Hindawi
Mathematical Problems in Engineering
Volume 2022, Article ID 2340671, 7 pages
https://doi.org/10.1155/2022/2340671

Research Article

A New Collaborative Filtering Algorithm Integrating Time and Multisimilarity

Qin Liu

Department of Information Science and Technology, East China University of Political Science and Law, Shanghai 200042, China

Correspondence should be addressed to Qin Liu; liuqin@ecupl.edu.cn

Received 30 April 2022; Revised 27 May 2022; Accepted 10 August 2022; Published 30 August 2022

Academic Editor: Chaoqun Duan

Aiming at the problem of low recommendation accuracy of existing recommendation algorithms, an algorithm integrating time factors and multisimilarity is proposed to improve the impact of long-term data, user attention, and project popularity on the recommendation algorithm and the similarity of user attributes is introduced to improve the problem of cold start to a certain extent. Considering that the longer the time, the less likely it is to be selected again, time is introduced into the algorithm as a weight factor. When the behavior occurs, i.e., interest in the project, so as to judge the similarity between users, not just the score value, we normalize the popularity to avoid misjudgment of high scoring and popular items. Because new users do not have past score records, the problem of cold start can be solved by calculating the similarity of user attributes. Through the comparative experiment on Movielens100K dataset and Epinions dataset, the results show that the algorithm can improve the accuracy of recommendation and give users a better recommendation effect.

1. Introduction

In work and life, people rely more and more on computers and mobile devices. They face a large amount of data every day. At the same time, each user will also produce a large amount of data. How to select the information they need from a large amount of data depends on various factors; many users will choose the recommendation given by others, and the recommendation system also came into being. The recommendation system is applied in more and more fields, including commodity recommendation, learning resource recommendation, and case recommendation in intelligent justice [1–3]. Recommendation algorithms can be roughly divided into knowledge-based recommendation, collaborative filtering-based recommendation, and content-based recommendation. Among them, the recommendation algorithm based on collaborative filtering has always been the most classic and popular recommendation algorithm. Collaborative filtering recommendation algorithms are divided into two categories: collaborative filtering based on selection items and collaborative filtering based on participating users. They both predict preferences and generate recommendations according to users' historical scoring data. The collaborative filtering recommendation algorithm has many problems, such as unreasonable similarity calculation, cold start, sparse data, and so on, which may lead to the decline of recommendation quality.

From different perspectives, many studies fully tap the potential information so that the collaborative filtering recommendation algorithm can be used to a wider range of scenarios. Literature [4] proposed a method to calculate the similarity of items utilized score and structural similarity, which can effectively work out the problem of poor effect of cold start. Reference [5] proposed a method of weighting items to obtain a user similarity calculation method considering the similarity weight of items. Literature [6] projected a method to calculate the similarity of items by combining the similarity of scores and the similarity of item attributes. Reference [7] proposed a similarity calculation method between users based on cosine similarity and fusion of relative differences in scores. In the nonpreference scoring system, the prediction error can be reduced and the recommendation quality can be improved. Literature [8] proposed this algorithm to cluster tags and generate topic tag

clusters, calculate the correlation between items and topics according to item labeling, and generate item g topic correlation matrix. At the same time, it is combined with item g score matrix to calculate the similarity between items, and collaborative filtering is used to complete the score prediction of target items, so as to realize personalized recommendation. The above methods are to enhance the accuracy of similarity calculation by fusing scores and other indicators. However, they did not consider the impact of the environment on user interest. In fact, there are many environmental factors affecting user interest and interest changes, such as time, weather, place, and mood; among them, the time factor is the main factor affecting user interest and interest changes. Therefore, this paper declares a collaborative filtering recommendation algorithm on the ground of time weight and multisimilarity. Multisimilarity includes the similarity of user attributes, user attention, and project popularity. The similarity of user attributes alleviates the problem of cold start to a certain extent, User attention and project popularity can reduce some significant "unfairness" in the existing scores from the perspective of users and projects, respectively.

2. Related Work

2.1. User-Based Collaborative Filtering Algorithm. The user-based collaborative filtering algorithm will involve user U, item I, and user rating matrix for items R_{ui}. The chief steps of the traditional user-based collaborative filtering algorithm are as follows: first, build the user's score matrix, and mark the unknown score as 0, as shown in Table 1; the second step is to estimate the similarity between users by using the scoring matrix; in the third step, according to the similarity matrix, some users who are the most alike to the target user, that is, the users with the highest similarity are found to form a nearest neighbor set; the fourth step is to predict the score of the target user according to the score of the neighbor user and generate a recommendation set. Among them, the calculation of similarity between users is the important step of the algorithm.

2.2. Calculation of User Similarity. The traditional user similarity calculation formulas mainly include COS (cosine similarity), PCC (Pearson correlation coefficient), and ACOS (adjusted cosine similarity) [1, 8–11]. Formula (1) means the cosine similarity between user u and user v, Formula (2) represents the Pearson correlation coefficient, and Formula (3) represents the adjusted cosine similarity. Table 2 shows the meanings of symbols used in the formula.

$$\mathrm{sim}(u,v)_{\mathrm{COS}} = \frac{\sum_{c\in I_{u,v}} r_{u,c} r_{v,c}}{\sqrt{\sum_{c\in I_u} r_{u,c}^2}\sqrt{\sum_{c\in I_v} r_{v,c}^2}}, \tag{1}$$

$$\mathrm{sim}(u,v)_{\mathrm{PCC}} = \frac{\sum_{c\in I_{u,v}} \left(r_{u,c} - \overline{r_u}\right)\left(r_{v,c} - \overline{r_v}\right)}{\sqrt{\sum_{c\in I_{u,v}} \left(r_{u,c} - \overline{r_u}\right)^2}\sqrt{\sum_{c\in I_{u,v}} \left(r_{v,c} - \overline{r_v}\right)^2}}, \tag{2}$$

Table 1: User item scoring.

	I_1	I_2	...	I_n
U_1	$r_{1,1}$	$r_{1,2}$	...	$r_{1,n}$
U_2	$r_{2,1}$	$r_{2,2}$	...	$r_{2,n}$
...	...	...	...	...
U_m	$r_{m,1}$	$r_{m,2}$	...	$r_{m,n}$

Table 2: Meaning of symbols in formulas.

Symbol	Meaning
$r_{u,c}$	User u's rating of item C
I_u	Collection of items with user u rated scores
$I_{u,v}$	A collection of items that users u and V have jointly rated
$\overline{r_u}$	Average value of user U's rating on all items

$$\mathrm{sim}(u,v)_{\mathrm{ACOS}} = \frac{\sum_{c\in I_{u,v}} \left(r_{u,c} - \overline{r_u}\right)\left(r_{v,c} - \overline{r_v}\right)}{\sqrt{\sum_{c\in I_u} \left(r_{u,c} - \overline{r_u}\right)^2}\sqrt{\sum_{c\in I_v} \left(r_{v,c} - \overline{r_v}\right)^2}}. \tag{3}$$

Formulas (2) and (3) take into account users' personal scoring habits. Some users prefer to give high scores and some users give low scores. Therefore, Formulas (2) and (3) calculates users' similarity after subtracting users' average scores, which can improve the recommendation quality. However, for some nonuser preference scoring systems, the scores are given by specific standards, and Formulas (2) and (3) are difficult to be compared for the differences between users.

3. Collaborative Filtering Recommendation Algorithm Combining Time and Multisimilarity

3.1. Similarity Calculation considering Time Factor. At present, the research on the time factor of recommendation system mainly has three aspects [12]: first, obtain the information related to users through modeling so as to study the relationship between user interest and time and provide reasonable recommendations for users, but it is not easy to obtain the characteristics of user information by the modeling method, and it is not universal. The second aspect is to set the effective time limit for the score. Only the items within the effective time range will calculate their similarity; otherwise, it will be ignored, which greatly decreases calculation difficulty. However, it only considers the impact of recent data and denies the impact of long-term data on recommendation. The third aspect is to use the time weighting method and introduce the forgetting curve to obtain the time attenuation function.

In this paper, the time weight function [12] is introduced. As shown in Formula (4), $t_{u,c}$ represents the degree of interest of user u in item c at a certain time, which can be expressed by score $r_{u,c}$. $w(t_{u,c})$ is a monotonically increasing function, whose value increases with the increase in time t, but it will not exceed the upper limit 1. Considering that the longer the time is, the less likely it is to be selected again, time is introduced into the algorithm as a weight factor. We

enhance the similarity formula based on the modified cosine similarity (Formula (3)). After introducing the time weight, the specific calculation formula is shown as

$$w\left(t_{u,c}\right) = \frac{1}{1 + e^{\left(-t_{u,c}\right)}}, \tag{4}$$

$$\operatorname{sim}(u,v)_w = \frac{\sum_{c \in I_{u,v}} \left(r_{u,c} * w\left(t_{u,c}\right) - \overline{r_u}\right)\left(r_{v,c} * w\left(t_{v,c}\right) - \overline{r_v}\right)}{\sqrt{\sum_{c \in I_u} \left(r_{u,c} * w\left(t_{u,c}\right) - \overline{r_u}\right)^2} \sqrt{\sum_{c \in I_v} \left(r_{v,c} * w\left(t_{v,c}\right) - \overline{r_v}\right)^2}}. \tag{5}$$

3.2. Similarity Calculation of User Attributes. Because new users have no past score records, it is difficult for the actual program to predict and recommend products to these users. Generally, they will choose to recommend popular products to new users. In fact, the system can use some attribute information provided during user registration for recommendations so as to clear up the problem of cold start to a certain degree extent [13].

Extract the necessary information for new user registration, including gender, occupation, age, address, and income, in this paper, and the attributes are divided into tree (hierarchical) attributes and linear attributes. Linear attribute refers to the linear juxtaposition relationship between attribute values that does not distinguish between front and rear positions. For example, gender attribute, age, and income fields can be segmented and identified. For example, age can be divided into five categories: less than 18 years old, coded as one, 19–28 years old, coded as two, 29–40 years old, coded as three, 41–65 years old, coded as four, and over 66 years old, coded as five. Tree attribute refers to the parent-child relationship between attribute values in the hierarchical or tree structure. For example, there are many types of traditional occupations, such as occupation and address. Each occupation corresponds to at least one main industry. According to the industry classification of the national economy, the industry is divided into 20 categories, including the major category, medium category, and small category. Considering the potential semantic relationship between values, a classification semantic hierarchy tree based on domain knowledge is constructed. The leaf nodes of the tree are different attribute values, and the similarity between attribute values depends on their position in the tree structure.

The formula for calculating the similarity of linear attributes is

$$\operatorname{sim}(u,v) = \frac{1}{1 + |u - v|}. \tag{6}$$

The formula for calculating the similarity of tree attributes is

$$\operatorname{sim}(u,v) = 1 - \frac{L(u,v)}{H}, \tag{7}$$

where H is the height of the hierarchical tree and $L(u,v)$ is the longest path length from the attribute node in the tree to the common node.

3.3. User Attention Similarity. The traditional similarity calculation is for the user's rating of the project. If the score similarity is high, the user similarity is high, and if the score similarity is low, the user similarity is low. In fact, this method ignores the behavior of the project, which can explain the similarity between users to a certain extent. Because usually only the items that users are interested in will produce behavior, and the score cannot completely explain whether users are interested in the field of the project [7]. For example, when a user buys an AI book, he cannot understand the content of the book due to his own knowledge structure, so he gives a low score, but this does not mean that he is not interested in AI books. Therefore, we can think that the behavior is interested in the project, so as to judge the similarity between users, not just the score value. The calculation formula of user attention similarity is

$$\operatorname{sim}(u,v) = \frac{|Iu, v|}{|Iu| + |Iv| - |Iu, v|}, \tag{8}$$

where $|Iu, v|i$ is the number of common scoring items of users u and V and $|Iu||Iv|i$ is the number of scored items of users u and V.

3.4. Project Popularity. Project popularity is a factor that will be considered in many recommendation algorithms, because generally goods with high popularity are easier to be found by users, which will have greater influence on users than other goods. The existing algorithms generally set the weight of popularity to reduce the role of popular items in similarity calculation and final recommendation. However, it is found that this method is unfair to high scoring and popular projects. Therefore, before calculating the weight, the popularity can be normalized [13], as shown in formula (9). $IP_i i$ is the popularity of item I, which can be expressed by the frequency of item I in the data. Calculate the weight of each item according to the normalized popularity as

$$\operatorname{nor}IP_i = \frac{IP_i - \min IP}{\max IP - \min IP}, \tag{9}$$

$$w_i = w_0 * \frac{\operatorname{nor}IP_i}{\sum_{j=1}^{N} \operatorname{nor}IP_j}. \tag{10}$$

3.5. Improved Algorithm. Through the above analysis, we propose a collaborative filtering recommendation algorithm (COS-MS) integrating time and multisimilarity. The basic steps are in Algorithm 1.

4. Experimental Analysis

4.1. Dataset. In this paper, Movielens100K dataset and Epinions dataset are used for experiments. The reason why the two datasets are used is that the sparsity of the two

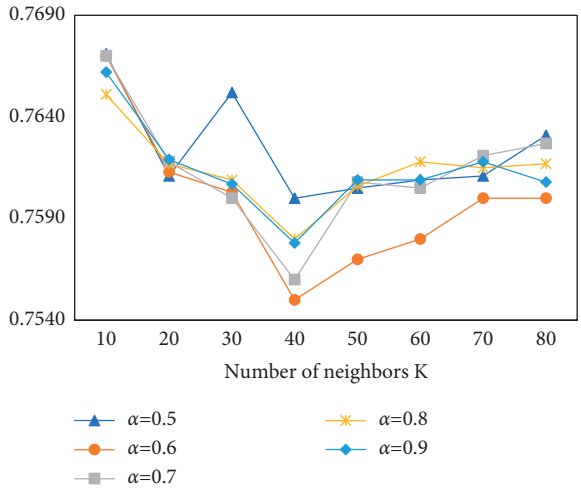

Figure 1: MAE changes with α value.

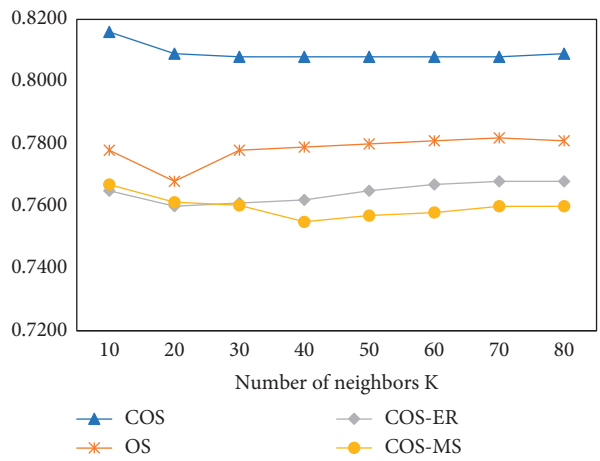

Figure 2: Comparison of MAE values of different algorithms based on Movielens100K dataset.

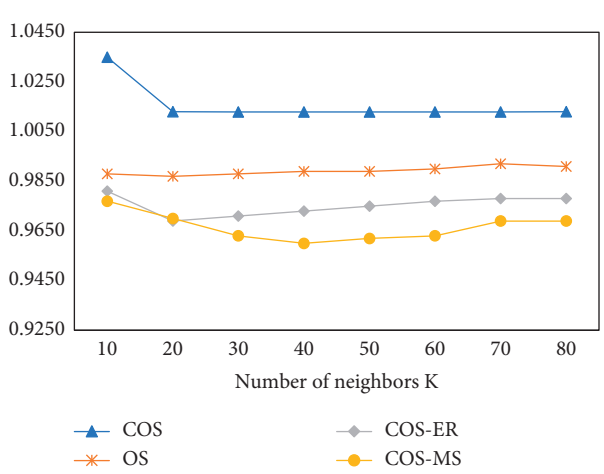

Figure 3: Comparison of RMSE values of different algorithms based on Movielens100K dataset.

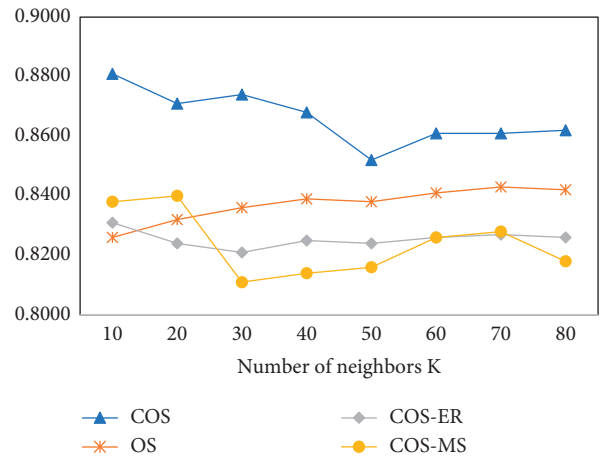

Figure 4: Comparison of MAE values of different algorithms based on Epinions dataset.

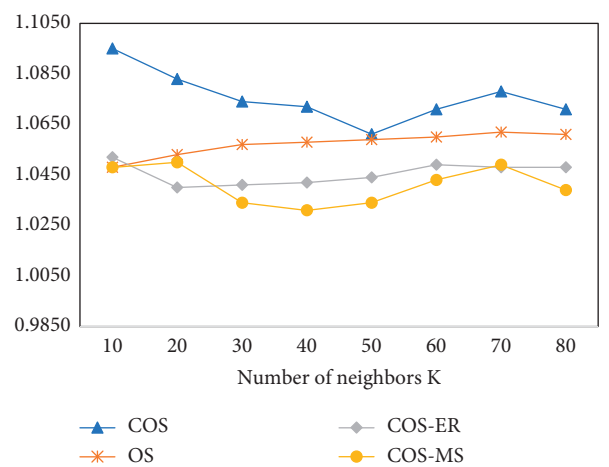

Figure 5: Comparison of RMSE values of different algorithms based on Epinions dataset.

datasets is different, which can better verify the universality of the algorithm.

Movielens100K dataset includes users' ratings of movies. 943 users rated 1682 movies, with a score of 1–5. Each user scored at least 20 movies, with a data density of about 6.3%. In addition, it also includes user attribute information and movie metadata information.

The Epinions dataset includes 664824 rating data (1–5 points) of 139738 items from 40163 users. The trust information between users includes 487181 pieces. The data sparsity is very low, so this paper preprocesses the data according to the method of literature [14]. Finally, the sparsity of the data is 98.22%.

We use Python for programming, and the experiment is carried out based on the Lenskit toolkit [15].The dataset is

Input: User item scoring matrix R, target user V, number of neighbor users K, number of recommended items n
Output: N items recommended to target user V
(1) Import user item scoring matrix data
(2) For the common attention matrix, the attention similarity between users is calculated by Formula (5)
(3) For user attributes, use Fformulas (6) and (7) to calculate the similarity between user attributes
(4) Through the common scoring items, the similarity of user attention is calculated by Formula (8)
(5) The attention similarity calculated in step 2 (the weight is α), the attribute similarity calculated in step 3 (the weight is $1-\alpha$), the weight fusion method is used to weight the score similarity to obtain the user multisimilarity
(6) According to the calculation result in step 5, extract K users' favorite m items ($m > n$, excluding the items already liked by the target user)
(7) According to the popularity, use Formula (10) to calculate the weight W of the improved project popularity
(8) M candidate items are multiplied by the weight W, and N recommended items are selected from high to low

Algorithm 1: Collaborative filtering recommendation algorithm integrating time and multisimilarity.

Table 3: Algorithm time efficiency (based on Movielens100K dataset).

Algorithm	Running time (unit: second)
COS	802.98
OS	1367.66
COS-ER	1211.23
COS-MS	2167.56

divided into training set and test set, with the proportion of 70% and 30%, respectively.

The experimental environment is 3.70 GHz CPU, with 32 GB Storage, windows 11 operating system.

4.2. Evaluation Index. Mean absolute error (MAE) and root mean squared error (RMSE) are the evaluation indicators for evaluating the accuracy of prediction scores, which are applied to calculate the difference between the predicted value and the actual value of the project. The smaller the MAE value (as shown in Formula (11)), where N represents the number of prediction scores) and the smaller RMSE value (as shown in Formula (12)) are, the higher the accuracy of the prediction value is [16]

$$\mathrm{MAE} = \frac{\sum_{u\epsilon U, j\epsilon I}\left|P_{u,i} - R_{u,i}\right|}{N}, \tag{11}$$

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{u\epsilon U, j\epsilon I}\left(P_{u,i} - R_{u,i}\right)^2}{N}}. \tag{12}$$

4.3. Results and Analysis

4.3.1. Determination of Relevant Parameters. Because the data sparsity of Movielens100K dataset is relatively moderate, we adopt them to determine the relevant parameters in order to prevent over fitting. In the algorithm, the proportion of different angle similarity in the calculation is different. Considering in the actual scene, the scoring similarity (i.e. in Formula (5)) can generally reflect the similarity between users better than the attribute similarity (i.e. in Formulas (6)–(8)). The weight of scoring similarity is set to α.The proportion is higher than the weight of attribute similarity $(1-\alpha)$.Therefore, the scoring weight α increases in steps of 0.01, and the value range is [0.5, 0.9].The number of neighbors K is 10, 20, 30, 40, 50, 60, 70, and 80, and the recommended number is 10. It can be found from Figure 1 that the value in the algorithm MAE is between 0.750 and 0.766; when α is at 0.6, the value of MAE is relatively low and stable. Therefore, in this paper, the weight value α is 0.6.

4.3.2. Comparative Analysis

(1) Algorithm Error Rate. For checking the effectiveness of this algorithm, the similarity COS-MS proposed in this paper is compared with COS, COS-ER [7], and OS [1] and other similarity calculation methods based on the above indicators. The number of neighbors K is 10, 20, 30, 40, 50, 60, 70, and 80. The MAE and RMSE values based on Movielens100K dataset are shown in Figures 2 and 3. The values of MAE and RMSE based on the Epinions dataset are shown in Figures 4 and 5.

From Figures 2 and 3, it can be found that the MAE and RMSE values of each algorithm based on Movielens100K dataset change with the change of K value. The algorithm COS-MS proposed in this paper has a certain betterment over COS and OS. When the number of neighbors is less than 30, the COS-ER algorithm is better than COS-MS. But when the number of neighbors gradually increases, the advantages of this algorithm are gradually reflected. This is mainly because this algorithm considers the multiangle similarity. When the number of neighbors is on the increase, the accuracy of relative recommendation will also improve.

Similarly, as can be seen from Figures 4 and 5, the MAE and RMSE values of each algorithm change with the change of K value. Compared with Figures 3 and 4, we can find that the values of MAE and RMSE have increased, indicating that when the data sparsity increases, the error will increase slightly. When the number of neighbors is greater than 30, the algorithm COS-MS proposed in this paper has certain improvement over other algorithm.

For different datasets, when the number of neighbors increases, the algorithm shows good accuracy, in other words, the values of MAE and RMSE decrease.

(2) Algorithm Time Efficiency. Table 3 lists the time required for each algorithm to compute the similarity based on Movielens100K dataset. It can be seen that the more factors considered in calculating the similarity, the higher the time complexity, and the longer the time required for calculation. The COS algorithm has the lowest time complexity, the OS and COS-ER algorithm have the similar time complexity, and the COS-MS algorithm has the highest time complexity and takes the longest time. COS is a relatively simple traditional calculation method, which takes the shortest time. OS and COS-ER have been modified on the basis of COS, and the amount of calculation is similar. Our algorithm combines many aspects of similarity calculation, and the time is relatively high. Therefore, for the calculation of large datasets, it is a requisite to study distributed calculation to enhance the recommendation efficiency. Due to space limitation, the running time based on Epinions dataset will not be described, which is similar to the conclusion shown in Table 3.

5. Concluding Remarks

In view of the existing algorithms that do not fully consider the impact of the environment on users' interest, this paper introduces the time factor to improve the influence of long-term data on recommendation. For the problem of cold start, new users are recommended through user attribute similarity. According to the idea that only the items that users are interested in will produce behavior, the improved algorithm pays more attention to users' attention and introduces the similarity of users' attention so as to reduce the impact of the score itself on the recommendation algorithm. Generally, products with high popularity are easier to be found by users. Different weights are set for different items to reduce the role of popular items in similarity calculation and final recommendation. Through the Movielens100K dataset and Epinions dataset as the experimental data, the experiment shows that this algorithm can effectively improve the accuracy of recommendation and give users a better recommendation effect.

Because the algorithm is slightly insufficient in time efficiency, the project team not only considers using distributed computing to improve efficiency when there is a large amount of data but also needs to optimize the algorithm steps to optimize the algorithm [17, 18]. At the same time, how to effectively model data with knowledge map and how to integrate deep learning technology to solve accurate recommendation under big data are the further works of our project team.

Data Availability

The data that support the findings of this study are available from the author.

Conflicts of Interest

The author declares that there are no conflicts of interest regarding this paper.

Acknowledgments

This work was supported by the National Social Science Foundation (No.16BFX085).

References

[1] A. Gazdar and L. Hidri, "A new similarity measure for collaborative filtering based recommender systems," *Knowledge-Based Systems*, vol. 188, Article ID 105058, 2020.

[2] W. U. Jian-xin and Z. Zhang, "Collaborative filtering recommendation algorithm based on user rating and similarity of explicit and implicit interest," *COMPUTER SCIENCE*, vol. 48, no. 5, pp. 147–154, 2021.

[3] A. H. Nabizadeh, J. P. Leal, H. N. Rafsanjani, and R. R. Shah, "Learning path personalization and recommendation methods: a survey of the state-of-the-art," *Expert Systems with Applications*, vol. 159, Article ID 113596, 2020.

[4] Y. Jin-ming, J. Meng, and W. Qiu-feng, "Item collaborative filtering recommendation algorithm based on improved similarity measurement," *Computer Applications*, vol. 37, no. 5, pp. 1387–1391, 2017.

[5] J. Luo and Z. Wen-qi, "User similarity calculation method considering item similarity weight," *Computer Engineering and Applications*, vol. 51, no. 8, pp. 123–127, 2015.

[6] B. Wang, *Collaborative Filtering Recommendation Algorithm Combining Item Attribute Similarity and Rating Similarity*, Yanshan University, Qinhuangdao China, 2017.

[7] W. Li-ping, F. U. Pan, and Q. yue, "Collaborative filtering algorithm combined with relative differences in scores," *Journal of Chinese Computer Systems*, vol. 43, no. 7, pp. 1388–1393, 2022.

[8] L. I. H. yang and F. U. Y. qing, "Collaborative filtering recommendation algorithm based on tag clustering and item topic," *COMPUTER SCIENCE*, vol. 45, no. 4, pp. 247–251, 2018.

[9] D. Wang, Y. Yih, and M. Ventresca, "Improving neighbor-based collaborative filtering by using a hybrid similarity measurement," *Expert Systems with Applications*, vol. 160, Article ID 113651, 2020.

[10] L. Lü, M. Medo, C. H. Yeung, Y. C. Zhang, Z. K. Zhang, and T. Zhou, "Recommender systems," *Physics Reports*, vol. 519, no. 1, pp. 1–49, 2012.

[11] R. E. V. D. S. Rosa, F. A. S. Guimarães, R. D. S. Mendonça, and V. F. D. Lucena, "Improving prediction accuracy in neighborhood-based collaborative filtering by using local similarity," *IEEE Access*, vol. 8, pp. 142795–142809, 2020.

[12] C. Dong, *Research on Improvement of Hybrid Algorithm Based on Content and Collaborative filtering*, Shanxi University of Finance & Economics, Taiyuan, China, 2021.

[13] L. Deng, J. Huang, and C. Yue, "Collaborative filtering algorithm integrating item popularity and multi-similarity among users," *Journal of Chinese Computer Systems*, vol. 7, 2021.

[14] S. Endle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: bayesian personalized ranking from implicit feedback," in *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pp. 452–461, AUAI Press, Hildesheim, Germany, June 2012.

[15] M. D. Ekstrand, "LensKit for python: Next-Generation Software for Recommender Systems experiments," in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pp. 2999–3006, Ireland, October 2020.

[16] Y. Wang, P. Wang, Z. Liu, and L. Y. Zhang, "A new item similarity based on α-divergence for collaborative filtering in sparse data," *Expert Systems with Applications*, vol. 166, Article ID 114074, 2021.

[17] L. Zhang, Z. Zhang, J. He, and Z. Zhang, "A User-BasedCollaborativeFiltering Recommendation System Basedon TrustMechanismandTimeWeighting," in *Proceedings of the 2019 IEEE25thInternationalConference on Paralleland Distributed Systems (ICPADS)*, pp. 69–76, Tianjin, China, January 2020.

[18] Y. Zhang, Z. Dong, and M. Xiang-Wu, "Research on personalized advertising recommendation system and their applications," *Chinese Journal of computer*, vol. 44, no. 3, pp. 531–563, 2021.