Hindawi
Scientific Programming
Volume 2017, Article ID 4379141, 9 pages
https://doi.org/10.1155/2017/4379141

Research Article

A Novel Hybrid Similarity Calculation Model

Xiaoping Fan,[1,2] Zhijie Chen,[1] Liangkun Zhu,[3] Zhifang Liao,[3] and Bencai Fu[3]

[1] *School of Information Science and Engineering, Central South University, Hunan, China*
[2] *Hunan University of Finance and Economics, Hunan, China*
[3] *School of Software, Central South University, Hunan, China*

Correspondence should be addressed to Zhifang Liao; zfliao@csu.edu.cn

Received 25 August 2017; Accepted 12 November 2017; Published 4 December 2017

Academic Editor: Longxiang Gao

This paper addresses the problems of similarity calculation in the traditional recommendation algorithms of nearest neighbor collaborative filtering, especially the failure in describing dynamic user preference. Proceeding from the perspective of solving the problem of user interest drift, a new hybrid similarity calculation model is proposed in this paper. This model consists of two parts, on the one hand the model uses the function fitting to describe users' rating behaviors and their rating preferences, and on the other hand it employs the Random Forest algorithm to take user attribute features into account. Furthermore, the paper combines the two parts to build a new hybrid similarity calculation model for user recommendation. Experimental results show that, for data sets of different size, the model's prediction precision is higher than the traditional recommendation algorithms.

1. Introduction

Traditional collaborative filtering (CF) algorithms usually calculate similarity between users or items based on user-item rating matrix, and in the light of the calculated similarity they choose the nearest neighbor and construct prediction scores to generate recommendation lists. Therefore, the similarity calculation decides the precision and quality of recommendations produced by the heuristic CF algorithm. However, the present traditional heuristic CF recommendation algorithms suffer from a range of problems in similarity calculation, such as the failure in finding changes of user interest; that is, by directly computing similarity on the basis of statistics, it considers user ratings and center ratings only while ignoring other factors when rating, such as user attributes, time weight, and user rating habits.

In order to solve the problems of similarity calculation in traditional heuristic CF recommendation and improve its performance, Luo et al. [1], Anand and Bharadwaj [2], and Lopes et al. [3] proposed the global similarity measure. Based on the traditional similarity algorithms, the global similarity measure takes the transitive relationships among users into account to calculate the global similarity and build the user's nearest neighborhood set. Results of Lopes' experiments indicated that, in case of the extremely sparse data set, the combination of traditional similarity algorithms and the global similarity measure can improve the accuracy of recommendation. Li et al. [4] proposed the concept of fluctuation factor. He considered the influence of fluctuation factors and removed the influence of them by z-score method when computing the similarity between users. Shen et al. [5] proposed a two-stage similarity learning algorithm, in which at the first stage it utilizes PCC to calculate the similarity and obtains the nearest neighbor, and at the second stage it uses the reduced gradient method to learn the similarity, which improves the recommendation accuracy. Gao and Huang [6] proposed the idea based on the model of item gravity attribute. Its similarity calculation contains two parts: one of which is the similarity obtained by the traditional calculation; the other part firstly defines the weight value of the item attribute, and then the initial similarity is calculated by the model of item gravity attribute, and, after the two similarities are weighted, the effect of the rating time is taken into account to calculate the final similarity value.

Starting from different perspectives, the studies above aimed at strengthening the association between users and items to improve the similarity between users or items and get the optimal nearest neighbor set, finally improving the recommendation accuracy and quality on this basis. However, when strengthening the association between users and items, we can take some factors into account, such as the demographic characteristics of users and the time decay caused by the time-effect of ratings, which have certain effects on the association. It is very effective to consider user attribute features when dealing with the problem of user's cold start.

Therefore, the paper proposes a new similarity calculation method: RIT-UA algorithm. The RIT-UA algorithm consists of two parts: one is the similarities of user rating-interest, which considers the similarities of user rating and interest as well as the changes and effects of the two under the constraints of rating time and confidence coefficient between users; the other part is the similarities of the user attributes, which takes into account the influence of the user attribute feature on the recommendation and calculates the similarity of the user attributes after getting the weight of each attribute feature. In the end, RIT-UA algorithm fits the two parts linearly. The experimental results show that, compared with the traditional methods, the algorithm proposed in this paper can obtain better prediction accuracy.

2. Related Work

In studies of recommendation system, though in recent years the recommender systems have been studied frequently and developed sufficiently, there are still some common problems, such as data sparsity, cold start, and user interest drift. In order to deal with these problems and improve the recommendation precision and accuracy, researchers may take many aspects into account, including the basic user attribute feature and the time and place where the user behavior occurred, and researches about these came into being correspondingly.

Demographic Recommender System (DRS) is an important part of recommender systems. Demographic characteristics can be used to identify the user's type and their preferences, and the system can sort users according to their attribute features and generate recommendations based on the sorting results. DRS plays a great supporting role in dealing with the problems of user cold start and data sparsity. Many of the present studies have proved that user attribute features can improve the accuracy in recommendations. Luo et al. [7] used improved quantized kernel least mean square (EQ-KLMS) algorithm, which improved the efficiency of machine learning and improved the accuracy of weather forecast. Beel et al. [8] elaborate the role of user attribute features in the recommended process and analyze and demonstrate that the user's attribute characteristics have a significant impact on click-through rates on recommender systems. From the perspective of tourism recommendation, Wang et al.'s [9] experiments proved that the combination of machine learning method (Naive Bayes, Bayesian network, and SVM) and demographic characteristics can improve the prediction accuracy of tourism recommendations. Zhao et al. [10] used visual tracking sensors to acquire biometric information and then used machine learning based biometrics to improve the accuracy of recognition. Combined with user attribute features, Al-Shamri [11] constructed five similarity measures, respectively, based on user preference modeling method, and the experimental results showed that the combination of user attribute features improves the recommendation accuracy of recommender systems. Santos et al. [12] applied user attribute features in real recommendation environment to mine and analyze the context constraints in the scene. Chen and He [13] constructed the user demographic vector by the user information, and, on this basis, took the corated item and the item's frequency into account to figure out a new similarity. The experimental results showed that this approach can solve the problem of cold start effectively and improve the recommendation accuracy. Luo et al. [14] achieve QoS prediction with automatic parameter tuning capability by using approximate dynamic programming, through online learning and optimization, without the need for preknowledge or prediction model identification. Then, through the use of a kernel least mean square algorithm [15], the lack of Web services QoS is forecasting. The experimental results show that the method can effectively solve the cold start problem and improve the prediction accuracy.

With the intensive development of recommender systems research, in order to obtain better recommendations and improve recommendation quality, many researchers began to incorporate contextual information into the research of the recommender systems. Relatively speaking, the time information is easier to collect among contextual information, and it provides significant value for researches on improving the diversity of timing sequence of the recommender systems, which has become a hot topic in the current studies [16]. Koren [17] used matrix factorization (SVD), which regards time as an important feature and add it to the feature data set of user-item, and solved the problem of user interest drift effectively. Karatzoglou et al. [18] and Xiong et al.[19] regarded the time information as the third eigenvector, employing the approach of tensor factorization to show the dynamic changes of time. According to the user's rating history, Rong et al. [20] divided it into several periods and analyzed the user's preference distribution in each period and quantified their preferences. Li et al. [21] split user preferences to stages over time and proposed the cross-domain CF framework. The experiments proved that the algorithm not only improves the recommendation prediction accuracy but also solves the problem of user interest drift.

3. Description of RIT-UA Algorithm

In the context of relatively sparse data, from the perspective of solving the problem of user interest drift, this paper proposes the RIT-UA algorithm on the basis of the traditional similarity calculation, with the introduction of factors (such as the user attribute characteristics and time decay of rating) which influence user's rating behaviors. The RIT-UA algorithm consists of two parts: one is the similarities of rating-interest, and the other part is the similarities of the user attributes.

3.1. The Similarities of Rating-Interest. The similarities of rating-interest are composed of rating similarity and interest similarity, mainly considering two aspects: users' preference for items and user's rating habits. Meanwhile, based on the

Table 1: User-item rating matrix.

	Item 1	Item 2	Item 3	Item 4
User 1	4	5	-	1
User 2	-	2	3	2
User 3	1	-	3	4
User 4	-	2	-	3

two aspects, the effect of time decay of rating is introduced and the confidence coefficient between users is also introduced with the combination of the fluctuation factor proposed in literature [4]. In the end, the similarities of rating-interest between users are obtained. The whole process is described as follows.

3.1.1. Rating Similarity. In the field of e-commerce systems, Rating or Voting is generally used to obtain the user's direct preference for items. Assuming that the degree of user's preference for items is classified as 5 levels, which is {adore, love, like, dissatisfied, and dislike}, and the corresponding grades are {5, 4, 3, 2, 1}. Consequently, the results will produce a rating matrix. The rating preference matrix of user-item can be shown in Table 1.

Table 1 is a rating matrix of user-item. In the rating matrix, when the ratings of two users are closer, it indicates that their preferences are similar. When the ratings of two users are the same, it implies that the users share the same preferences. If there is big gap between ratings, then it means that the two users have opposite preferences. Therefore, in order to describe the nonlinear correlation of the similarity between users' ratings, the paper constructs sigmoid function to express the similarity of user ratings based on the literature [22], in which sigmoid function is put forward as the expression of the similarity. In this paper, the sigmoid function is also used to represent the similarity between users. The equation is shown below:

$$\operatorname{sim}(u,v,i)_{\text{rate}} = 2\cdot\left(1-\frac{1}{1+\exp\left(-\left|r_{ui}-r_{vi}\right|\right)}\right). \quad (1)$$

Equation (1) represents the similarity between the ratings for item i from user u and user v. r_{ui} represents the ratings for item i from user u and r_{vi} represents the ratings for item i from user v.

3.1.2. Interest Similarity. Every user has their own rating habits. For instance, some users who do not stick to rifles always tend to give a high score, while some rigorous users who pay much attention to details is likely to give a low score. Because they are more strict with the score, they do not give high scores easily. Hence, the description of user habits is helpful to improve the prediction accuracy. For the user rating habits and the inherent attributes of the item, Koren [17] used an equation to define them, as shown in equation (2), that is, regarding user's own rating habits as a factor having an impact

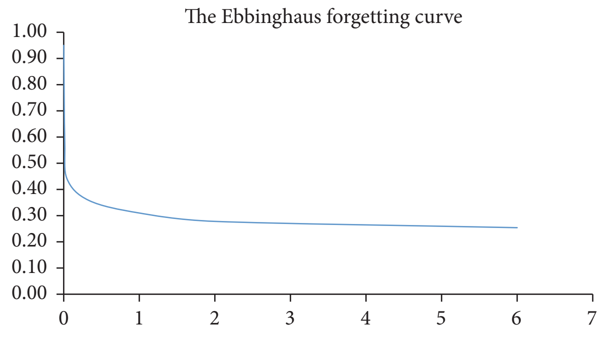


Figure 1: Changes of Ebbinghaus forgetting curve.

on rating. In (2) b_u stands for the user's own rating habits, while b_i stands for the user's rating for item i.

$$b_{ui} = \mu + b_u + b_i. \quad (2)$$

Therefore, within the range of rating for items, when a user tends to score highly and likes an object, he/she usually gives a high score for it. However, even though the user does not like the object, he/she will not give a low score and vice versa. Therefore, according to the average score given by the user for an item, his/her interest and preference of rating habits can be showed. Similarly, based on literature [22], which proposed sigmoid function as the expression of the similarity, the paper also constructs sigmoid function to express the similarity of user interests, shown in

$$\operatorname{sim}(u,v,i)_{\text{interest}} = 1 - \frac{1}{1+\exp\left(-\left(r_{ui}-\bar{r}_u\right)\left(r_{vi}-\bar{r}_v\right)\right)}. \quad (3)$$

Equation (3) represents the similarity of the interest of user u and user v on item i. Then, combining the rating similarity and interest similarity between users, we get a computational equation, shown as

$$\operatorname{sim}(u,v)'_{\text{score}} = \operatorname{sim}(u,v,i)_{\text{rate}} + \operatorname{sim}(u,v,i)_{\text{interest}}. \quad (4)$$

3.1.3. Time Factor. Generally speaking, treating user behaviors that occurred at various time equally leads to the shortage of effective quantitative analysis. Time factor shows the degree of changing tendency of user interest drift. The closer the rating information to the present time, the better recommendation effects it has and vice versa. Based on this, some studies used linear and nonlinear functions to quantify the rating behaviors over time.

In the literature [23], in order to solve the difficult problem of tracking the changes of user interest, the Ebbinghaus forgetting curve is put forward for the research of user interest fitting. Changes of Ebbinghaus forgetting curve is shown in Figure 1. Based on the literature [23], combined with the trend of Ebbinghaus forgetting curve this paper uses the following function to describe the trend of user interest drift, that is, draw the impact direction of the time factor, as shown in

$$f(\Delta t) = \frac{2e^{-\alpha\Delta t}}{1+e^{-\alpha\Delta t}} \in (0,1]. \quad (5)$$

Δt represents the time difference between users' rating on item I, which is the parameter, and in this paper, we set it as 0.005. After taking time-effect into account, therefore, the new computational equation for similarities of user rating-interest arrives:

$$\operatorname{sim}(u,v)''_{\text{score}} = \frac{1}{\left|I_{uv}\right|}\sum_{i\in I_{uv}}\left(\operatorname{sim}(u,v)'_{\text{score}}\cdot f(\Delta t)\right). \quad (6)$$

$|I_{uv}|$ represents the number of items corated by user u and user v.

3.1.4. Confidence between Users. When the user data is extremely sparse and the number of corated items is very small, there is a large fortuitous factor in the similarity calculation. Li et al. [4] eliminate this effect by using the fluctuation factor. Based on this, the paper introduces the number of corated items to adjust the weight of similarity through nature exponential, shown as

$$\operatorname{confident}(u,v) = \exp\left(\frac{\left|I_u \cap I_v\right|}{\max\left(\left|I_u \cap I_w\right|\right)} - 1\right). \quad (7)$$

Equation (7) represents the confidence coefficient between user u and user v, stands for the item rated by user u, stands for the item rated by user v, shows the corated items of user u and user v, represents the corated item between user u and the nearest neighbor, and stands for the nearest neighbor set.

After taking confidence coefficient into account, the adjusted equation to calculate the similarity of user rating-interest arrives:

$$\operatorname{sim}(u,v)_{\text{score}} = \operatorname{sim}(u,v)''_{\text{score}}\cdot\operatorname{confident}(u,v). \quad (8)$$

3.2. The Similarity of User Attributes. Considering the similarity of user attributes, on the one hand it can improve the accuracy of prediction, and on the other hand it can solve the problem of new user's cold start; that is, when there is no other available rating data, data of user attribute features can be used to build models and give recommendations. As for the description about the similarity of user attributes, literature [20] divided the user attributes into numerical attributes and name attributes and defined and expressed them, respectively. From the perspective of being easy to understand and implement, this paper defines the similarity of user attributes as follows.

For single user attribute, it is expressed as $\operatorname{sim}(u,v,i)_{\text{attr}} = 1/0$.

It indicates that when user u and user v share the same attribute i, the value is 1; otherwise the value is 0.

$$\operatorname{sim}_{\text{attr}}(u,v) = \sum_{i\in\text{Attr}} w\cdot\operatorname{sim}(u,v,i)_{\text{attr}}. \quad (9)$$

In (9) w is the value of feature weight of user attribute i. In order to obtain all weight values of each feature attribute, this paper chooses the feature selection algorithm of Random Forest to calculate the importance degree of each user attributes feature and generates a rank of it. Then we conduct experiments according to the rank and acquire the relative importance weight value of each attribute further.

```
Algorithm 1 RIT-UA similarity calculation
Input:
   Testset
Algorithm
(1) For user in Testset do:
(2)    For item in Testset[user] do:
(3)       //get co-rated items
(4)       Users: getCorateditemsUserset(item)
(5)       //get the similarity between user and Users
(6)       calculateRituaSimilarity(Users, user)
(7)       //According similarity select neighbors
(8)       getTonKNeighbors(K)
(9)       //calculate predicted rating
(10)      rating: getRating(Neighbors)
(11)   end for
(12) end for
```

Algorithm 1: Description of RIT-UA algorithm.

3.3. Similarity Calculation Based on RIT-UA. Sections 3.1 and 3.2 consider the similarity of rating-interest and the similarity of user attributes, respectively; hence we carry out weighted combination for the two and get a new computational equation of similarity:

$$\operatorname{sim}(u,v) = \alpha\cdot\operatorname{sim}_{\text{score}}(u,v) + \beta\cdot\operatorname{sim}_{\text{attr}}(u,v). \quad (10)$$

In (10), $\beta = 1-\alpha$. After the computational equation of similarity is obtained, we get the prediction equation of user to item, shown as

$$\widehat{r}_{ui} = \overline{r}_u + \frac{\sum_{v\in U}\operatorname{sim}(u,v)\cdot\left(r_{vi}-\overline{r}_v\right)}{\sum_{v\in U}\left|\operatorname{sim}(u,v)\right|}. \quad (11)$$

$\overline{r}_u$ and $\overline{r}_v$ mean the average scores of user u and user v, respectively, and U stands for the neighbor set of users u.

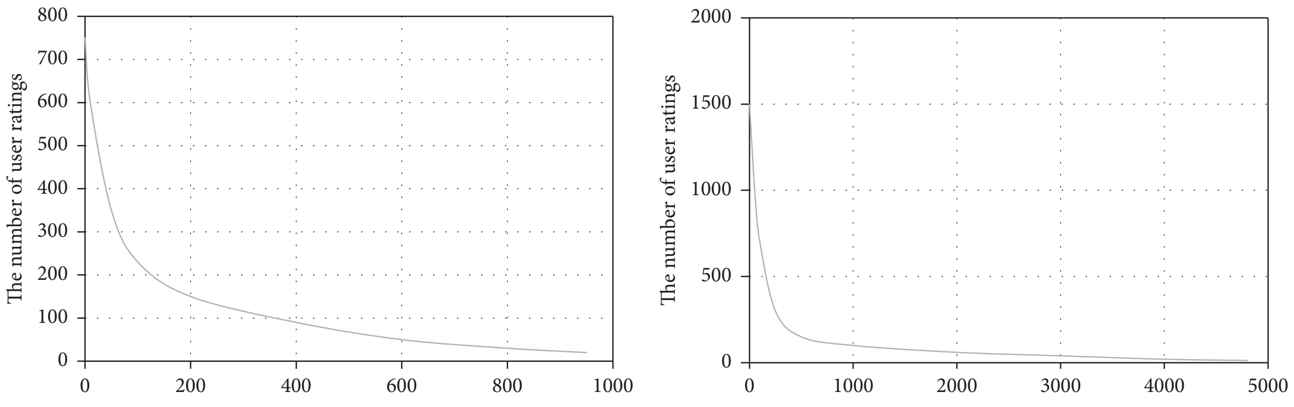
The description of RIT-UA similarity algorithm is in Algorithm 1.

Therefore, from the description in Algorithm 1 we can see that the time complexity of operating RIT-UA algorithm is $O(m * n)$, where m means the number of users and n means the number of items.

4. Description of RIT-UA Algorithm

4.1. Experimental Data Sets. Taking into account the openness and authority of data sets, at the same time, our simulation experiment is based on the scoring matrix, so we chose two data sets, namely, Movielens-100k and Netflix, to carry out experimental analysis and comparison. The process is shown as follows.

4.1.1. Movielens-100k Data Set. The data set is a film rating data set provided by the GroupLens Research. The data set contains 100,000 ratings from 943 users for 1682 movies, where each user has rated 20 movies at least, and the rating interval is {1–5} which is shown as Table 2. Meanwhile, the sparseness of the data set is $1-100000/(943*1682) = 93.7\%$.



(a) ML-100K data set (b) Netflix data set

Figure 2: Changing tendency of the number of user-rated items (descending order).

Table 2: User-item rating matrix.

	Movielens-100k	Netflix
Users	943	4861
Items	1682	5080
Ratings	100000	387939
Rating scale	$\{1, 2, 3, 4, 5\}$	$\{1, 2, 3, 4, 5\}$
Sparseness of data	93.7%	98.4%

Figure 2(a) shows the number of items rated by users on the ML-100k data set in a descending order. From the figure, we can see that the number of items rated by many users is less than 100. In order to test the performance of the algorithm, the data set is divided into two parts: 80% as the training set and 20% as the test set.

In ML-100k data set, there are only 4 attributes about users' attribute feature: gender, age, occupation, and zip code.

4.1.2. Netflix Data Set. Netflix data set is a section of the original Netflix Game data. After the proper data cleaning, the data set contains 387,939 ratings from 4861 users for 5080 objects, where each user has rated 20 objects at least, and the rating interval is {1–5} which is shown as Table 2.

The sparseness of the data set is $1-387939/(4861*5080) = 98.4\%$, and Figure 2(b) shows the number of items rated by users on the ML-100k data set in a descending order. From the figure, we can see that the number of items rated by a large number of users is less than 100. Similarly, in order to test the performance of the algorithm, the data set is divided into two parts: 80% as the training set and 20% as the test set.

In the process of cleaning the Netflix data set, since there is no user attribute feature data in it, according to the features of the user attribute data of ML-100k, this paper randomly generates data of three user attributes in Netflix through the simulation experiment: gender, age, and occupation. The range of age attribute is {10–65}, the occupation attribute has 20 occupations with the range {0–19}, and the value of gender is given within the range {0–2}. Because of the high sparsity of our data sets, we use resource scheduling and processing methods for sparse data [24, 25].

4.2. Experiment Evaluation Quantity. Generally speaking, there are evaluation quantities such as MAE (mean absolute error) and RMSE (root mean squared error) in the experimental evaluation about prediction precision in recommender systems. After comparison, RMSE (root mean squared error) is used as the evaluation quantity in this paper. The equation is

$$\text{RMSE} = \sqrt{\frac{\sum_{i\in\text{Test}} \left(r_{ui} - \widehat{r}_{ui}\right)}{\left|N_{\text{Test}}\right|}}. \qquad (12)$$

$|N_{\text{Test}}|$ represents the size of test data set and refers to the real rating value while referring to the predicted rating value. The smaller the value shown by RMSE, the higher the predicted precision; that is, the smaller the value, the closer the prediction.

4.3. Experimental Process and Analysis

4.3.1. Experiment 1: Experimental Analysis of the Weight Value of User Attribute Feature. From (9) we can see that, in order to obtain the weighted value of each user attribute feature, the Random Forest algorithm is chosen in this paper.

Random forests are an ensemble learning method that can analyze the complicated interactive feature data, even under the influence of certain data noise it is very robust, and it is very efficient in feature learning and analysis. Its variable importance measure can be a feature selection tool for high dimensional data. In recent years, it has been widely used in various kinds of prediction, feature selection, and outlier detection [26].

Therefore, we obtain the weight value of each user attribute feature with Random Forest algorithm on ML-100k and Netflix data set. The experimental results are shown in Figures 3 and 4.

On ML-100k data set, from Figure 3 we can see that among the 4 attributes (age, gender, occupation, and zip code) gender is the most important, indicating that gender attribute exerts a significant role in recommendation and the user rating is more similar when it relates to this attribute. Compared to the gender attribute, zip code attribute exerts

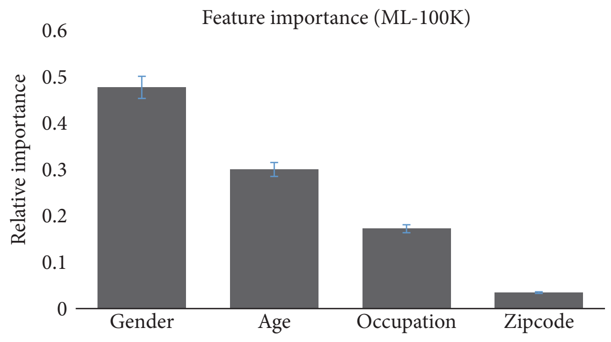


FIGURE 3: Ranking of the weight value of user attributes feature (ML-100k).

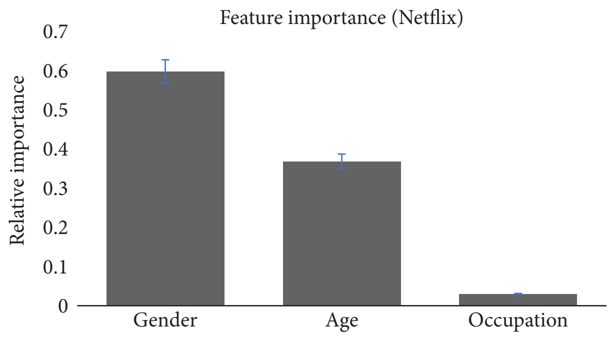


FIGURE 4: Ranking of the weight value of user attributes feature (Netflix).

a relatively low role in recommendation so its weight value is low correspondingly. But the other two attributes age and occupation show relatively medium influence of feature weight, as the experiment implies whose weight value is about 0.284 and 0.186, respectively.

The illustration parts of Figures 3 and 4 show the domain of walker of possible weight values for each feature. For the Netflix data set, the gender and age attributes have very obvious effects in recommendation, and the overall importance rank of the weight value is similar to ML-100k.

In order to test the relative optimal weight values of every and each attribute of (age, gender, occupation, and zip code) and (age, gender, and occupation) on the ML-100k and Netflix data sets, we carry out several sets of comparative experiments in this paper, and experimental results are shown in Figures 5 and 6. From Figures 5 and 6 we can see that on ML-100k data set when "age, gender, occupation, and zip code" are given the values "0.3, 0.3, 0.25, and 0.15," respectively, we get better experimental results. As for Netflix data set, when "age, gender, and occupation" are given the values "0.5, 0.4, and 0.1," respectively, we get better experimental results. Therefore, the values above will be used in the following experiments.

4.3.2. Experiment 2: Experimental Analysis of the Weight Value of Alpha and Beta. According to (10), in order to get the values of α and β which will generate relatively good experimental results, we used the RIT-UA algorithm to carry

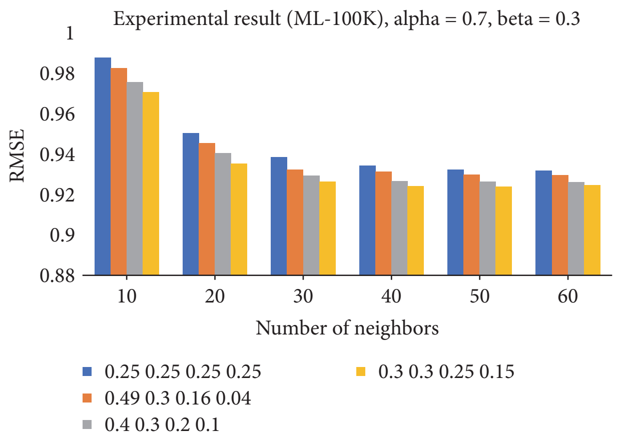


FIGURE 5: Experiment comparison of weight values of different user attributes (ML-100k).

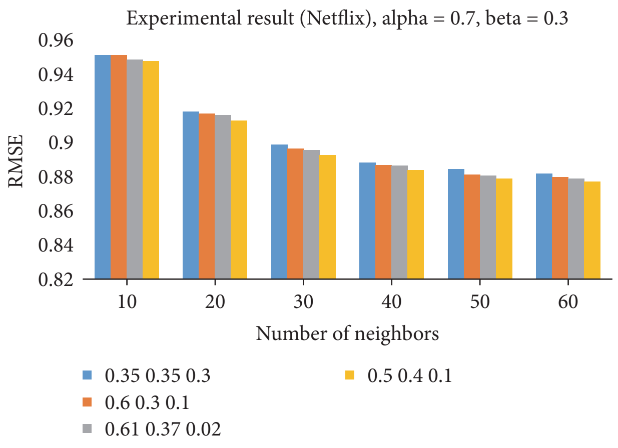


FIGURE 6: Experiment comparison of weight values of different user attributes (Netflix).

out the following groups of experiments based on ML-100k and Netflix data sets. The experimental results are shown in Figures 7 and 8.

From results shown by Figures 7 and 8 we can see that, for ML-100k data set, when $\alpha = 0.75$ and $\beta = 0.25$, we get relatively better experimental results. And, for Netflix data set, when $\alpha = 0.7$ and $\beta = 0.3$, the relatively better experimental results are obtained.

4.3.3. Experiment 3: Experimental Analysis of the Comparison with Other Similarity Measures. In order to verify the validity of the algorithm proposed in this paper, we compare it with other similarity measures, including the Pearson similarity, the adjusted cosine similarity (Acosine), the PIP [27] similarity, and the NHSM [28] similarity on the ML-100k and Netflix data sets. The experimental results are shown in Figures 9 and 10, respectively.

From Figure 9 we know that, on ML-100k data set, the overall experimental results show that, with the increase of neighbors, the algorithm of this paper outperforms others

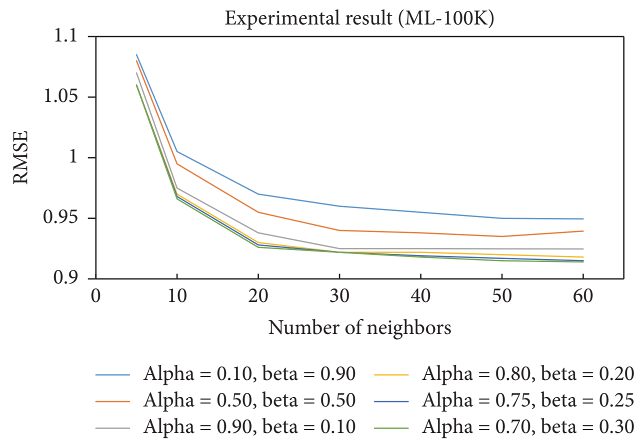


Figure 7: Experimental results with different alpha and beta (ML-100k).

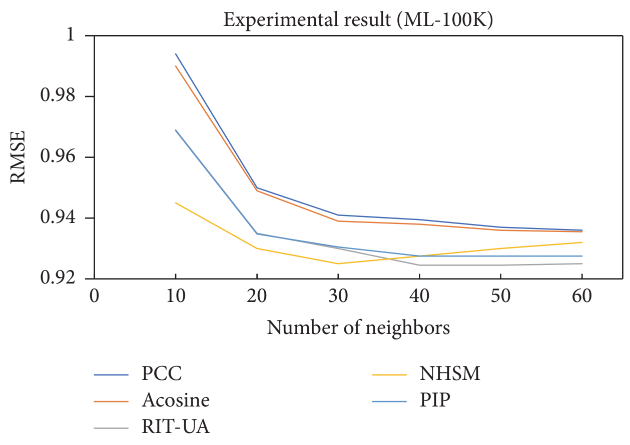


Figure 9: Experimental comparison of different similarity algorithms (ML-100k).

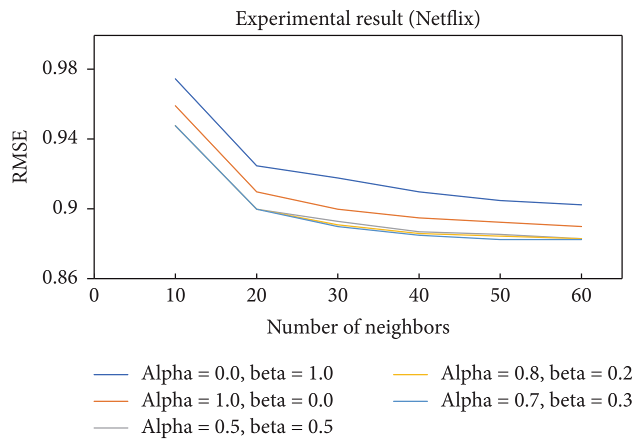


Figure 8: Experimental results with different alpha and beta (Netflix).

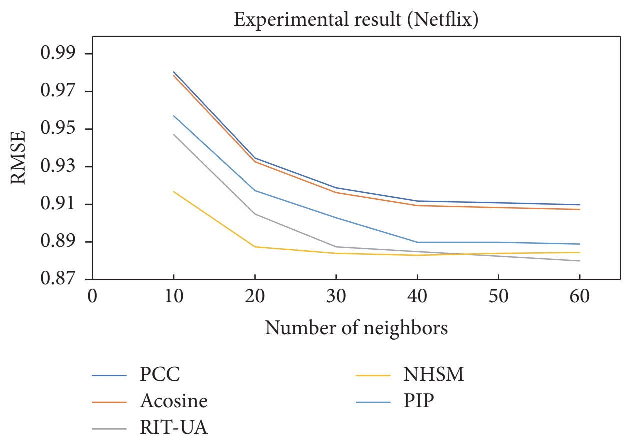


Figure 10: Experimental comparison of different similarity algorithms (Netflix).

gradually. At the beginning stage when the number of neighbors is within [10, 30], the results of the algorithm proposed by this paper are close to those of PIP but slightly better than that of PIP in later period. The experimental results of NHSM are good when the number of neighbors is within [10, 30] but worse in later period. The experimental results of PCC and Acosine are worse than other algorithms. On Netflix data set, from Figure 10 we know that the algorithm proposed in this paper outperforms other algorithms gradually with the increase of neighbors. NHSM outperforms others when the number of neighbors is within [10, 40] but performs not so well as the algorithm proposed by this paper later.

4.3.4. Experiment 4: Comparison of Precision on Data Sets of Different Sizes. Based on ML-100k data set, the paper chooses 20%, 40%, 60%, and 80% of the data set, respectively. Neighbors $k = 20$ as a prerequisite, and we verify the comparison of precision of different algorithms on data set of various sizes. Fivefold cross validation is used to get the average value of experimental results, which is shown as Figure 11.

From Figure 11 we can see that the proposed algorithm produces better and stable results on varied sizes of ML-100k data sets, indicating that, in the case of sparse data, the proposed algorithm has higher identification degree. As for the other three algorithms, performance of PIP algorithm is relatively stable, and RMSE value is relatively low. However, for NHSM algorithm, RMSE value is higher when the data set is relatively sparse, while, with the sizes of the data set increase, the NHSM algorithm performs better and becomes more stable.

5. Conclusion

Aiming at some problems in traditional similarity calculation, this paper proposes a new similarity calculation model. The model describes and expresses aspects such as user rating preference, user rating habits, and time factor. Furthermore, user attributes feature is taken into account for its influence on user ratings, and the role of each attribute feature played in recommendation is studied. Then Random Forests algorithm is used to calculate the weight value of each attribute. The final

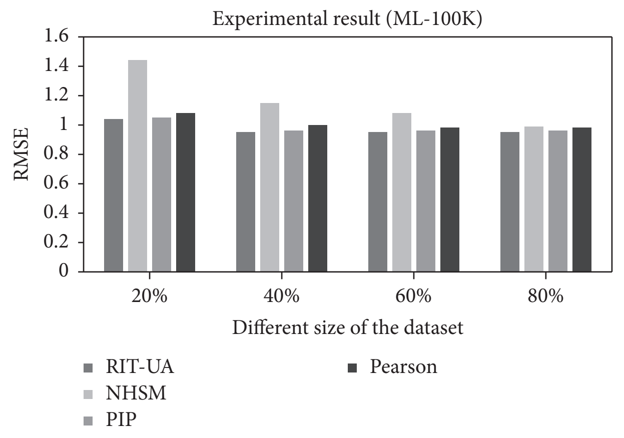


Figure 11: Comparison of results produced by different algorithms on data sets of different sizes (ML-100k).

experimental results show that, compared to other similarity measures, the approach proposed in this paper improves the recommendation precision significantly, and even in the case of sparse data it still shows better experimental results. The deficiency of experiments is that since the user attribute data is relatively small in data set, there is no obvious difference when calculating the feature weight value of user attributes, as the part of user attributes data is private and not easy to obtain, which inevitably cast a shadow on the experiments.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This project is supported by the Fundamental Research Funds for the Central Universities of Central South University with no. 2017zzts623 and Hunan Provincial 2011 Collaborative Innovation Center for Development and Utilization of the Financial and Economic Big Data Property.

References

[1] H. Luo, C. Niu, R. Shen, and C. Ullrich, "A collaborative filtering framework based on both local user similarity and global user similarity," *Machine Learning*, vol. 72, no. 3, pp. 231–245, 2008.

[2] D. Anand and K. K. Bharadwaj, "Utilizing various sparsity measures for enhancing accuracy of collaborative recommender systems based on local and global similarities," *Expert Systems with Applications*, vol. 38, no. 5, pp. 5101–5109, 2011.

[3] A. R. S. Lopes, R. B. C. Prudencio, and B. L. D. Bezerra, "A collaborative filtering framework based on local and global similarities with similarity tie-breaking criteria," in *Proceedings of the International Joint Conference on Neural Networks*, pp. 2887–2893, July 2014.

[4] H. Li, G. Wang, and M. Gao, "A novel similarity calculation for collaborative filtering," in *Proceedings of the International Conference on Wavelet Analysis and Pattern Recognition*, pp. 38–43, IEEE, July 2013.

[5] J. Shen, Y. Wei, and Y. Yang, "Collaborative filtering recommendation algorithm based on two stages of similarity learning and its optimization," in *Proceedings of the 13th IFAC Symposium on Large Scale Complex Systems: Theory and Applications*, pp. 335–340, July 2013.

[6] L. Gao and M. Huang, "A collaborative filtering recommendation algorithm with time adjusting based on attribute center of gravity model," in *Proceedings of the 12th Web Information System and Application Conference*, pp. 197–200, September 2015.

[7] X. Luo, J. Deng, J. Liu, W. Wang, X. Ban, and J. Wang, "A quantized kernel least mean square scheme with entropy-guided learning for intelligent data analysis," *China Communications*, vol. 14, no. 7, pp. 1–10, 2017.

[8] J. Beel, S. Langer, A. Nürnberger et al., "The impact of demographics (age and gender) and other user-characteristics on evaluating recommender systems," in *Research and Advanced Technology for Digital Libraries*, pp. 396–400, Springer, Berlin, Germany, 2013.

[9] Y. Wang, S. C.-F. Chan, and G. Ngai, "Applicability of demographic recommender system to tourist attractions: a case study on trip advisor," in *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology Workshops*, pp. 97–101, December 2012.

[10] W. Zhao, R. Lun, C. Gordon et al., "A human-centered activity tracking system: toward a healthier workplace," *IEEE Transactions on Human-Machine Systems*, vol. 47, no. 3, pp. 343–355, 2017.

[11] M. Y. Al-Shamri, "User profiling approaches for demographic recommender systems," *Knowledge-Based Systems*, vol. 100, pp. 175–187, 2016.

[12] E. B. Santos, M. Garcia Manzato, and R. Goularte, "Evaluating the impact of demographic data on a hybrid recommender model," *IADIS International Journal on WWW/Internet*, vol. 12, no. 2, pp. 149–167, 2014.

[13] T. Chen and L. He, "Collaborative filtering based on demographic attribute vector," in *Proceedings of the International Conference on Future Computer and Communication*, pp. 225–229, IEEE Computer Society, June 2009.

[14] X. Luo, H. Luo, and X. Chang, "Online optimization of collaborative web service QoS prediction based on approximate dynamic programming," in *Proceedings of the International Conference on Identification, Information and Knowledge in the Internet of Things (IIKI '14)*, pp. 80–83, IEEE, Beijing, China, October 2014.

[15] X. Luo, J. Liu, D. D. Zhang, and X. Chang, "A large-scale web QoS prediction scheme for the Industrial Internet of Things based on a kernel machine learning algorithm," *Computer Networks*, vol. 101, pp. 81–89, 2016.

[16] L. Y. Dou and X. H. Wang, "A collaborative filtering recommendation algorithm based on the context of time and tags," *Journal of Taiyuan University of Technology*, no. 6, 2015.

[17] Y. Koren, "Collaborative filtering with temporal dynamics," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '09)*, pp. 447–456, Paris, France, June 2009.

[18] A. Karatzoglou, X. Amatriain, L. Baltrunas, and N. Oliver, "Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering," in *Proceedings of the 4th ACM Recommender Systems Conference (RecSys '10)*, pp. 79–86, ACM, Barcelona,Spain, September 2010.

[19] L. Xiong, X. Chen, T. K. Huang et al., "Temporal collaborative filtering with bayesian probabilistic tensor factorization," in *Proceedings of the Siam International Conference on Data Mining (SDM '10)*, pp. 211–222, Columbus, Ohio, USA, April-May 2010.

[20] H. G. Rong, S. X. Huo, C. H. Hu et al., "User similarity-based collaborative filtering recommendation algorithm," *Journal on Communications*, vol. 35, no. 2, pp. 16–24, 2014.

[21] B. Li, X. Zhu, R. Li et al., "Cross-domain collaborative filtering over time," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI '11)*, pp. 2293–2298, Barcelona, Spain, July 2011.

[22] M. Jamali and M. Ester, "TrustWalker: a random walk model for combining trust-based and item-based recommendation," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '09)*, pp. 397–405, July 2009.

[23] H. Yu and Z. Y. Li, "A collaborative filtering recommendation algorithm based on forgetting curve," *Journal of Nanjing University (Natural Sciences)*, vol. 46, no. 5, pp. 520–527, 2010.

[24] W. Wei, X. Fan, H. Song, X. Fan, and J. Yang, "Imperfect information dynamic stackelberg game based resource allocation using hidden markov for cloud computing," *IEEE Transactions on Services Computing*, 2016.

[25] T. Li, Y. Liu, L. Gao, and A. Liu, "A cooperative-based model for smart-sensing tasks in fog computing," *IEEE Access*, vol. 5, pp. 21296–21311, 2017.

[26] D.-J. Yao, J. Yang, and X.-J. Zhan, "Feature selection algorithm based on random forest," *Journal of Jilin University (Engineering and Technology Edition)*, vol. 44, no. 1, pp. 137–141, 2014.

[27] H. J. Ahn, "A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem," *Information Sciences*, vol. 178, no. 1, pp. 37–51, 2008.

[28] H. Liu, Z. Hu, A. Mian, H. Tian, and X. Zhu, "A new user similarity model to improve the accuracy of collaborative filtering," *Knowledge-Based Systems*, vol. 56, pp. 156–166, 2014.