse environment, this algorithm introduces the scoring difference threshold and the same scoring penalty factor on the basis of the work by J. S. Breese et al. [11] and user similarity algorithm IU-CS and item similarity algorithm II-CS are proposed. Secondly, to address the problem of missing score matrix data, this study proposes a filling optimization approach for score prediction based on the work by Deng et al. [6]. In this method, the missing items in the scoring matrix are preliminarily predicted and scored by using the II-CS algorithm proposed in this paper, and then the predicted scores are corrected by using the user's preference scores in the item category dimension to improve the rationality of filling scores. Finally, the HCFDS algorithm uses IU-CS provided in this paper to make score predictions on the filled matrix. This technique overcomes the problem of data sparsity while simultaneously enhancing the accuracy of recommendations.

2. Methods

This paper offers a hybrid collaborative filtering recommendation algorithm for sparse data (HCFDS) to tackle the problem of data sparsity. To begin with, in the event of sparse data, the difference in user rating criteria and the weight of the same rating in the similarity contribution rise, resulting in a higher level of significant deviation in user similarity computation. Therefore, in this study, firstly, the user ratings are normalized, and then the user rating difference threshold and the same rating penalty factor are introduced, and a user similarity optimization algorithm IU-CS is proposed. After correcting the significant influence of users' difference in high scores on a single item and the influence of higher proportion of the same score in the total score on users' similarity, the algorithm introduces the threshold value of item score difference and the penalty factor of item score. This II-CS algorithm aims at the problems that (1) the single user's score difference between items is too large with fewer users score data and that (2) the weight of the same score in similarity contribution has a great influence on the calculation results of item similarity with few users. An item similarity optimization algorithm II-CS is proposed, which corrects the high score difference of individual users between items, reduces the contribution weight of the same score similarity of items, and optimizes the calculation of item similarity. Secondly, aiming at the problem of missing data in the scoring matrix, a scoring prediction filling optimization method is proposed, which uses the above-mentioned II-CS algorithm to preliminarily predict the score and then uses the item category preference score to correct the filling score. Finally, the IU-CS algorithm proposed above is used for recommendation on the filled matrix. The following is a description of the algorithm in this paper.

2.1. User Similarity Optimization Algorithm IU-CS Based on User Score Difference Threshold and Same Score Penalty Factor. Cosine similarity is a common computing method in collaborative filtering algorithm. The degree of similarity between users is shown in the following formula:

$$\mathrm{sim}(x, y) = \frac{|P(x) \cap P(y)|}{\sqrt{|P(x)||P(y)|}}. \tag{1}$$

Here, $P(x)$ represents the item set of user x's behavior.

Formula (1) measures the similarity between users according to the cooccurrence of users in items but does not consider the influence of popular items, that is, users' behaviors on popular items cannot reflect their personal preferences, whereas behaviors on some unpopular items can better explain their personal interests. Therefore, J. S. Breese et al. [11] punish popular items to reduce the contribution of popular items to user similarity, as shown in the following formula:

$$\operatorname{sim}(x, y) = \frac{\sum_{t\in V} (1/\log(1 + |Q(t)|))}{\sqrt{|P(x)||P(y)|}}. \quad (2)$$

Here, $Q(t)$ is the user set that produces behavior on item t, and V represents user x and user y common item set.

Formula (2) only considers the user's behavior on the item and does not consider the significant influence of the user's rating difference and the same rating weight on the similarity calculation under the condition of sparse data. In order to improve the accuracy of the results, the following improvements are made based on formula (2).

2.1.1. Normalization of User Ratings. Since different users have different rating standards, the same rating may represent different meanings, so it is necessary to normalize the rating according to the maximum value of users' rating, as shown in the following formula:

$$s'_{ut} = \frac{s_{ut}}{\max(R_u)}. \quad (3)$$

Here, s_{ut} represents the score of user u on item t, and s'_{ut} is the normalized result. $\max(R_u)$ represents the maximum score of user u.

2.1.2. Introducing User Score Difference Threshold to Correct the Significant Influence of High Rating Difference on User Similarity under Sparse Data. In the case of sparse data, the number of items evaluated by users is relatively small, and the difference of individual items between users will obviously influence the result of similarity calculation.

In order to correct the influence of user score difference on similarity under sparse data, a threshold of score difference α is introduced. When the absolute value of different users' score differences for the same item is below the threshold, it is considered that the score has a positive effect on similarity; otherwise, it is a bad influence, as shown in the following formula:

$$\operatorname{sim}(x, y) = \frac{\sum_{t\in V}\left(\left(\alpha - abs\left(s'_{xt} - s'_{yt}\right)\right)/\log(1 + |Q(t)|)\right)}{\sqrt{|P(x)||P(y)|}}. \quad (4)$$

In formula (4), s'_{xt} and s'_{yt} represent user x and user y's normalized ratings of item t.

2.1.3. Punishing the Similarity Contribution Weight of the Same Score with Less Common Items among Users. In formula (4), when users have the same score for the same item, there is the greatest influence on similarity. However, different users produce the same score for the same item with low frequency. Therefore, under normal circumstances, the same score accounts for a relatively low proportion in the positive contribution of similarity, but few items shared by users will lead to a higher proportion of the same score in the positive contribution of similarity, which will lead to the deviation of calculation results, especially in the case of sparse data. In order to reduce the influence of its high proportion on similarity, it is necessary to punish the same score with less common items and reduce its positive contribution to similarity.

On the basis of equation (4), the penalty factor λ_u of user's same score is introduced to punish the same score of users who have less common items and reduce their positive contribution to the similarity.

$$\operatorname{sim}(x, y) = \frac{\sum_{t\in V}\left(\left(\alpha - abs\left(s'_{xt} - s'_{yt}\right)\right)/\log(1 + |Q(t)|)\right)\bullet\varepsilon}{\sqrt{|P(x)||P(y)|}},$$

$$\varepsilon = \begin{cases} \dfrac{\min(\operatorname{Len}(V), \lambda_u)}{\lambda_u}, & s'_{xt} = s'_{yt}, \\ 1, & s'_{xt} \neq s'_{yt}. \end{cases} \quad (5)$$

Here, $\operatorname{Len}(V)$ is the common item set length of user x and user y.

2.2. An Optimization Algorithm II-CS for Item Similarity Degree Based on Item Difference Threshold and Penalty Factor of the Same Score. The similarity of items is improved with the idea of improving similarity of users. J. S. Breese et al. [11] penalize active users for improvements based on cosine similarity, as shown in the following equation:

$$\operatorname{sim}(a, b) = \frac{\sum_{x\in U} (1/\log(1 + |P(x)|))}{\sqrt{|Q(a)||Q(b)|}}. \quad (6)$$

Here, U stands for a set of users who score items a and b, $P(x)$ stands for a list of items for user x, and $|Q(a)|$ and $|Q(b)|$ stand for sets of users who score items a and b, respectively.

On the basis of formula (6), the factors such as the score difference between items and the number of common users of items are introduced for improvement.

2.2.1. Introducing the Threshold of Item Score Difference to Correct the Significant Influence of High Score Difference on Item Similarity under Sparse Data. In the case of sparse data, the user's item score data will be less, and the individual user's item score has a significant impact on the calculation of similarity, resulting in a deviation of results. In order to reduce the influence of individual user's item score on the calculation of similarity, the difference threshold β of item score is introduced. When the absolute value of user's item score difference is below the threshold, the similarity will increase; otherwise, the similarity will decrease.

The formula for calculating the similarity after the score is introduced is as follows:

$$\operatorname{sim}(a, b) = \frac{\sum_{x\in U} ((\beta - abs(s_{xa} - s_{xb}))/\log(1 + |P(x)|))}{\sqrt{|Q(a)||Q(b)|}}. \quad (7)$$

In formula (7), s_{xa} and s_{xb} represent user x's ratings of items a and b, respectively.

2.2.2. The Similarity Contribution Weight of the Same Score with Few Common Users Is Penalized. Because the ratings of different items by the same user may be influenced by internal factors such as personal preference, when there are few common users of items, it is difficult for users to exclude the influence of internal factors from the same ratings of different items. However, the positive contribution of the same ratings to the similarity of items at this time is relatively high, which leads to the deviation of calculation results. In the case of sparse data, the number of common users will be even less, which is particularly significant. Based on this situation, it is necessary to reduce its proportion to correct the influence of users' internal factors on item similarity. The penalty factor λ_i of item's same score is introduced to punish the item's same score with few common users on the basis of formula (7) and reduce its positive contribution to similarity. The final improved formula is as follows:

$$\operatorname{sim}(a,b) = \frac{\sum_{x\in U}\left(\left(\beta - abs\left(s_{xa} - s_{xb}\right)\right)/\log\left(1 + |P(x)|\right)\right)\bullet\varepsilon}{\sqrt{|Q(a)||Q(b)|}},$$

$$\varepsilon = \begin{cases} \dfrac{\min\left(\operatorname{Len}(U), \lambda_i\right)}{\lambda_i}, & s_{xa} = s_{xb}, \\ 1, & s_{xa} \neq s_{xb}. \end{cases} \tag{8}$$

Here, $\operatorname{Len}(U)$ is the length of the common user list for items a and b.

2.3. Optimization of Sparse Score Matrix Vacancy Prediction and Filling. For the missing values in the scoring matrix, this paper proposes an optimization method for the filling of scoring prediction; the item category preference score was used to revise the filling score.

2.3.1. Score Matrix Filling Based on II-CS. The score matrix is shown in Table 1. U_i represents user i, I_{i} represents item i, and the values in the matrix represent the user's score on the item. *?* indicates that the user has not scored an item.

Table 1: User-item scoring matrix.

	I_1	I_2	I_3	I_4
U_1	1	?	3	2
U_2	?	4	?	?
U_3	?	2	1	?
U_4	2	?	5	?

Because there are a lot of missing values in the matrix, it is impossible to calculate the user similarity when the data is sparse. For example, if you calculate the similarity between U_1 and U_2, you cannot use formula (5) to calculate the user similarity.

In order to solve the problem of data sparsity, firstly, the II-CS algorithm is used to predict the users' ungraded items based on the existing data in the scoring matrix, and then the results are filled into the matrix. The predicted score of user x for item a is calculated by the following formula:

$$F(x,a) = \overline{s_a} + \frac{\sum_{b\in n}\operatorname{sim}(a,b)\left(s_{xb} - \overline{s_a}\right)}{\sum_{b\in n}|\operatorname{sim}(a,b)|}. \tag{9}$$

Here, n is a similar set of items for a, s_{xb} is user x's rating of item b, and $\overline{s_a}$ is item a's average rating.

2.3.2. Integration with User Item Category Preference Score. Because the preliminary prediction score by formula (9) only depends on the user's scoring data, and in fact the user's scoring behavior will also be affected by their category preference, in order to make the filled score more reasonable, the prediction score is revised by calculating the user's category preference score for the item.

Each item may correspond to multiple categories, and the items-category matrix is shown in Table 2.

In Table 2, C_{ij} represents whether item i belongs to category j. If item i belongs to category j, the value is 1; otherwise, it is 0.

Combining the user-item scoring matrix and the item-category matrix, the user's score of the item category is calculated. User x ratings for item categories i are defined as follows:

$$f(x,i) = \frac{\sum_{C_{ai}=1, a\in P(x)} s_{xa}}{\operatorname{Len}(x,i)}. \tag{10}$$

Here, $\operatorname{Len}(x,i)$ is the number of items in category i in the list of items for user x, and a is the item for which user x has a score.

The degree of preference of users for different categories is different, and the degree of preference of users x for item categories i is calculated as shown in the following formula:

$$\varphi(x,i) = \frac{\operatorname{Len}(x,i)}{\operatorname{Len}(x)}. \tag{11}$$

Here, $\operatorname{Len}(x)$ is the number of items that the user has scored.

An item may belong to more than one category. The score of the item on the category preference dimension is influenced by the category preference degree and the category score. Comprehensive formulas (10) and (11) are used to score the category preference of user x for item a, as shown in the following formula:

$$G(x,a) = \frac{\sum_{i\in C(a)} f(x,i)\varphi(x,i)}{\sum_{i\in C(a)}\varphi(x,i)}. \tag{12}$$

In formula (12), $c(a)$ represents the category set to which item a belongs. The initial item-based collaborative filtering prediction score is revised using the user's rating on the item's category preference, and the final filled score matrix is as follows:

$$R(x,a) = kF(x,a) + (1-k)G(x,a). \tag{13}$$

In formula (13), k is the score correction coefficient.

2.4. Improved Algorithm Description. To solve the problem of data sparsity, this paper proposes a hybrid collaborative filtering algorithm for sparse data (HCFDS). Firstly, the missing items in the scoring matrix are preliminarily

Table 2: Items-category matrix.

	C_1	C_2	...	C_i
I_1	1	0	...	1
I_2	1	1	...	0
I_3	0	1	...	1
I_4	1	1	...	1

predicted and scored by using the II-CS algorithm proposed in this paper. Secondly, the user's preference score in the item category dimension is used to correct the score prediction value. Finally, the IU-CS algorithm proposed in this paper is used to make recommendations on the filled score matrix.

The pseudocode of HCFDS is as follows:

```
(i) HCFDS
(ii) **INPUT**: trainset, testset, cateitem, cateuser
(iii) **OUTPUT**: $L$
(iv) **BEGIN**:
(1) FOR item $a \in$ trainset
(2) FOR item $b \in$ trainset
(3) IF $a\ != b$
(4) $sim(a,b)$ = formula (8)
(5) END FOR
(6) END FOR
(7) FOR user $x \in$ trainset
(8) FOR item $a \in$ itemset $(x)$ − ownset $(x)$
(9) IF cateitem$(a)$ ∩cateuser$(x)$ $!= \phi$
(10) $G(x,a)$ = formula (12)
(11) $R(x, a)$ = formula (13)
(12) trainset.*add* (user, item, $R(x, a)$)
(13) END FOR
(14) END FOR
(15) FOR user $x \in$ trainset
(16) FOR user $y \in$ trainset
(17) IF $x\ != y$
(18) $sim\ (x, y)$ = formula (5)
(19) END FOR
(20) END FOR
(21) FOR user $u \in$ testset
(22) FOR item $a \in$ itemset $(u)$ − ownset $(u)$
(23) $P(u,a)$ = formula (14)
(24) END FOR
(25) END FOR
(26) $L = \mathrm{sort}(u, P, N)$
(i) **RETURN** $L$
(ii) **END**
```

In the pseudocode, trainset stands for training set, testset stands for test set, L stands for a user generated list of recommendations of length N, itemset stands for item set, ownset stands for user-owned item set, cateitem stands for category set, and category user represents a collection of user-owned item categories.

The final algorithmic steps in this article are as follows:

(i) Step 1: on the initial dataset, HCFDS uses the item II-CS optimization algorithm to calculate the item similarity by formula (8).

(ii) Step 2: using the similarity of items in step 1, the user's unrated items in the user-item scoring matrix are scored and predicted. Select the nearest neighbor of the item and get the preliminary prediction score $F(x,a)$ by formula (9).

(iii) Step 3: HCFDS calculates the user's score $G(x,a)$ on category preference by formula (12), revises the initial predicted score in step 2, gets the final filling score $R(x,a)$, fills in the missing items in the scoring matrix, and solves the problem of data sparsity.

(iv) Step 4: on the basis of filling the matrix in step 3, using IU-CS optimization algorithm, the user similarity is calculated by formula (5).

(v) Step 5: HCFDS uses the user similarity in step 4 to predict the score. It selects the user's nearest neighbor and calculates the prediction score of the user's unrated items by the following formula:

$$P(x,a) = \overline{s_x} + \frac{\sum_{y\in m} \mathrm{sim}(x,y)\left(y_a - \overline{s_y}\right)}{\sum_{y\in m} |\mathrm{sim}(x,y)|}. \tag{14}$$

(vi) In formula (14), m is a similar set of users x, y_a is the score of user y on item a, and $\overline{s_x}$ and $\overline{s_y}$ are the average scores of users x and y.

(vii) Step 6: according to the score prediction results ranking, HCFDS selects the highest score of the first N items for recommendation.

3. Experiments

Aiming at the above-mentioned algorithms proposed in this paper, firstly, the comparison experiments of user-based and item-based collaborative filtering algorithm before and after improvement are carried out to verify the effectiveness of the improved similarity calculation. Then, the comparison experiments are carried out between the final optimized algorithm HCFDS and the improved user-based collaborative filtering algorithm to verify the effectiveness of the HCFDS algorithm in solving the data sparsity problem. Finally, the experimental comparison is made between the HCFDS algorithm and other algorithms to verify the improved performance of the algorithm.

3.1. Dataset. The dataset used in this experiment is MovieLens dataset provided by GroupLens project team of University of Minnesota. The scale of data used in the experiment is 100 K, including 100,000 pieces of historical scoring data of 1682 movies by 943 users. $u1$ base is selected as the training set, and $u1$ test is selected as the test set.

3.2. Evaluation Metrics. In this paper, the Mean Absolute Error (MAE) is used as an evaluation index, which reflects the error between the predicted value and the true value of the item score. The smaller the value of MAE, the higher the accuracy of the improved algorithm. If the predicted score is $\{r_1, r_2, r_3, \ldots, r_N\}$ and the actual score is$\{s_1, s_2, s_3, \ldots, s_N\}$, the calculation formula is as follows:

$$\mathrm{MAE} = \frac{\sum_{i=1}^{N} |r_i - s_i|}{N}. \quad (15)$$

In formula (15), N is the number of prediction scores, r_i is the prediction score of item i, and s_i is the actual score of item i.

3.3. Experimental Results and Analysis

3.3.1. Comparison before and after Improvement of User-Based Collaborative Filtering Algorithm. The enhanced similarity calculation formula (5) is used to improve the user-based collaborative filtering (UBCF) algorithm. The user rating difference threshold and the same rating penalty factor choose the value that corresponds to the minimal MAE and different neighbors K for trials, respectively. As demonstrated in Figure 1, the outcomes before and after the algorithm improvement are compared.

It can be seen from Figure 1 that MAE decreases with the increase of the number of neighbors K before and after the improvement of the algorithm, indicating that the more the number of neighbors is, the more accurate the prediction result is. It can be clearly seen from the figure that the improved algorithm has smaller MAE under different K values, and when K is 5, it increases by 3.6%. With the increase of K value, MAE tends to be stable and increases by 1.5% when K is 50. The result of scoring prediction is more accurate, which shows that the improvement can improve the algorithm's performance.

3.3.2. Comparison before and after Improvement of Item-Based Collaborative Filtering Algorithm. The item-based collaborative filtering technique is improved using the improved item similarity calculation formula (8). The best solution is the equivalent value when MAE is the least, and different neighbors K are chosen for the experiment based on the difference threshold of item score and the penalty factor of the same score. Figure 2 depicts a comparison of the outcomes before and after the algorithm modification.

It can be seen from Figure 2 that, under different neighbor numbers K, the MAE of the improved algorithm is significantly lower than that before the improvement, and when K is smaller, it is increased by 4.7% when K is 5. With the increase of K value, MAE tends to be stable and increases by 4.5% when K is 50. It is shown that the improved algorithm is effective and can improve the accuracy of prediction.

3.3.3. Comparison between HCFDS Algorithm and Improved UBCF Algorithm. To solve the problem of data sparsity, this paper uses the improved item-based collaborative filtering

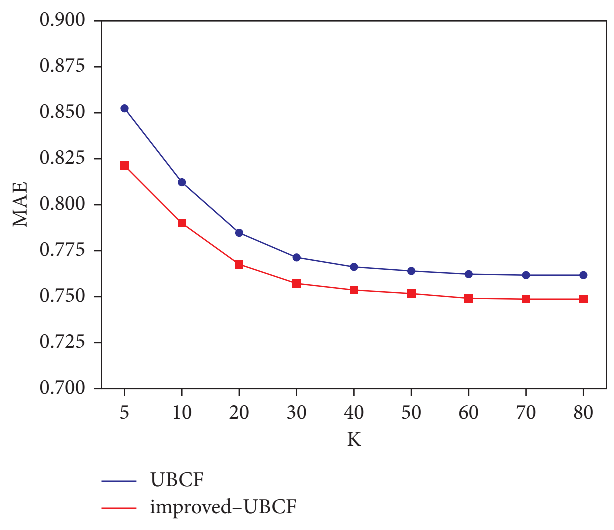

Figure 1: MAE before and after improvement of UBCF algorithm under different K values.

prediction score to fill the missing items in the score matrix, so that the data distribution becomes dense. The sparsity [2] is introduced to measure the sparsity of data, as shown in the following formula:

$$d = 1 - \frac{T}{|U| \cdot |I|}. \quad (16)$$

In formula (16), T represents the size of the initial dataset, and $|U|$ and $|I|$ represent the numbers of users and items, respectively.

In this paper, the sparsity of the dataset is 0.949 before filling, and it is 0.720 after filling, which is 5.5 times lower than that before filling, so the sparsity problem is solved effectively.

In order to verify the influence of solving the problem of data sparsity on the algorithm results, the improved UBCF algorithm is used to generate recommendations on the filled datasets. A comparison between the final optimization algorithm HCFDS and the improved UBCF algorithm is shown in Figure 3.

The improved UBCF algorithm in the figure is based on the unfilled original dataset and has the problem of data sparsity. The final optimization algorithm HCFDS is recommended based on the filled dataset, which solves the problem of data sparseness.

The overall accuracy of the HCFDS algorithm was improved significantly before the comparison. When K is smaller, that is, is 5, it increases by 7.9%. With the increase of K value, MAE tended to be stable. When K was 50, MAE increased by 1.6%. Therefore, the HCFDS algorithm proposed in this paper is effective and can improve the accuracy of recommendation.

3.3.4. Comparison of Different Optimized Recommendation Algorithms. To verify that the accuracy of HCFDS is better than that of the single optimization similarity algorithm or that of the matrix filling algorithm, the optimization

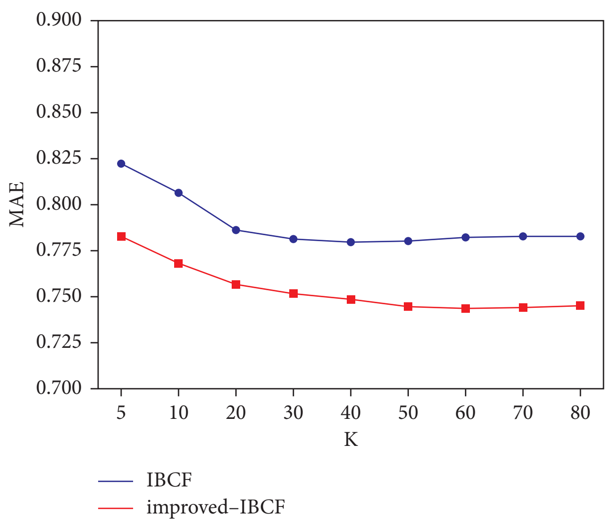

Figure 2: MAE before and after improvement of IBCF algorithm under different K values.

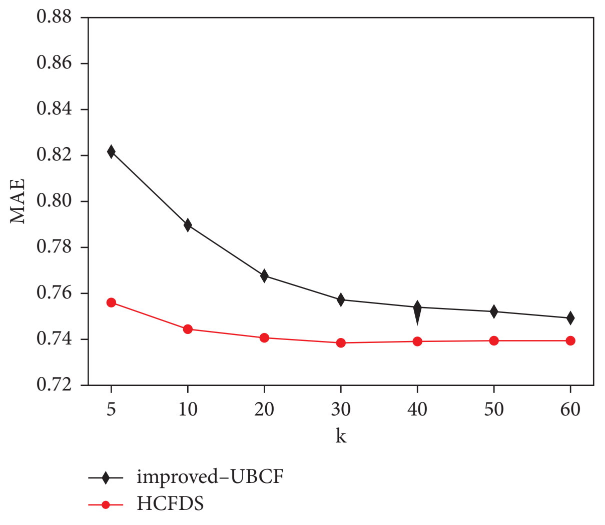

Figure 3: Comparison of HCFDS with modified UBCF under different K values.

algorithm HCFDS in this paper is compared with the optimization algorithm of Deng et al. [6], the optimization algorithm of support weight of Wang and Zheng. [3], and the optimization algorithm proposed by X. Gao et al. [8] based on cosine similarity, and the results are shown in Figure 4.

It can be seen from Figure 4 that the fluctuation of each optimization algorithm tends to be stable with the increase of K value, and the gap decreases. However, regardless of the value of K, the accuracy of the HCFDS algorithm suggested in this study is always superior to those of other algorithms. When K value is small, such as $K = 5$, it is 5.1% higher than Wang Wei, 5.5% higher than Deng et al., and 1.8% higher than X. Gao et al. With the increase of K value, MAE tends to be stable. when $K = 30$, it is 2.7% higher than Wang and Zheng, 3.9% higher than Deng et al., and 2.2% higher than X. Gao et al. Therefore, the HCFDS algorithm proposed in this paper has higher accuracy.

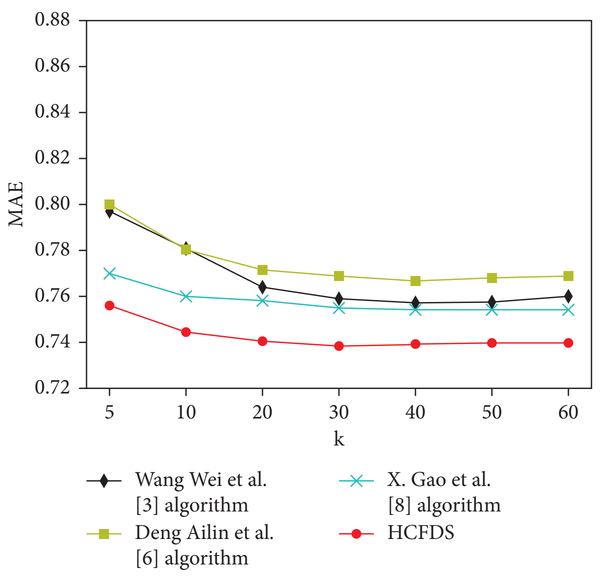

Figure 4: Comparison of different optimal recommendation algorithms under different K value.

4. Conclusion

In this paper, an improved hybrid collaborative filtering algorithm (HCFDS) is proposed to solve the problem of data sparsity in traditional collaborative filtering algorithms. To begin, in a sparse data environment, the user rating is normalized to unify the rating standards owing to differences in user rating standards and the rise in the weight of the same user rating in the overall rating. Following that, the IU-CS user similarity optimization method is presented, which incorporates the user rating difference threshold and the same rating penalty factor. In view of the problems that the difference of individual users' scores between items is too large when users' scoring data on items are few and the weight of the same scores of items increases in similarity contribution when users are few, this algorithm introduces item scoring difference threshold and penalty factor of the same scores of items and an item similarity optimization algorithm II-CS is proposed to correct the significant influence of the difference of individual users' or items' scores and the high proportion of the same scores in the total scores on similarity calculation under the condition of sparse data. Secondly, aiming at the problem of missing data in the scoring matrix, this study proposes a scoring prediction filling optimization method. In this method, aiming at the missing items in the scoring matrix, the II-CS algorithm proposed in this paper is used to preliminarily predict the score, and then the item category preference score is used to correct the predicted score, so that the filling value is reasonable. Finally, the algorithm uses the IU-CS algorithm proposed in this paper to make recommendations on the filled matrix. The aforementioned methods suggested in this study are evaluated on the public dataset MovieLens against the unoptimized algorithm and other algorithms. When K is 5, MAE is 7.9% greater than the modified UBCF algorithm, 5.1% higher than Wang and Zheng, 5.5% higher than Deng et al., and 1.8% higher than X. Gao et al. MAE becomes more stable as the K value increases. When $K = 30$, it outperforms the modified UBCF algorithm by 2.4%, Wang and Zheng by 2.7%, Deng et al. by 3.9%, and X. Gao et al. by 2.2%. As a result, the HCFDS method presented in this work may considerably increase score prediction accuracy and successfully address the problem.

Data Availability

Previously reported MovieLens dataset data were used to support this study and are available at https://movielens.umn.edu. These prior studies (and datasets) are cited at relevant places within the text as references [9].

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the https://doi.org/10.13039/501100001809National Natural Science Foundation of China (61876023).

References

[1] Z. Qu, J. Yao, X. Wang, and S. Yin, "Attribute weighting and samples sampling for collaborative filtering," in *Proceedings of the IEEE International Conference on Big Data & Smart Computing.*, January 2018.

[2] Y. Xi, T. Dan, S. Hongping, and A. Yiwen, "Improvement of collaborative filtering recommendation algorithm based on data sparsity," *Engineering Science and technology*, vol. 52, no. 1, 2020.

[3] W. Wang and J. Zheng, "Improvement of collaborative filtering algorithm based on user similarity," *Journal of East China Normal University*, no. 3, pp. 60–66, 2016.

[4] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: item-to-item collaborative filtering," *IEEE Internet Computing*, vol. 7, no. 1, pp. 76–80, 2003.

[5] C. Li and L. Ma, "Item-based collaborative filtering algorithm based on group weighted rating," in *Proceedings of the 2020 13th International Symposium on Computational Intelligence and Design (ISCID)*, pp. 114–117, Hangzhou, China, December 2020.

[6] A. Deng, Y. Zhu, and B. Shi, "Collaborative filtering recommendation algorithm based on item scoring prediction," *Journal of Software Engineering*, vol. 14, no. 09, pp. 1621–1628, 2003.

[7] R. Ji, Y. Tian, and M. Ma, "Collaborative filtering recommendation algorithm based on user characteristics," in *Proceedings of the 2020 5th International Conference on Control, Robotics and Cybernetics (CRC)*, pp. 56–60, Wuhan, China, October 2020.

[8] X. Gao, Z. Zhu, X. Hao, and H. Yu, "An effective collaborative filtering algorithm based on adjusted user-item rating matrix," in *Proceedings of the 2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA)*, pp. 693–696, Beijing, Chna, March 2017.

[9] H. Huo, P. Zhu, and H. Zhang, "Time context unifying collaborative filtering," in *Proceedings of the 2016 2nd IEEE International Conference on Computer and Communications (ICCC)*, pp. 1274–1278, Chengdu, China, October 2016.

[10] N. Zhao, P. Wenchao, and C. Xu, "A video recommendation algorithm for multidimensional feature analysis filtering," *Computer Science*, vol. 47, no. 04, pp. 103–107, 2020.

[11] J. S. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," in *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, Madison WI, USA, July 1998.