Association for Information Systems
AIS Electronic Library (AISeL)

WHICEB 2017 Proceedings | Wuhan International Conference on e-Business

Summer 5-26-2017

Collaborative Filtering Similarity Algorithm Using Common Items

Qian Wang
Business School, Sun Yat-Sen University, GuangZhou, 512056, China, mnswq@mail.sysu.edu.cn

Taoqun Zhang
Business School, Sun Yat-Sen University, GuangZhou, 512056, China, 502982914@qq.com

Zhe Rong
Business School, Sun Yat-Sen University, GuangZhou, 512056, China, rongzhe@mail2.sysu.edu.cn

Collaborative Filtering Similarity Algorithm Using Common Items

*Qian Wang, Taoqun Zhang, Zhe Rong**
Business School, Sun Yat-Sen University, GuangZhou, 512056, China

Abstract: Collaborative filtering (CF) plays an important role in reducing information overload by providing personalized services. CF is widely applied, but common items are not taken account in the similarity algorithm, which reduces the recommendation effect. To address this issue, we propose several methods to improve the similarity algorithm by considering common items, and apply the proposed methods to CF recommender systems. Experiments show that our methods demonstrate significant improvements over traditional CF.

Keywords: Personalized recommendation, collaborative filtering, common rating items, similarity

1. INTRODUCTION

The rapid development of Web 2.0 has created millions of commodities that make the Web more convenient for users. However, information overload has become a problem because of the enormous amount of content that is added to the Web daily[1]. Personal recommender (PR) systems can effectively reduce this problem by recommending goods which users may like. As a result, they may also increase users' loyalties and improve sales on e-commerce websites [2].

The recommendation process is performed using the following algorithms:

(1) Content-based filtering makes recommendations by matching the characteristic profiles of items to the interest profiles of users based on the user's purchase history[3, 4];

(2) Hybrid recommender systems combine different recommendation algorithms, such as content-based filtering and collaborative filtering, to improve the accuracy of the recommender system[5, 6];

(3) Resource allocation-based recommender systems hypothesize that the items chosen by the objective user contain related resources, determine those resources through a resource allocation algorithm, and then recommend items to the user based on these resources[7, 8];

(4) Collaborative filtering (CF) seeks the nearest neighbors of the objective user based on a similarity algorithm and predicts the rating of items which the objective user has not rated[9, 10].

Because CF is not restricted by whether an item can be directly assessed by a computer it is one of the most successful recommender system technologies, and has been used by large on-line enterprises (such as Amazon and Netflix). However, similarity algorithms have some limitations.

Breese et al. suggested that an item's weighting should be connected to the number of ratings, and proposed using an IUF method to determine an item's relative weight[11]. Alternatively, Herlocker et al. weighted items based on the spread of other user's ratings[12]. Candillier et al. combined Jaccard similarity and other similarity algorithms to improve the accuracy of the recommendations[13]. Ahn et al. proposed that the similarity algorithm must take account of the following factors: proximity, impact, and popularity[14]. Babadilla et al. used mean square difference to measure the similarity of users[15]. Heung-Num Kim et al. used prediction error information to refine the predicted ratings of the objective user[16].

In this paper, we show that the traditional similarity algorithm does not consider common items, which reduced the accuracy of the similarity algorithm and lessens the effect of the recommender. We propose several methods to improve the similarity algorithm by considering common items. Experiments demonstrate that our

* Corresponding author.
Email: mnswq@mail.sysu.edu.cn (Qian Wang), 502982914@qq.com (Taoqun Zhang), rongzhe@mail2.sysu.edu.cn (Zhe Rong)

methods have significant advantages over traditional CF.

The remainder of this paper is organized as follows. In section 2 we use simulations to determine weaknesses of the traditional similarity algorithm. In section 3 several improved similarity algorithms are proposed. Section 4 we determine the recommendation effects of different methods through experiments. In section 5 we evaluate and compare our methods with existing methods. In section 6 we conclude our work and describe future studies.

2. WEAKNESS IN THE SIMILARITY ALGORITHM

In the traditional CF algorithm (CFA), which chooses the nearest neighbor of the objective user, users are likely to be selected with fewer common items than the objective user. For example, in the MovieLens dataset (Figure 1), the average number of items in common with the objective user steadily increases with the length of the nearest neighbor path. This demonstrates that the fewer common items, the more likely it is to be at the top of the nearest neighbor list.

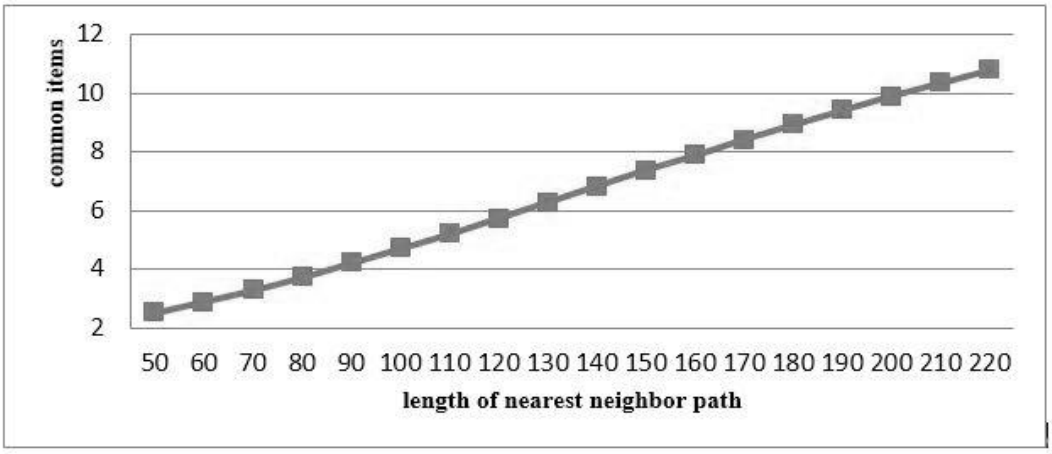

Figure 1. Average number of common items relative to the length of nearest neighbor path

The main reason for this is that the traditional similarity algorithm mostly considers the differences between users' ratings. By ignoring common items, it is unable to reliably determine the similarities between users, which results in reducing the effectiveness of the recommender. In the following section an experiment is presented to confirm the cause of this problem.

We propose that when adding a new common item where the difference between ratings is less than 1, the similarity of the users should be increased or reduced, as appropriate. Supposing that a user can rate an item from 1 to 5, there are 25 possible combinations of the ratings, u_i and u_j, of two users. If the difference between the ratings is less than 1, we use "+", otherwise we use "−".

Table 1. Similarities of users with a 5 score rating system

u_i \ u_j	1	2	3	4	5
1	+	+	−	−	−
2	+	+	+	−	−
3	−	+	+	+	−
4	−	−	+	+	+
5	−	−	−	+	+

From Table 1, we can determine that when adding a new common item, the probability of increasing or decreasing the similarity is almost equal (13/25 chance to increase; 12/25 chance to decrease). In the next section we will simulate the above process as follows:

1) Randomly generate two group numbers, $N_1 = \{n_1, \dots, n_s\}$ and $N_2 = \{m_1, \dots, m_s\}$, assign a number to each user i and j, then the initial similarity between users i and j is $sim(i,j)_{old}$.

2) Randomly generate two group numbers, $M_1 = \{p_1, \dots, p_t\}$ and $M_2 = \{q_1, \dots, q_t\}$, assign these numbers to users i and j, then combine with the existing data to calculate the similarity of users i and j as $sim(i,j)_{new}$.

3) Compare $sim(i,j)_{old}$ and $sim(i,j)_{new}$. If $sim(i,j)_{new} \geq sim(i,j)_{old}$, the counter is incremented to $Count + 1$.

4) Repeat the above process $T = 100000$ times, the calculate the percentage similarity $P(s, s+t) = Count/T$.

Table 2. Results of similarity simulation

S	*S* + *T*					
	s + t = 3	s + t = 4	s + t = 5	s + t = 6	s + t = 7	s + t = 8
s = 3	1	0.47	0.40	0.36	0.34	0.33
s = 4		1	0.51	0.44	0.41	0.40
s = 5			1	0.53	0.47	0.44
s = 6				1	0.54	0.48

Table 2 shows the results of the simulation. Combining the cases of 1 or 2 common items where the similarity of every pair of users is 1, we can conclude that a user having fewer items in common the target user is more likely to be selected as the nearest neighbor.

3. IMPROVING THE SIMILARITY ALGORITHM BASED ON COMMON ITEMS

Instead of the normal definition of similarity, we propose that user similarity contain two aspects: the similarity of users' ratings, which ensures the when two users are considered similar the differences between their ratings will be small; and the number of common items between users, such that only when the number of common items is above a threshold value can we consider users to be similar. Taking into account these factors, we propose new similarity algorithms based on whether the number of common items is more than a threshold value, in order to remove users with few items in common with the target user, and thus improve the effect of recommender systems.

3.1 Similarity algorithm based on percentage-threshold.

It is inappropriate to use a constant threshold for all users, for instance, a threshold of 5 items is too high when a user has rated 10 items, and too low when they have rated 100 items. Therefore, we suggest using a percentage of the number of items rated to determine the threshold value. There are two methods to take into account the number of items rated by the user. The first is the symmetrical method, which uses the minimum of two users' ratings. The second is the unsymmetrical method, which uses the target user's rating.

The similarity algorithm based on the symmetrical method is as follows:

$$sim(i,j) = \begin{cases} 0 & , \ CNum(i,j) < p * min\big(Num(i), Num(j)\big) \\ \dfrac{\sum_{u \in I_{ij}} (R_{i,u} - \bar{R}_i) \cdot (R_{j,u} - \bar{R}_j)}{\sqrt{\sum_{u \in I_{ij}} (R_{i,u} - \bar{R}_i)^2} \sqrt{\sum_{u \in I_{ij}} (R_{j,u} - \bar{R}_j)^2}} & , \ CNum(i,j) \geq p * min\big(Num(i), Num(j)\big) \end{cases} \quad (1)$$

Where p is the percentage, $Num(i)$ and $Num(j)$ is the number of items rated by users i and j, respectively, and $CNum(i,j)$ is the number of common items between users i and j. This is the symmetrical-percentage collaborative filtering algorithm (SPCFA).

The similarity algorithm based on the unsymmetrical method is defined as:

$$sim(i,j) = \begin{cases} 0 & , CNum(i,j) < p * Num(i) \\ \frac{\sum_{u \in I_{ij}} (R_{i,u} - \bar{R}_i) \cdot (R_{j,u} - \bar{R}_u)}{\sqrt{\sum_{u \in I_{ij}} (R_{i,u} - \bar{R}_i)^2} \sqrt{\sum_{u \in I_{ij}} (R_{j,u} - \bar{R}_j)^2}} & , CNum(i,j) \geq p * Num(i) \end{cases} \quad (2)$$

This is the dissymmetrical-percentage collaborative filtering algorithm (DSPCFA).

3.2 Similarity algorithm based on log-threshold value.

When using a percentage-based method to determine the threshold value, the threshold value increases proportionally with the number of items a user has rated. Therefore, this method is unsuitable for users with a large number of rated items. For example, if two users rated have rated 100 and 120 items the percentage difference is 30%, and the threshold value is 30. Therefore, these two used would not be considered similar. To resolve this problem, we propose using the log function to determine threshold values, because as the log function increases, the rate of the increase in the threshold decreases. As with the percentage threshold method, the log-based threshold is divided into symmetrical and dissymmetrical methods.

The similarity algorithm based on the symmetrical-log method (SLCFA) is defined as:

$$sim(i,j) = \begin{cases} 0 & , CNum(i,j) < a + b * log\left(min\left(Num(i), Num(j)\right)\right) \\ \frac{\sum_{u \in I_{ij}} (R_{i,u} - \bar{R}_i) \cdot (R_{j,u} - \bar{R}_j)}{\sqrt{\sum_{u \in I_{ij}} (R_{i,u} - \bar{R}_i)^2} \sqrt{\sum_{u \in I_{ij}} (R_{j,u} - \bar{R}_j)^2}} & , CNum(i,j) \geq a + b * log\left(min\left(Num(i), Num(j)\right)\right) \end{cases} \quad (3)$$

Where a, b is a coefficient variable.

The similarity algorithm based on the dissymmetrical-log method (DLCFA) is defined as:

$$sim(i,j) = \begin{cases} 0 & , CNum(i,j) < a + b * log\left(Num(i)\right) \\ \frac{\sum_{u \in I_{ij}} (R_{i,u} - \bar{R}_i) \cdot (R_{j,u} - \bar{R}_j)}{\sqrt{\sum_{u \in I_{ij}} (R_{i,u} - \bar{R}_i)^2} \sqrt{\sum_{u \in I_{ij}} (R_{j,u} - \bar{R}_j)^2}} & , CNum(i,j) \geq a + b * log\left(Num(i)\right) \end{cases} \quad (4)$$

3.3 Similarity algorithm based on probability threshold value.

We believe that if two users are similar, the number of common items should be larger than the expectation of the common items between the target user and a randomly selected user. Therefore, we propose using the expectation and the standard deviation of common items to determine the threshold value. Supposing the target user has rated n items, the frequency of items rated is M_i, and U is the total number of users, so the probability of an item selected by the user is defined as:

$$P = \frac{M_i}{U} \quad (5)$$

To simplify the calculation we assume that every item is dependent, hence, the expectation of the number of common items with the target user can be calculated as:

$$E(U_i) = p_1 + p_2 + \cdots + p_i + \cdots + p_n \quad (6)$$

The calculation of the standard deviation is as follows:

$$D(U_i) = p_1(1 - p_1) + p_2(1 - p_2) + \cdots + p_i(1 - p_i) + \cdots + p_n(1 - p_n) \quad (7)$$

According to the central limit theorem, the number of common items obeys normal distribution, although n may be less than 30 in some case. For a degree of confidence a, the threshold value can be calculated as follows:

$$N(U_i) = E(U_i) + z_{\frac{a}{2}} \sqrt{D(U_i)} = E(U_i) + w * \sqrt{D(U_i)} \quad (8)$$

The above method is the probability and statistics collaborative filtering algorithm (PSCFA).

4. EXPERIMENTAL EVALUATIONS

In this section, we report the results of the experimental evaluations of the proposed similarity algorithms, and determine the optimal parameters for each method.

4.1 Dataset and evaluation metrics

The experimental dataset was from MovieLens, an online movie recommender system developed by the University of Minnesota. The dataset includes 1,000,000 ratings of 1682 movies by 943 users. This dataset is publicly available. The dataset was randomly divided into two parts: the training dataset (90% of the data), and the test dataset (10% of the data).

To measure the accuracy of the predicted score, we employed the root mean squared error method, which is widely used in research. The root mean squared error of user i for M items is defined as:

$$RMSE_i = \sqrt{\frac{1}{M}\sum_{(i,j)}\left(p_{ij} - r_{ij}\right)^2} \tag{9}$$

Where $< p_{ij}, r_{ij} >$ are the predicted and actual ratings, respectively, of user i. The error of the whole recommender system is:

$$RMSE = \frac{\sum_{i=1}^{N} RMSE_i}{N} \tag{10}$$

4.2 Parameter experiments

In this section, we show detailed experiments results of the proposed algorithms with different parameters.

4.2.1 Effectiveness of CFA using adjusted cosine similarity

A single parameter, the length of the nearest neighbor path, L, was adjusted in this experiment. L was varied from 50 to 230 in intervals of 10.

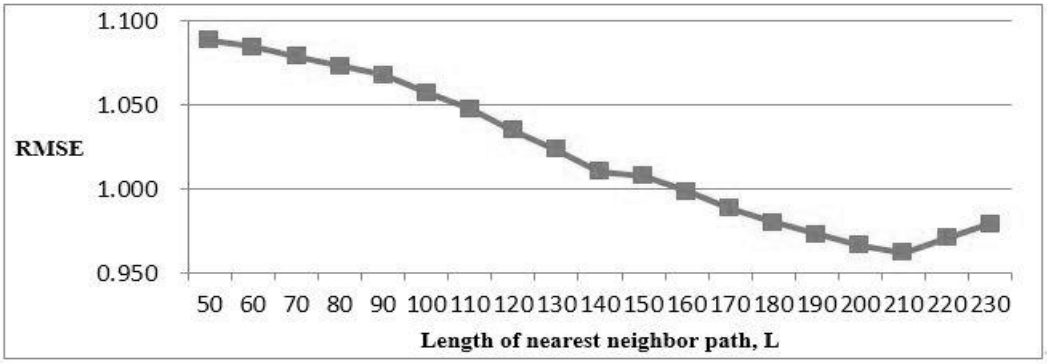

Figure 2. Effectiveness of CFA using adjusted cosine similarity

Figure 2 shows that the length of the nearest neighbor path significantly influences the recommender. When L=205, the $RMSE$ was minimized to 0.962.

4.2.2 The effectiveness of CFA based on percentage threshold value

In this experiment, the length of the nearest neighbor path, L, and the percentage, P, were controlled. L ranged from 50 to 200 in intervals of 10, and P from 0 to 1 in intervals of 0.05. The detailed results of SPCFA are shown in figure 3.

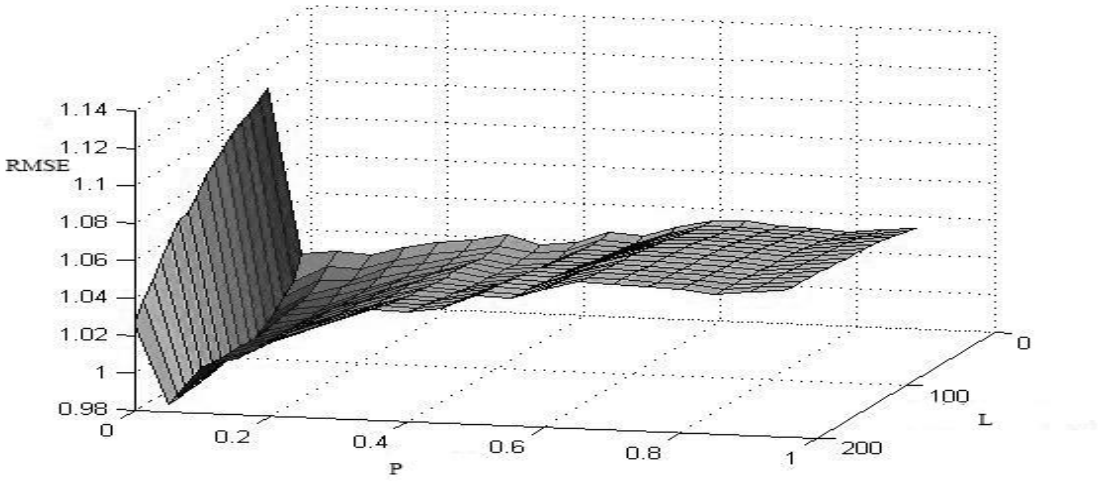

Figure 3. Results of SPCFA

When L=200, the $RMSE$ was minimized to 0.960. This algorithm did not greatly improve the effectiveness of the recommender over that using the traditional CFA, because it does not remove users with few items in common with the target user from the nearest neighbor list.

The parameters of the DSPCFA experiment were the same those of the SPCFA experiment, and the results are presented in figure 4.

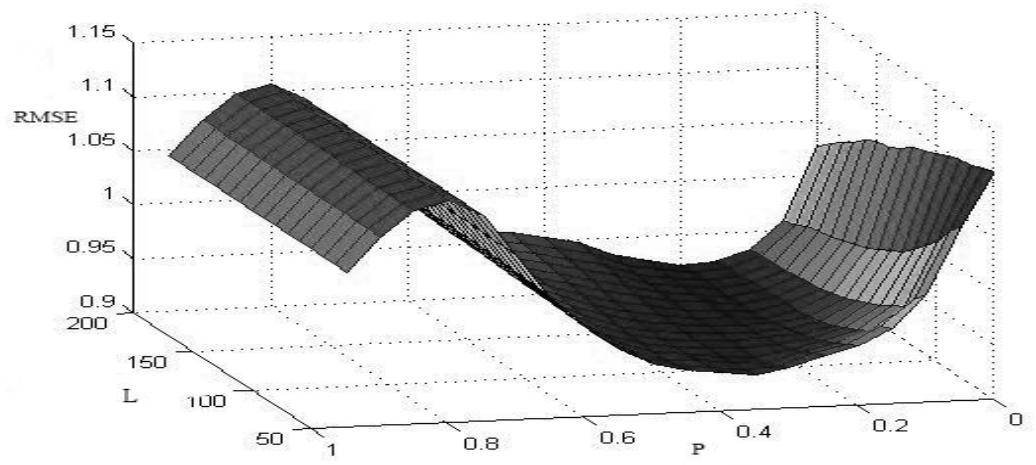

Figure 4. Results of DSPCFA

When $RMSE$ was minimized L=80, P=0.30, and $RMSE$=0.925. This is a great improvement over the traditional CFA.

4.2.3 Effectiveness of CFA based on log function threshold value

Three parameters were adjusted in this experiment: the length of the nearest neighbor path, L, and parameters a and b. As the effects of adjusting parameter a can be achieved by adjusting parameter b, there is no need to change parameter a. Therefore, we set a=4. L was adjusted from 50 to 200 in intervals of 10, and b from 1 to 3 in intervals of 0.1.

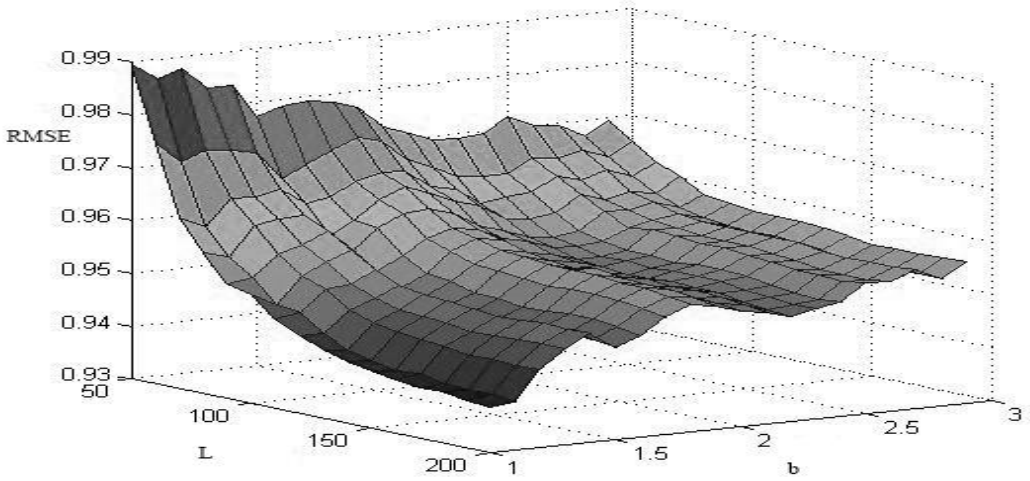

Figure 5. Result of SLCFA

From figure 5 that when L=190, b=1, and $RMSE$=0.939. This is a slight improvement over the traditional CFA.

The parameters of the DSLCFA experiment were the same as the SLCFA experiment, and the results are shown in figure 6.

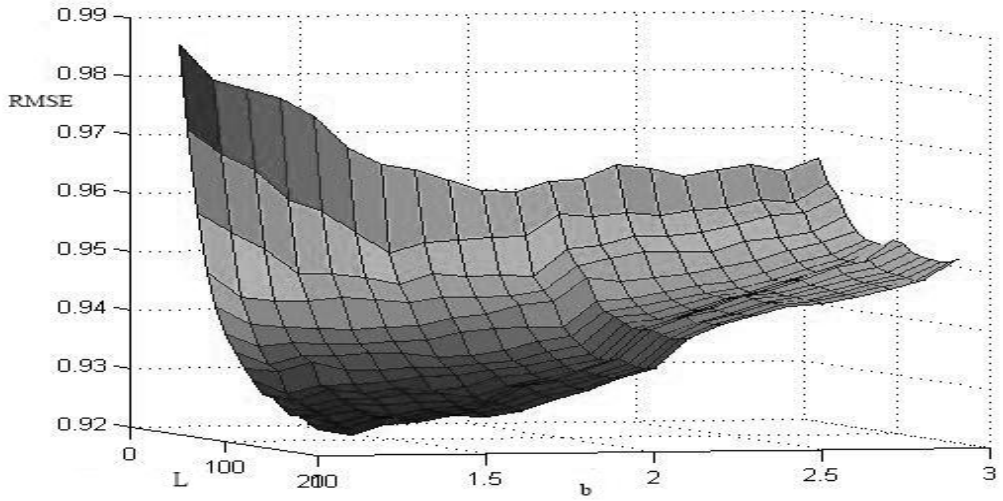

Figure 6. Result of DSLCFA

From graph 5, when L=150, b=1.3, and $RMSE$=0.9272. It is a very significant improvement over the traditional CFA.

4.2.4 Effectiveness of CFA based on probability threshold value

In this experiment, the length of the nearest neighbor path, L, and the parameter w were adjusted. L ranged from 50 to 200 in intervals of 10, and w from 1 to 3 in intervals of 0.1. The detailed results are shown in figure 7.

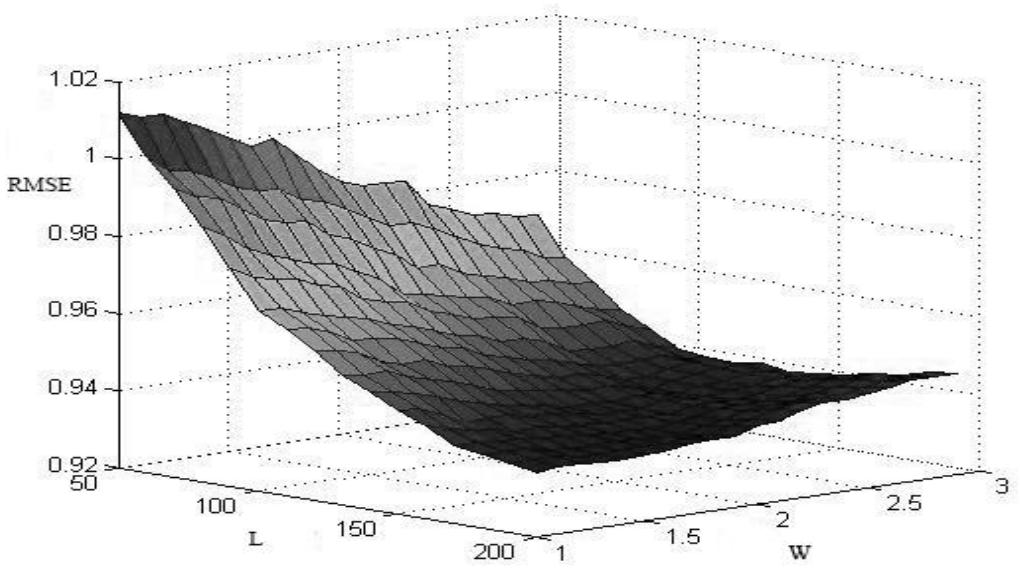

Figure 7. Results of PSCFA

Graph 7 shows that when L=150, w=2.2, and $RMSE$=0.920. This is a significant improvement over the traditional CFA.

5. COMPARISON OF DIFFERENT METHODS

In this section, we compare the proposed methods with each other and traditional CFA with regards to prediction accuracy and computational complexity. The results with the lowest error for each method are presented in table 3.

Table 3. Results of different methods

	Method	Parameter	*L*	*RSME*	Improvement
1	CFA	——	210	0.962	——
2	$p * min(L(i), L(j))$	p = 0.03	200	0.960	0.2%
3	$p * L(i)$	p = 0.32	80	0.925	3.85%
4	$a + b * log\left(min(Num(i), Num(j))\right)$	a = 4, b = 1	190	0.939	2.39%
5	$a + b * log(L(i))$	a = 4, b = 1.6	150	0.927	3.64%
6	$E(U_i) + w * \sqrt{D(U_i)}$	w = 2.2	150	0.920	4.37%

5.1 Comparing prediction accuracy

In Table 3, the methods that consider common items show great improvements in prediction accuracy over the traditional CFA. As it does not take account of common items, the traditional CFA is poor at determining similarity, and selects some users with few items in common with the target user as the nearest neighbor. This limits the effectiveness of the traditional CFA. All of the methods which consider common items determined similarities between users more accurately than the traditional CFA. The greatest improvement was 4.37%, achieved by the PSCFA, followed by 3.85% attained by the DSPCFA. These results demonstrate that similarity algorithms based on common items improve the accuracy of recommender systems.

5.2 Comparing computation complexity

In this section, we compare the computational complexity of the proposed methods and of the traditional CFA. Generally, increasing the prediction accuracy requires a corresponding increase in computational complexity. The magnitude of this increase affects the extensibility and practicality of using a method in real-time. Therefore, when evaluating methods we also need to compare the computational complexities.

The computational complexity is related to the following parameters: the number of users *m*, the number of items *n*, and the length of the nearest neighbor path L. The recommendation process can be divided into two parts: 1) calculating similarities with other users, and 2) predicting ratings based on the scores of the nearest neighbor. The upper bound of the similarity calculation is $O(m^2n)$, and the upper bound of the prediction is $O(mln)$. However, the similarities can be calculated off-line, so this part does not significantly effect real-time requests. Therefore, the prediction part of the process plays a greater role in determining the computational complexity. The length of the nearest neighbor path can be used to compare the complexities of different methods. From table 3, the similarity algorithms which consider common items are superior to the traditional CFA, because these algorithms remove users that are have few items in common with the target user, which reduces the computation complexity. The lowest complexity was for the DSPCFA where L= 80.

6. CONCLUSIONS AND FUTURE WORK

We have shown that the main problem of the traditional similarity algorithm is that it ignores common items. Based on this analysis, we proposed several similarity algorithms based on common items. Our experiments demonstrated that the proposed similarity algorithms show significant improvements over the traditional CF.

In future work, we intend to use social networks to improve the similarity algorithms further. Because the preferences of friends tend to be similar, we would like to be able to recommend goods to users based on their friends' purchase histories.

ACKNOWLEDGEMENT

This research is sponsored by the National Natural Science Foundation of China (70971141) and the Natural Science Foundation of Guang Dong (2014A030313184), the Ministry of education of Humanities and Social Science (15YJA630070).

REFERENCES

[1] Suo Qi, Sun Shiwei, Hajli N, Love PED. (2015).User ratings analysis in social networks through a hypernetwork method. Expert Systems with Applications, 41(5): 7317-7325.

[2] Gogna A, Majumdar A. (2015).Matrix completion incorporating auxiliary information for recommender system design. Expert Systems with Applications, 42(4): 5789-5799.

[3] Adomavicius G, Tuzhilin A. (2005).Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Transactions on Knowledge and Data Engineering, 17(6).734-749.

[4] Park D H, Kim H K, Choi Y I, Kim K J. (2012).A literature review and classification of recommender systems research. Expert Systems with Applications, 39(11): 10059–10072.

[5] Nilashi M, bin Ibrahim O, Ithnin N. (2014).Multi-criteria collaborative filtering with high accuracy using higher order singular value decomposition and Neuro-Fuzzy system. Knowledge-Based Systems, 60: 82-101.

[6] Robin B. (2002).Hybrid recommender systems: Survey and experiments. User Modeling and User-Adapted Interaction, 12(4): 331-370.

[7] Gan Mingxin, Jiang Rui. (2013).Constructing a user similarity network to remove adverse influence of popular objects for personalized recommendation, 40(10): 4044-4052.

[8] Hong J Y, Suh E H, Kim S J. (2009).Context-aware systems: A literature review and classification, 36(4): 8509–8522.

[9] Bauer J, Nanopoulos A. (2014).Recommender systems based on quantitative implicit customer feedback. Decision Support Systems, 68: 77-88.

[10] Herlocker J L, Konstan J A, Terveen L G, Riedl J T. (2004).Evaluating collaborative filtering recommender systems. ACM Transactions on Information Systems, 22(1): 5-53.

[11] Breese J S, Herckerman D, Kadie C. (1998).Empirical analysis of predictive algorithm for collaborative filtering. Proceedings of the 14th conference on Uncertainty in Artificial Intelligence. San Francisco: Morgan Kaufmann Publishers Inc , 43-52.

[12] Herlocker J L, Konstan J A, Borchers A, Riedl J. (1999).An algorithmic framework for performing collaborative filtering. Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. New York: ACM, 230-237.

[13] Candillier L, Meyer F, Fessant F. (2008).Designing Specific Weighted Similarity Measures to Improve Collaborative Filtering Systems. In: Perner P. (eds) Advances in Data Mining. Medical Applications, E-Commerce, Marketing, and Theoretical Aspects. ICDM 2008. Lecture Notes in Computer Science, vol 5077. Springer, Berlin, Heidelberg

[14] Ahn H J. (2008). A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem. Information Sciences, 178(1,2): 37-51.

[15] Bodadilla J, Serradilla F, Bernal J. (2010).A new collaborative filtering metric that improves the behavior of recommender systems. Knowledge-Based Systems, 23(6): 520-528.

[16] Kim H N, Ha I, Lee K S, Jo G S, El-Saddik A. (2011).Collaborative user modeling for enhanced content filtering in recommender systems. Decision Support Systems, 51(4): 772-781.