Hindawi
Mobile Information Systems
Volume 2021, Article ID 3209140, 9 pages
https://doi.org/10.1155/2021/3209140

Research Article

A Personalized Recommendation Algorithm Based on Weighted Information Entropy and Particle Swarm Optimization

Shuhao Jiang,[1,2] **Jincheng Ding**,[3] **and Liyi Zhang**[1,2]

[1]*School of Information Engineering, Tianjin University of Commerce, Tianjin 300134, China*
[2]*School of Electrical Automation and Information Engineering, Tianjin University, Tianjin 300072, China*
[3]*School of Science, Tianjin University of Commerce, Tianjin 300134, China*

Correspondence should be addressed to Shuhao Jiang; 920122185@tjcu.edu.cn

Received 31 August 2021; Revised 3 November 2021; Accepted 2 December 2021; Published 24 December 2021

Academic Editor: Fazlullah Khan

Similarity calculation is the most important basic algorithm in collaborative filtering recommendation. It plays an important role in calculating the similarity between users (items), finding nearest neighbors, and predicting scores. However, the existing similarity calculation is affected by over reliance on item scores and data sparsity, resulting in low accuracy of recommendation results. This paper proposes a personalized recommendation algorithm based on information entropy and particle swarm optimization, which takes into account the similarity of users' score and preference characteristics. It uses random particle swarm optimization to optimize their weights to obtain the comprehensive similarity value. Experimental results on public data sets show that the proposed method can effectively improve the accuracy of recommendation results on the premise of ensuring recommendation coverage.

1. Introduction

With the development of mobile Internet technology, people have more and more methods and ways to obtain all kinds of information on the Internet anytime and anywhere and can more conveniently choose the content they want to obtain. However, at the same time, it is also actively or passively submerged in a large amount of data. The content selection process of Internet users becomes cumbersome and complex due to the increase of the amount of data. They cannot quickly obtain information that is really useful to themselves, and the efficiency of using information is reduced. This is the so-called information overload problem. The emergence of personalized recommendation effectively solves this problem. Without too much interaction, users can recommend items or content they are interested in for users to choose. Personalized recommendation is defined as predicting users' preferences by analyzing users, projects, and the information associated with users and projects and then recommending the most suitable projects to specific users. In the recommendation system algorithm, collaborative filtering recommendation algorithm is a more mature personalized recommendation method. The collaborative filtering recommendation algorithm [1,2] is to obtain the k nearest neighbors that are most similar to the user/item through similarity calculation and then use the nearest neighbor information to predict the user's preference for a certain item.

However, as the number of users and items continues to increase, the sparseness of the user-item rating matrix increases, the number of common item ratings is relatively reduced, and the accuracy of the recommendation results decreases. For this reason, the researchers focused on the similarity calculation method [3]. Optimization and improvements have been made to improve the accuracy of the recommended results. On the one hand, it is aimed at the sparseness of the user-item scoring matrix. Sanjeevi et al. [4] proposed a collaborative filtering method based on biclustering (CBE-CF), which uses biclustering to identify dense modules of the scoring matrix and then measures the similarity between new users and dense modules through information entropy. A collaborative filtering algorithm

based on improved weighted slope one is proposed in the literature, which reduces the sparsity of the original scoring matrix and avoids the problem of too single backfill data [5]. Zhang Chen et al. [6] proposed a hybrid collaborative filtering recommendation algorithm. Slope one algorithm is used to fill in the unscored items in the original score matrix and then the SVD technology is used to perform singular value decomposition on the filled score matrix to further modify the predicted score. On the other hand, for traditional similarity calculation methods, it is easy to exaggerate or reduce the actual similarity of users. Wang Yonggui [7] proposed a hybrid recommendation algorithm integrating content and collaborative matrix decomposition technology, effectively alleviating data sparsity and improving recommendation accuracy. SUN et al. [8] obtain comprehensive similarity by fusing Triangle similarity and Jaccard similarity, which alleviates the problem that the number of common items among users affects similarity calculation. Nilashi [9] corrected cosine similarity through Jaccard coefficient weighting, which solved the problem of data sparsity to a certain extent. In order to solve the disadvantages of single score-based method, multi-index collaborative filtering technology has also been widely used [10]. Zhou et al. [11] proposed a modified user similarity calculation method. The user similarity is obtained by modifying the Pearson similarity by the proportion factor of the number of weighted common evaluation items. Wu Senet al. [12] proposed a sparse cosine similarity calculation method, by distinguishing users into high-relevance users and low-relevance users, and further designed differentiated numerical similarity calculation methods for different types of users to alleviate the traditional numerical similarity.

The traditional similarity calculation only calculates the phase degree between users ((items) based on the user's historical score of the project [13]). However, there is a problem that the evaluation factors, such as the behavior characteristics of users and the category attributes of the project, are not comprehensive. In other words, the similarity in collaborative filtering recommendation only calculates the similarity of item scores, while the similarity of user characteristics and item contents (category attributes) is not within the calculation range. Such calculation results can not completely reflect the similarity between users (items), thus affecting the quality of recommendation results. In this paper, a similarity calculation method based on information entropy optimization is proposed, which takes into account the user's score similarity and preference feature similarity and uses the random particle swarm optimization algorithm to optimize their weights to obtain the comprehensive similarity value, which ensures the coverage of the recommendation results and improves the accuracy.

1.1. Personalized Recommendation Algorithm Based on Information Entropy and Particle Swarm Optimization. The technical route of this article is shown in Figure 1.

Aiming at the problems faced by traditional user-based collaborative filtering algorithms, this paper proposes a personalized recommendation method that improves the calculation of user similarity. On the one hand, in view of the sparsity of the user-item rating matrix, this paper uses the SVD++ algorithm to fill the user-item rating matrix and then calculates the user's Pearson similarity to get the user ratings linear similarity $\mathrm{psim}_{u,v}$; on the other hand, for different user rating features, there are differences in the impact of user similarity. The concept of information entropy is introduced to calculate the similarity, and the weighted information entropy is obtained by adding the difference of the common score items d_i between users to the information entropy calculation formula. After normalizing it, use Jaccard similarity correction to get the user rating feature similarity $\mathrm{qsim}_{u,v}$. The particle swarm optimization algorithm is used to solve the weight coefficient in the weighted sum of the two similarities, so as to obtain the comprehensive similarity. Then, the comprehensive similarity is used in the personalized recommendation process to calculate the final prediction accuracy of the model.

1.2. User Feature Similarity Calculation Based on Weighted Information Entropy. In the collaborative filtering algorithm based on neighbors, similarity calculation is the core step of personalized recommendation. However, the traditional similarity calculation method has the following problems. (1) It only reflects the linear correlation between the user vectors. When the scores of two users differ greatly, a higher similarity may be obtained. For example, if the score vectors of user A and user B are (1,2,1,2) and (4,5,4,5), respectively, the result calculated by Pearson similarity between user A and user B is 1, and the actual situation is contrary. (2) The influence of the proportion of the number of shared item ratings between users on the calculation of similarity is not considered. When two users have only a few shared item ratings and this part of the score is similar, a higher degree of similarity will be obtained. For example, when the score vectors of user A and user B are both 100-dimensional, but the score vectors of the common items are only two-dimensional and both are (4, 5). Only the linear similarity $\mathrm{psim}_{u,v}$ is used to calculate the user similarity to 1, which obviously does not meet the actual situation.

Therefore, this paper introduces the concept of information entropy to calculate the method of user similarity. The concept of information entropy [14,15] was proposed by Claude Shannon, the father of information theory, in 1948. Information entropy can measure the degree of confusion of a set of data; that is, the more chaotic the data, the greater the information entropy; the more concentrated the data, the smaller the information entropy. The calculation formula is

$$H(X) = \sum_{i=1}^{n} p(a_i) \log_2 \frac{1}{p(a_i)}. \tag{1}$$

Here, $p(a_i)$ is the probability that the element a_i appears in the sample X, and $H(X)$ is the information entropy of the sample X. Introducing information entropy into the user similarity calculation process of the recommendation system, the calculation formula is

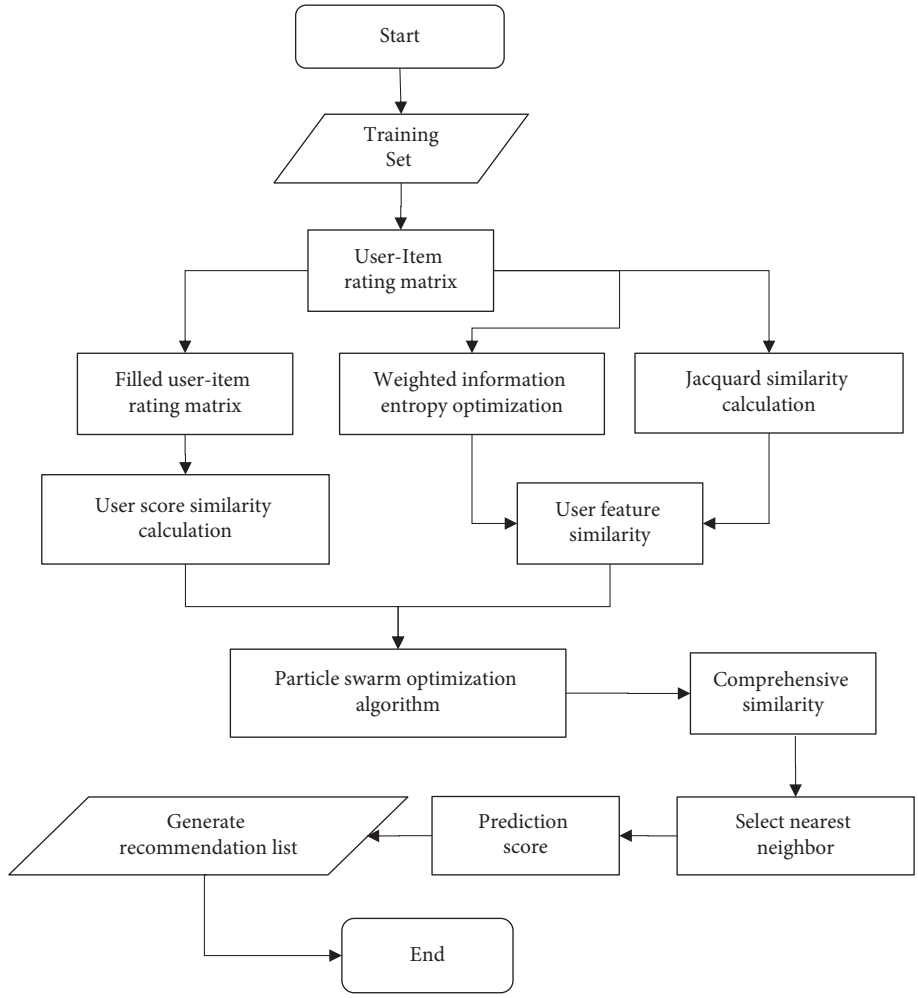

Figure 1: Technical roadmap of this article.

$$H(u,v) = \sum_{i=1}^{n} p(d_i)\log_2 \frac{1}{p(d_i)}. \quad (2)$$

Here, d_i is the difference value of the common item score, and $p(d_i)$ is the probability of the difference value d_i appearing in the common score difference vector. The smaller the information entropy of the calculated score difference between two users, the smaller the change in the score difference between the two users, and the higher the similarity of the corresponding two users. Formula (5) still has certain shortcomings when directly applied to the recommendation process. For example, the common score difference vectors of user A and user B and user A and user C are (4,3,3,4) and (1,2,1,2), the similarity between user A and user B and user A and user C are calculated by formula (5), which is obviously not in line with the actual situation.

Therefore, while introducing information entropy, considering that the difference in the scores of each shared item also affects the similarity of the two users; that is, the greater the difference in the scores of the shared item, the greater the degree of difference between the two users; in addition, consider the influence of the number of ratings of two users' shared items on user similarity, a weight coefficient $1/n$ is introduced in the information entropy formula, where n is the number of ratings of two users' shared items. Therefore, the calculation formula of weighted information entropy [16,17] is

$$H(u,v) = -\frac{1}{n}\sum_{i=1}^{n} p(d_i)\log_2 \frac{1}{p(d_i)}\left|d_i\right|. \quad (3)$$

Since $H(u,v)$ is greater than or equal to 0, it cannot be directly used as the similarity between users. It needs to be normalized to the interval [0,1]. This paper refers to the literature and proposes a normalization method for $H(u,v)$; the calculation formula is

$$\mathrm{Sim}_H(u,v) = \frac{1}{1+H(u,v)}. \quad (4)$$

Here, the larger the value of $H(u,v)$, the smaller the similarity between the two users; the smaller the value of $H(u,v)$, the greater the similarity between the two users. Other normalization methods are as follows in literature [19] and literature [20], but in the actual application of the recommendation system process, the normalization function established in this paper has a small prediction error.

When formula (4) is applied to the user similarity calculation of the recommendation system, it is found that the above calculation formula still has certain shortcomings; that is, only the number of common item ratings of two users is considered, but the number of common item ratings between users is not considered. This will have a negative impact on similarity calculation, so this article further revises the similarity calculated from the information entropy of the weighted score difference and introduces the Jaccard similarity as the correction factor. The calculation formula of the correction factor is

$$JS(u,v) = \frac{|I_u \cap I_v|}{|I_u \cup I_v|}. \tag{5}$$

Here, the numerator represents the number of common item ratings of user u and user v, I_u represents the number of items evaluated by user u, and I_v represents the number of items evaluated by user v.

The correction factor for the proportion of the total score is added to the similarity calculated by the weighted score difference information entropy, and the similarity calculation formula is

$$q\text{sim}_{u,v} = \text{sim}_H(u,v) \times Js(u,v). \tag{6}$$

1.3. User Score Similarity Calculation. The SVD++ algorithm [18–20] is an improved algorithm based on the SVD (singular value decomposition) algorithm. SVD++ is based on the SVD algorithm considering explicit feedback and adding implicit feedback information to predict the user's unrated items. As far as movie recommendation is concerned, hidden factors can be factors such as the type of movie and production cost. The explicit preference is obtained by analyzing the user's rating of the movie. The implicit preference is obtained by analyzing the user's historical viewing records and other behaviors. SVD++ combines explicit feedback and implicit feedback to make scoring predictions for unrated items. The popular understanding of SVD++ is as follows: if a user has rated a movie, it means that the user has watched the movie. This user behavior contains certain preference information of the user, and this information is reflected in the form of implicit parameters. In the model, a more refined model is thus obtained. The formula for predicting scoring is

$$\hat{r}_{ui} = \mu + b_i + b_u + q_i^T\left(p_u + |I_u|^{-1/2}\sum_{j\in I_u} y_j\right). \tag{7}$$

Here, μ is the global average of the scores of all records in the training set. b_i is a project bias item. b_u is a user bias item. p and q are the two matrices after dimensionality reduction, q_i are the ith column of matrix q, and p_u are the uth row of matrix p, the matrix p is the user hidden factor matrix, and the matrix q is the hidden factor item matrix. I_u is the collection of all the movies that the user has evaluated, and y_j is the user's personal preference bias items hidden in "movies evaluated j."

In order to obtain the effectiveness of the prediction score, the loss function is defined as

$$Q = \min \sum_{(u,i)\in K}\left(r_{ui} - \hat{r}_{ui}\right)^2 + \lambda\left\{\sum_u\left(b_u^2 + \|p_u\|^2\right) + \sum_i\left(b_i^2 + \|q_i\|^2 + \|y_i\|^2\right)\right\}. \tag{8}$$

Here, $\sum_{(u,i)\in K}(r_{ui} - \hat{r}_{ui})^2$ is the square term of the difference between the predicted score and the actual score, $\lambda\left\{\sum_u(b_u^2 + \|p_u\|^2) + \sum_i(b_i^2 + \|q_i\|^2 + \|y_i\|^2)\right\}$ is the regularization item, and λ is the regularization coefficient. The gradient descent method is used to calculate the model parameters b_i, b_u, q_i, p_u, y_j under which the loss function Q is minimized.

Pearson correlation coefficient is used to calculate the score similarity value between user u and user v. Its calculation formula is shown in the following formula:

$$\text{psim}_{u,v} = \frac{\sum_{u,v\in U}\left(R_{u,i} - \overline{R}_u\right)\left(R_{v,i} - \overline{R}_v\right)}{\sqrt{\sum_{u\in U}\left(R_{u,i} - \overline{R}_u\right)^2}\sqrt{\sum_{v\in U}\left(R_{v,i} - \overline{R}_v\right)^2}}. \tag{9}$$

Here, users $u, v \in U$; U is a collection of users u and users of the userv; $R_{u,i}$ indicates the user's score on the item i; $\overline{R}_u$ is the average value of all over rated items by users u; $R_{v,i}$ indicates score on the item i by userv; and $\overline{R}_v$ is the average value of all over rated items by user v.

1.4. Comprehensive Similarity. This paper considers user similarity from two aspects. On the one hand, the direction of user rating linear similarity $\text{psim}_{u,v}$ is the linear correlation between user rating vectors; on the other hand, the direction of user rating feature similarity $\text{qsim}_{u,v}$ is the actual user rating vector. For deviation and the percentage of the total number of user ratings, comprehensively consider the similarities obtained from the two aspects and obtain the comprehensive similarity through weighted fusion. The comprehensive similarity calculation formula is

$$\text{Sim}_{u,v} = \lambda q\text{sim}_{u,v} + (1-\lambda)q\text{sim}_{u,v}. \tag{10}$$

1.5. Particle Swarm Optimization. Particle swarm optimization [21] is a representative algorithm of swarm intelligence optimization algorithm, which is a kind of evolutionary algorithm. The particle swarm optimization algorithm [22] is designed by simulating the behavior of birds looking for food. The basic idea is to use the sharing of information by individuals in the group so that the movement of the entire group produces an update process from disorder to order in the problem-solving space, so as to obtain the optimal solution of the problem. The main process is firstly to generate random initial particles. Each particle is a possible solution of the objective function. The particles seek the optimal solution by iterating to make the

objective function the best. In each iteration, the particles compare two extreme values that are updated iteratively. These two extreme values are the local optimal solution *pbest* obtained by the particle itself and the global optimal solution currently *gbest* obtained by the entire particle population. The formulas for particle update speed and position are as follows.

The speed conversion formula is

$$v_{i+1} = w * v_i + c_1 * \text{rand}_1 * (p\text{best}_i - x_i) + c_2 * \text{rand}_2 * (g\text{best}_i - x_i). \quad (11)$$

The position transformation formula is

$$x_i = x_i + v_{i+1}. \quad (12)$$

Here, w is the inertia factor, generally taken as 1. c_1 and c_2 are learning factors, the meaning of c_1 is the individual learning factor of each particle, and the meaning of c_2 is the social learning factor of each particle. $rand_1$ and $rand_2$ are random numbers within (0, 1).

This paper uses the particle swarm optimization algorithm to solve the logic steps of the weight α of similarity: set the weight α as the particle, and the constraint condition of the particle is $0 < \alpha < 1$. Initialize n particles, get different comprehensive similarity from n particles, and then synthesize similarity. In the personalized recommendation process, the root mean square error RMSE (root mean squared error) is calculated, and the root mean square error RMSE is used as the objective function of the particles. The particles obtain the global optimal solution that minimizes the root mean square error RMSE through the flow chart steps of the particle swarm optimization algorithm.

1.6. Nearest Neighbor Selection and Prediction Score. According to the comprehensive similarity, select the nearest neighbor of the user, calculate the prediction score of the item J to be recommended by the target user u with the following formula, and select the top k items with the highest prediction score to recommend to the user. The calculation formula is

$$r_{uj} = \overline{r_u} + \frac{\sum_{v \in N_u} \text{Sim}_{u,v} \times \left(r_{vj} - \overline{r_v}\right)}{\sum_{v \in N_u} \text{Sim}_{u,v}}, \quad (13)$$

where r_{uj} represents the prediction score of user u on item j; $\overline{r_u}$ is user u's average score of all items; N_u represents the nearest neighbor set of user u; $\text{Sim}_{u,v}$ represents the comprehensive similarity between user u and user v; and $\overline{r_v}$ represents the average score of users v.

2. Experiment

2.1. Data Set Introduction. The data set selected in this article is the Movielens-100K data set, which is collected and published by the Grouplens team and is a commonly used data set for testing the effect of collaborative filtering algorithms. The data set is 10,000 rating data generated by 943 users on 1,682 movies, and the rating range is (1,5). In this experiment, 80% of the data is randomly selected as the training set, and the remaining 20% as the test set.

2.2. Evaluation Index. After training on the training set data, the user's prediction value for the movie is obtained, and the prediction result is compared with the actual test set score data to measure the quality of the prediction. In this paper, the average absolute error MAE (mean absolute error) value and the root mean square error RMSE value are used as evaluation indicators [23], and the specific calculation formula is

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{n} \left(\hat{r}_{ui} - r_{ui}\right)^2}{n}}, \quad (14)$$

$$\text{MAE} = \frac{\sum_{i=1}^{n} \left|\hat{r}_{ui} - r\right|_{ui}}{n}. \quad (15)$$

Here, n represents the number of user ratings in the test set, $\hat{r}_{ui}$ represents the predicted score of user u on item i, and r_{ui} represents the actual score of user u on item i.

The recommended coverage is used as another evaluation index. Recommendation coverage rate refers to the proportion of items recommended by the recommendation system to all items, and its calculation formula is

$$COV\,(\text{Length}) = \frac{N_d\,(\text{Length})}{N}, \quad (16)$$

where $N_d\,(\text{Length})$ represents the number of different items that the recommendation system can recommend to users when the length of the recommendation list isLength and N represents the number of all items.

2.3. Analysis of Results

2.3.1. Number of Hidden Factors of SVD++ Algorithm. The influence of the number of hidden factors on the SVD++ algorithm is reflected in the fact that when the number of hidden factors is too small, the SVD++ decomposition will lose a large amount of information of the original matrix, and the prediction effect will be worse when filling; when the number of hidden factors is too large, the SVD++ decomposition will lead to excessive decomposition of the original matrix information, and the purpose of dimensionality reduction is not achieved, which leads to overfitting during filling, which makes the prediction effect worse. Therefore, the design experiment explores the influence of the number of hidden factors in the SVD++ model on the accuracy of prediction. The design parameters [24,25] are as follows: the number of iterations is 30, the regularization coefficient is 0.15, the initial learning rate is 0.04, and the number of iterations is 0.93. The learning rate decreases. The experimental results are drawn as a broken line graph, as shown in Table 1.

In Table 1, the abscissa is the number of hidden factors, and the ordinate is RMSE. It can be seen from the figure that

Table 1: The value of RMSE under different hidden factors.

The number of hidden factors	10	20	30	40	50	60	70
RMSE	0.9161	0.9128	0.9114	**0.9105**	0.9109	0.9113	0.9120

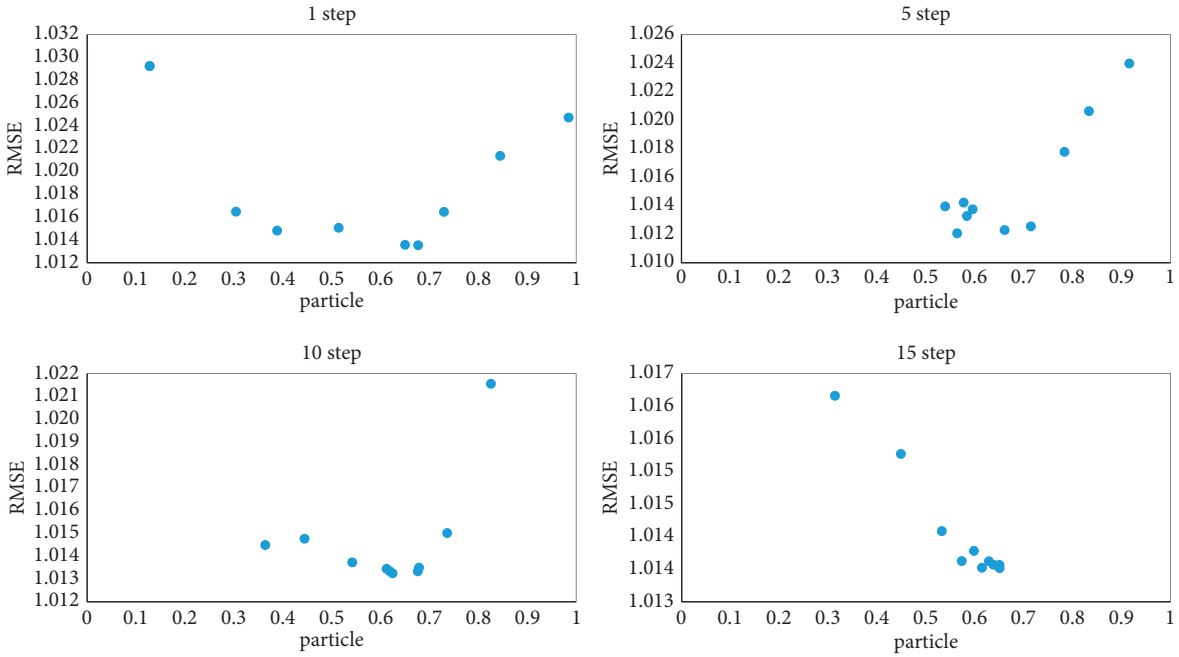

Figure 2: State diagram of the particle iteration process.

under other conditions and the number of hidden factors changes, when the number of hidden factors $N = 40$, the RMSE value reaches the minimum. Therefore, in the next experiment, the number of hidden factors in the SVD++ model is set to 40.

2.3.2. Fusion Coefficient of Comprehensive Similarity. Design experiments to explore the impact of the fusion coefficient α on the accuracy of prediction. The parameters of the particle swarm optimization algorithm are set as follows. The maximum number of iterations is 15, the number of particles is 10, the range of generated particles is [0,1], and the learning factors c_1 and c_2 are set to 2. The inertia factor w is 0.8. The comprehensive similarity is weighted by $\text{psim}_{u,v}$ and $\text{qsim}_{u,v}$, and the size of the fusion coefficient represents the proportion of the two similarities. The larger the value of λ, the larger the proportion of similarity calculated by the modified weighted information entropy; the smaller the value of λ, the larger the proportion of Pearson similarity calculated after SVD++ is filled. To show the particle iteration process, select the number of iterations as the first, fifth, tenth, and fifteenth iterations to draw the following particle state diagrams.

As shown in Figure 2, from the results of the first iteration, the initialized 10 particle points are uniformly distributed in the interval [0,1]. Judging from the results of the fifth iteration, the particles begin to initially converge, concentrating on the interval [0.5, 0.7]. From the results of the 10th iteration, the particles are looking for the best points around the interval [0.5, 0.7] on the basis of the previous iteration. At the 15th iteration, the particles basically converged, and most of the particles were concentrated in the range of [0.5, 0.7]. After 15 iterations, the particle swarm optimization algorithm searched for the optimal solution $\alpha = 0.5616$. The RMSE value obtained at this time reaches a minimum of 1.01158546.

In order to verify the effectiveness of the particle swarm to solve the fusion weights λ, in the interval [0,1], 9 points including the beginning and the end are taken as the value of the fusion weight λ with an interval of 0.1, and the RMSE and MAE values under different weights λ are calculated and plotted into the following figure.

The abscissas of the left and right graphs in Figure 3 are the weight coefficients α, which are taken in the interval [0,1], and the ordinates are the RMSE and MAE values, respectively. The line graph in the figure shows the weight coefficient α in the interval [0,1] and its corresponding RMSE value and MAE value drawn at intervals of 0.1. The scattered points connected by the dotted line in the interval [0.5, 0.6] are the particle swarms. The calculated weight coefficient and its corresponding RMSE value and MAE value are plotted. It can be seen from the values of the discrete points on the left of Figure 3 that when the weight coefficient is 0.7, the RMSE value is the smallest. However, when the MAE value is the smallest under the same conditions, the weight coefficient should be 0.6. Combining with the better results obtained by the particle swarm, it can be

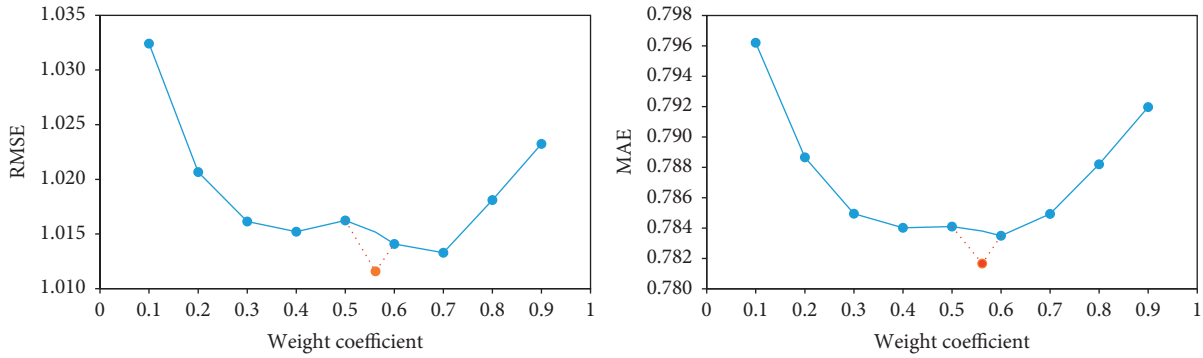

Figure 3: Weight coefficient under discrete value corresponding RMSE value and MAE value.

seen that in the method of taking discrete points to find the better value of the weight coefficient, the optimal position of the particle may be missed, which makes the prediction accuracy of the recommendation system not optimal. The experimental results verify the validity of the weight coefficients obtained by the particle swarm, and it can be seen that the optimal parameters obtained by the particle swarm are more specific and accurate than those obtained by discrete points.

2.3.3. Coverage Analysis of Experimental Results. The recommendation coverage COV (Length) can be used as an indicator to measure the recommendation ability of the algorithm for trailing items. If the items in the recommendation list of a recommendation system cannot reach most of all items, that is, when the recommendation coverage is low, the recommendation system is likely to reduce the user's satisfaction with the recommendation system due to the limitations of the scope of recommended items. Therefore, on the basis of measuring the prediction accuracy of the algorithm and considering the impact of the recommendation algorithm on the coverage of the recommendation list, the design parameter: the number of nearest neighbors is 30. Draw the coverage curve of each algorithm under the same number of nearest neighbors, as shown in Figure 4.

The abscissa in the figure is the length of the recommendation list, and the ordinate is the coverage. As can be seen from the figure, when the number of nearest neighbors is 30, with the expansion of the recommendation list, the proportion of items in the recommendation list covering the item set also increases. Compared with the based SVD++ algorithm, the recommended coverage of the algorithm proposed in this paper is greatly improved, and the recommended coverage of the other algorithms is relatively close, which also fully shows that the optimization of similarity calculation plays a positive role in improving the recommended coverage.

2.4. Accuracy Analysis of Experimental Results. After determining the parameters of the algorithm, comparison with other collaborative filtering models was performed under different number of nearest neighbors in order to verify the

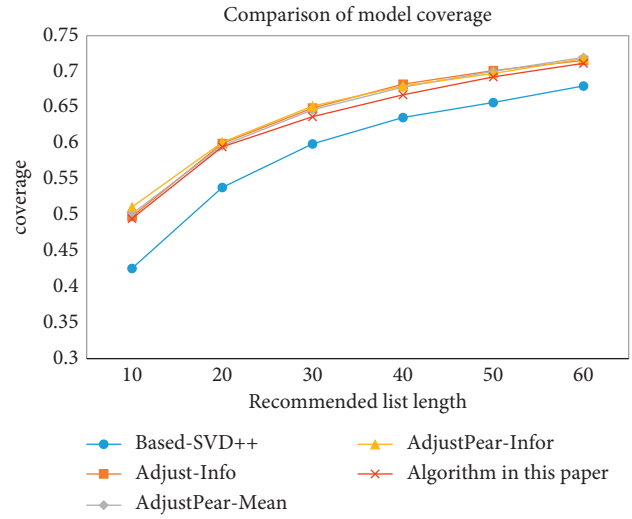

Figure 4: The COV (Length) value of each model under different nearest neighbors.

effectiveness of the algorithm proposed in this paper. Comparison models include collaborative filtering (Based-SVD++) using Pearson similarity based on SVD++ filling matrix, collaborative filtering (Adjust-Info) using modified weighted information entropy as user similarity, and literature [26] penalizes users. The similarity obtained by the coefficient correction Pearson similarity is fused with the similarity obtained by the rating information entropy, and collaborative filtering (AdjustPear-Infor) is performed. Literature [27] adds the Pearson similarity to the common item score ratio correction factor and average score. The correction factor obtains the comprehensive similarity for collaborative filtering (AdjustPear-Mean). Draw the above model to calculate the model accuracy index RMSE value under different number of nearest neighbors, and draw it as Figure 5.

In Figure 5, the horizontal axis is the number of nearest neighbors, and the vertical axis is the RMSE value. From the analysis of the comparative experimental results of each algorithm, it can be seen that the Based-SVD++ algorithm solves the sparseness problem of the user-item rating matrix when calculating the user similarity but does not consider the impact of the shared rating items between users on the

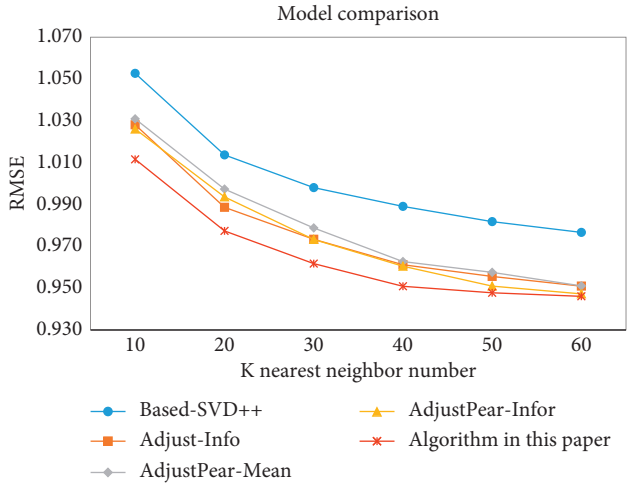

Figure 5: The RMSE value of each model under different nearest neighbors.

similarity, and the overall similarity obtained is very high. When choosing the nearest neighbor, it is difficult to choose the best nearest neighbor. Although the AdjustPear-Mean optimization algorithm uses a correction factor to optimize the Pearson similarity, it does not consider the problem that the Pearson similarity cannot correctly measure the impact of the difference in ratings between users on the similarity. The AdjustPear-Infor optimization algorithm introduces the concept of information entropy in user differences and innovates in the normalization formula, but it still does not consider the impact of the proportion of shared scoring items among users on similarity.

The optimization algorithm in this paper is compared with the AdjustPear-Mean and AdjustPear-Infor optimization algorithms in the literature. The optimization algorithm in this paper comprehensively considers the linear similarity of user ratings and the similarity of user ratings features and proposes one in user rating feature similarity. A modified weighted information entropy is calculated, and the two similarities measure two aspects of user similarity so that a better comprehensive similarity can be obtained after weighted fusion. The experimental results show that this paper proposes a personalized recommendation method to optimize user similarity, which can effectively improve the accuracy of the recommendation system on the premise of ensuring the recommendation coverage.

3. Conclusion

This paper proposes a personalized recommendation algorithm based on weighted information entropy and particle swarm optimization, which comprehensively considers the similarity of user characteristics and score. From the simulation results, the proposed method can effectively improve the accuracy of recommendation results and has a positive impact on recommendation coverage.

With the in-depth development of recommendation system research, more and more theories and methods of swarm intelligence algorithm and machine learning are applied to the field of personalized recommendation. In this paper, particle swarm optimization algorithm is used to optimize the weight factor of comprehensive diversity, and some results are achieved. Therefore, the next research direction is how to deeply combine swarm intelligence algorithm and machine learning with personalized recommendation to improve the quality of recommendation results.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] A. Dubey and R. Ranjan, "A collaborative filtering movie recommendation system based on the plain bayesian algorithm[J]," *Journal of Research in Science and Engineering*, vol. 3, no. 4, 2021.

[2] W. Zhao, Y. Zeng, and Y. He, "Collaborative filtering via factorized neural networks," *Applied Soft Computing*, vol. 109, Article ID 107484, 2021.

[3] K. Deng, *Research on the Similarity of Collaborative Filtering Recommendation System Based on Neighborhood*, Jiangxi University of Finance and Economics, Jiangxi, China, 2017.

[4] S. G. Sanjeevi, S. Grandhe, and H. Shrivastav, "An adaptive clustering and incremental learning solution to cold start problem in recommendation systems," in *Proceedings of the International Scientific Academy of Engineering and Technology (ISAET) Conference Proceeding*, Dubai, UAE, 2015.

[5] J. Wang, G. Tan, and W. Zhang, "Collaborative filtering algorithm integrating improved weighted slope one [J]," *Microelectronics & Computer*, vol. 37, no. 04, pp. 37–42, 2020.

[6] C. Zhang, J. Xiao, and L. Zhou, "Research on hybrid collaborative filtering recommendation algorithm based on matrix filling[J]," *Mathematics in Practice and Knowledge*, vol. 51, no. 10, pp. 81–89, 2021.

[7] C. Y. WangYonggui, "Hybrid recommendation algorithm integrating content and matrix decomposition," *Computer Application Research*, vol. 37, no. 5, pp. 1359–1363, 2020.

[8] S.-B. Sun, Z.-H. Zhang, X.-L. Dong et al., "Integrating Triangle and jaccard similarities for recommendation," *PLoS One*, vol. 12, no. 8, Article ID e0183570, 2017.

[9] M. Nilashi, O. b. Ibrahim, and N. Ithnin, "Multi-criteria collaborative filtering with high accuracy using higher order singular value decomposition and Neuro-Fuzzy system," *Knowledge-Based Systems*, vol. 60, pp. 82–101, 2014.

[10] B. Smyth, *Case-Based Recommendation the Adaptive Web*, pp. 342–376, Springer, Berlin, Germany, 2007.

[11] W. Zhou, R. Li, and W. Liu, "Collaborative filtering recommendation algorithm based on improved similarity," in *Proceedings of the IEEE 5th Information Technology and Mechatronics Engineering Conference (ITOEC)*, pp. 321–324, Chongqing, China, June 2020.

[12] S. Wu, Y. Dong, W. Guiying, and X. Gao, "Research on user similarity calculation of collaborative filtering for sparse data [J/OL]," *Computer Science and Exploration*, vol. 12, 2021.

[13] P. H. Moghadam, "An exponential similarity measure for collaborative filtering," *SN Applied Sciences*, vol. 1, no. 10, pp. 1–4, 2019.
[14] H. He, Y. Hong, W. Liu, and S.-A. Kim, "Data mining model for multimedia financial time series using information entropy," *Journal of Intelligent and Fuzzy Systems*, vol. 39, no. 4, pp. 5339–5345, 2020.
[15] S. Wang, *Research on Collaborative Filtering Algorithm Based on Fuzzy Theory and Information Entropy [D]*, Northwest University, Kirkland, Washington D C., USA, 2020.
[16] P. Xia, *Research on Collaborative Filtering Algorithm in Personalized Recommendation Technology [D]*, Ocean University of China, Qingdao, China, 2011.
[17] X. Shi and Q. Lei, "Collaborative filtering algorithm based on information entropy and user interest temporality," *Journal of Donghua University*, vol. 45, no. 4, pp. 555–558+570, 2019.
[18] S. Wang, G. Sun, and Y. Li, "SVD++ recommendation algorithm based on backtracking," *Information*, vol. 11, no. 7, 2020.
[19] W. Shi, L. Wang, and J. Qin, "User embedding for rating prediction in SVD++-Based collaborative filtering[J]," *Symmetry*, vol. 12, no. 1, 2020.
[20] Li Rui and F. Feng, "Research on movie recommendation system based on clustering and SVD++[J]," *Computer Age*, vol. 9, pp. 88–90+94, 2020.
[21] X. Bai, X. He, C. Sun, and G. Zhang, "Improved PSO algorithm based on hierarchical learning to solve complex optimization problems[J]," *Journal of Taiyuan University of Science and Technology*, vol. 42, no. 3, pp. 169–174, 2021.
[22] T. Tong Xin, "Stability bounds and almost sure convergence of improved particle swarm optimization methods[J]," *Research in the Mathematical Sciences*, vol. 8, no. 2, 2021.
[23] Y. Shi, "Overview of personalized recommendation algorithms," *Intelligent Computers and Applications*, vol. 10, no. 8, pp. 110–112, 2020.
[24] ZhangZhenpeng, Y. Wang, F. Renyan, and Q. Li, "Hyperparameters of SVD++ recommendation algorithm," *Journal of Guizhou University (Natural Science Edition)*, vol. 35, no. 3, pp. 97–100, 2018.
[25] Y. Wang, F. Li, X. Zhang, and Y. Tian, "An efficient SVD++ algorithm to improve learning rate," *Modern Electronic Technology*, vol. 41, no. 3, pp. 146–150+156, 2018.
[26] J. Zhang and G. Li, "A fusion collaborative filtering algorithm based on rating information entropy," *Journal of Nanjing University of Posts and Telecommunications (Natural Science Edition)*, vol. 2, pp. 76–81, 2021.
[27] Li Rong and G. W. LiMingqi, "Research on collaborative filtering algorithm based on improved similarity[J]," *Computer Science*, vol. 43, no. 12, pp. 206–208+240, 2016.