TELKOMNIKA Indonesian Journal of Electrical Engineering
Vol. 12, No. 8, August 2014, pp. 6393 ~ 6402
DOI: 10.11591/telkomnika.v12i8.5257

Time-Weighted Uncertain Nearest Neighbor Collaborative Filtering Algorithm

Zheng Zhi-Gao*[1], Wang Ping[1,2], Sun Sheng-Li[1]
[1]School of Software and Microelectronics, Peking University,
Beijing 100260, China
[2]National Engineering Research Center for Software Engineering, Peking University,
Beijing 100871, China
*Corresponding author, e-mail: zhengzhigao@pku.edu.cn

Abstract

To overcome the limitations of the traditional collaborative filtering recommendation algorithm, this paper proposed a Time-Weighted Uncertain Nearest Neighbor Collaborative Filtering Algorithm (TWUNCF). According to the actual application situation of recommendation system, the author weighted the product similarity and user similarity to ensure the data validity firstly, and then calculate the similarities of user and product and choose the trusted neighbor group as the recommended object adaptively based on the weight. Experimental results show that the algorithm can be used to improve data validity according to the time attribute, and balance the impact the different groups on the recommendation result, and avoid the problems which caused by the data sparseness. Theoretical analysis and experimental demonstrations show that the algorithm this paper proposed outperforms existing algorithms in recommendation quality, and improve the system's accuracy and recommendation efficiency.

Keywords: *collaborative filtering, time weight, uncertain neighbors, trustworthy subset, recommendation system*

1. Introduction

With the development and popularization of e-commerce, many researchers and scholars have made the relevant research on the recommendation efficiency and accuracy of recommendation system in order to mining potential customers greatly. Many scholars have proposed a variety of recommendation algorithm, among which collaborative filtering recommendation algorithm is the most widely used.

Currently the research, based on collaborative filtering, is mainly divided into two kinds: user-based collaborative filtering and product-based collaborative filtering. Either the user or the product, there is individual variation, which results in the recommendation differences. In the user-based recommendation, the traditional user-based collaborative filtering algorithm or product-based one has some limitations, due to the inherent differences among users and the uncertainty of scenarios prediction and the variability of production prediction, mainly in the following three areas: (1) Many scholars usually use the *k*NN [1-3] method to recommend an object for the target. Since the *k*NN method chooses *k* nearest neighbor through a certain similarity comparison, it can present the characteristics of the predicted target to a certain extent. However, *k* is a common argument and usually do not have the particularity, which makes it may be not available in certain scenarios. For example, an extreme case that when the number of the object's neighbor is smaller than *k*, the *k*NN methods will recommend several individuals which is quite different from the object as the object's neighbor, making the recommendation unreliable. (2) Current recommendation algorithm often only recommend for a certain single group of users or products, ignoring the impact for the other groups [3-6]. Since we neither have deeply understanding in the recommended individual dimension trustworthy subset before we recommend, nor study its influence on recommendation result, the quality of recommendation to some extent can't meet the needs of people in every aspect. (3) Traditional recommendation algorithm does not consider the fact that the value of the data decreases with time when make decisions. Treating different data during different periods affects the accuracy and feasibility of recommendation to some extent.

Received November 30, 2013; Revised April 16, 2014; Accepted May 5, 2014

In order to solve the three issues above, this paper, using the idea of dynamic selection, on the basis of the existing research, adaptively choose recommended object's trustworthy subset in different dimensions under different scenarios and requirements as recommended candidate set, and propose a Time Weighted Uncertain Neighborhood Collaboration Filtering, TWUNCF. This algorithm, by using uses logistic methods to weigh time for scores, distinguish scores in different periods and fully consider the fact that the value of data decreases over time to ensure the validity of the data against time. As to the data with same time attribute, it is calculated based on the similarity of users and products, in the meantime it determines which trustworthy subset of recommended target as the recommended candidate set and selects the neighbor of predicted target adaptively. Experiments show that this algorithm, Time Weighted Uncertain Neighborhood Collaboration Filtering, can effectively balance the impact of different groups on recommendation, has good performance in solving score data sparseness problems, and takes into account the impact of the diminishing value of time on the recommendation result. As a result, the proposed algorithm in this paper improves the quality of recommendation and has good stability to some extent. This paper is organized as follows: Section 2 describes related work and defines the issues to be solved in this paper; Section 3 details the Time Weighted Uncertain Neighborhood Collaboration Filtering algorithm; Section 4 designs multiple experiments to validate the proposed algorithm and makes a simple analysis; finally make a summary.

2. Relevant Work and Problem Definition

2.1. Relevant Work

Currently, data sparseness [7], cold start [8] and scalability [9] are common in recommendation system. To solve these three issues, many researchers have made a lot of research, hoping to improve them by a new algorithm. For example, Seung-Taek Park created a new search model MAD6 [10] by combining collaborative filtering algorithm with search engine tools, and applied it to Yahoo! [4]; Tomoharu used maximum entropy principle to predict those products consumers interested based on collaborative filtering algorithm [5], and it turned out to be a success when applied to E-commerce system; Chen [2] etc. applied dual collaborative filtering algorithm to search for products that target users may be interested in, which using collaborative filtering methods again in the first result set for a second recommendation; Gu proposed a time weighted recommendation algorithm, taking time properties [11] into consideration properly in the process of recommendation; Huang brought out an uncertain nearest neighbor recommendation algorithm which comprehensively considered different groups of users and products, balancing the user group and the product group [12]; Chen improved resource assessment density through the establishment of k nearest neighbor and its impact set, and he defined a new recommendation mechanism to calculate the score of the prediction [13], which alleviated the data sparseness problem effectively and improved the quality of the recommendation; Liu used Beta distribution to predict the similarity of users based on trustworthy group, improving the recommended result to a certain extent [14]; Jamali, in order to improve the quality of recommendation, made a deep digging in the trust relationship between users, found the deep user similarity and made a recommendation by some data mining methods [15], and finally improved the recommendation efficiency of the system.

2.2. Problem Definition

Traditional collaborative filtering algorithm finds k neighbors who influence current individual most through the inner individual interaction between the user group and the product group to predict current individual property. But with the increasing use of recommendation system and the increasing complexity of the environment, this method of intercepting k neighbors turns to be partial. This one-sidedness is mainly manifested in two aspects. One is that considering separately the impact of a particular group and ignoring the impact of another group which itself is unreasonable. The other one is that when the data of individuals in the group is sparse, the number of neighbors in the cluster may be less than the value of k in kNN, the basic recommendation algorithm if simply recommend based on users or products. In this case, users or products with low similarity have to be added into the training set, thus leading to a sharp decline in the accuracy of the algorithm. And its result is often inaccurate or even wrong.

For example, the methods currently used to predict how much a person likes an item is User Based Collaborative Filtering, UBCF and Item Based Collaborative Filtering, IBCF. Though these methods to some extent are capable of recommending, the constraints of reality make its accuracy far more than satisfactory. For instance, when a user is interested in an item while few people did and made little evaluation, the accuracy of recommendation based on UBCF is relatively low. Due to the complexity of the reality, we needs to consider every aspect of the users and the products adaptively selects in user group and product group, but not simply consider one aspect and ignore the others when making a recommendation. According to the needs of actual situation, we can dynamically select the nearest neighbor and the neighborhood factor, pick neighbors properly out of user groups and product groups and also recommend for the current object.

Even the number of the user in the user cluster or the number of the product in the product cluster satisfies the lowest value of k in kNN, time inconsistency may also exist. Since a user's interest may change with time, the score that a user gives to the same project may vary with time too. However, traditional algorithm treats a user's scores equally in finding a user's nearest neighbor without taking the variation with time of the user's interest into account, resulting in the calculated may not be the neighbor group the user really interests. The point is the accuracy of the kNN algorithm depends on how much the selected neighbors match with the target users, which is one of the important reasons why the accuracy of traditional algorithm should be improved. For example, a user A was interested in action movies during a certain period of the past and scored highly on such films while another user B is interested in it currently and scores highly on the same films. According to the calculation principle of similarity, the two users should be each other's neighbors. But in fact, it's obviously unreasonable to recommend based on two users' interests in different time. First of all, the current interest of A may not be as same as B's. Secondly, influenced by social popularity, what A likes at that time may not be favored by B currently. Usually, searching the similarity among the scores that different users give to the same item within the same or similar period of time can ensure the effectiveness of the selected neighbor. Table 1 illustrates it with an example.

Table 1. Simple Example

User	Project					
	I_1	I_2	I_3	I_4	I_5	I_6
u_1(T_1)	4	3	4	3	3	4
u_2(T_4)	3	4	2	3	4	2
u_3(T_1)	3	4	4	3	3	4
u_4(T2)	4	3	2	4	3	2

Table 1 shows the scores that 4 users give to 6 projects in 3 periods where T1 and T2 are similar period of time, T1 and T4 are far away from each other.

According to the assumption above, u_2、u_3 and u_4 constitute the nearest neighbor set of u_1 when recommend with 3 neighbor users by the traditional collaborative filtering recommendation algorithm, and the similarity satisfies the condition $sim(u_1,u_2) > sim(u_1,u_3) > sim(u_1,u_4)$. However, according to the data given in Table 1, the time gap is large between the time when user u_1 and user u_2 each scores the same projects. As the analyses above, it is unreasonable to predict the interest of user u_1 based on the score data of user u_2 in a certain period of the last and give a reconmendation to u_1. If taking the time into consideration, only u_3 and u_4 is the neighbor of u_1, which is more consistent with the fact.

3. Time-Weighted Uncertain Nearest Neighbor Collaborative Filtering Algorithm

3.1. Time-Weighted Nearest Neighbor

Without limits to the period of time when selecting the k neighbors of the user or the item, it is easy to take outdated data as neighbors into consideration (Like make a recommendation based on the thing that a user interested 20 years ago), which turns out to be unsatisfactory. In the traditional collaborative filtering algorithm, not distinguishing the data's time effectiveness, to some extent, affects the accuracy of the result. Taking the influence that

the time has on the target group comprehensively and modifying the similarity by weighing the time can avoid the low accuracy of the recommendation result caused by time inconsistency effectively.

As a result, this paper proposes a Time Weighted Selected Neighbor method. Before selecting the neighbors, time-weighs and modify the scores with $logistic$ function. In this paper, it uses the Pearson Correlation Method to calculate the similarity. Firstly, weigh the scores of different users and different periods of time with $logistic$ function, replace $r_{u,c}$ with $r_{u,c} \times logistic\left(r_{u,c}\right)$, making each score has its time weight. Since the latest score represents a user's interest most, the current weight should be larger than the last one and the scores of different time should be distinguished.

Here is the $logistic$ function.

$$logistic(t_{u,j}) = \frac{1}{1+e^{-t_{u,j}}} \tag{1}$$

Among them, $-1 \le t \le 1$, $0 < logistic(t_{u,j}) < 1$. $logistic(t_{u,j})$ is a monotonic increasing function, the weight increases with time t increasing, and the value is always within (0,1). This paper firstly converts the range of time to [-1,1] by using a standardized conversion method. By doing this, the weight's variation with time is almost linear, and the change of user's interest can be detected intuitively [16].

The most important step in collaborative filtering algorithm is the formation of the target user's neighbors. Here it uses the nearest neighbor which is used in the computing of Pearson correlation. The Pearson relevant similarity based on time-weighted of the users is:

$$sim(U_a, U_b) = \frac{\sum_{c \in I_U} (r_{U_{a,c}} \times logistic(t_{U_{a,c}}) - \overline{r_{U_a}})(r_{U_{b,c}} \times logistic(t_{U_{b,c}}) - \overline{r_{U_b}})}{\sqrt{\sum_{c \in I_U} (r_{U_{a,c}} \times logistic(t_{U_{a,c}}))^2} \sqrt{\sum_{c \in I_{Ui}} (r_{U_{b,c}} \times logistic(t_{U_{b,c}}))^2}} \tag{2}$$

In the above formula, $sim(U_a, U_b)$ is the similarity between the target user U_a and its nearest neighbor U_b, I_U presents the project set that both user U_a and U_b scores, the score user U_a gives to project c is , the each average score that user U_a and user U_b give to a project is $\overline{r_{U_a}}$ and $\overline{r_{U_b}}$. Assuming that U_a is the target user, we can predict the score that U_a gives with the scores U_b gives to any item $j(j \notin I_U)$. Here is the formula:

$$P_{u,j} = \overline{r_{U_b}} + \frac{\sum_{i=1}^{n} [sim(U_a, U_b)(r_{U_{a,c}} - \overline{r_{U_a}})]}{\sum_{i=1}^{n} [sim(U_a, U_b)]} \tag{3}$$

Here it calculates the similarity by time-weighted Pearson Correlation Function, and the similarity calculation of the users and the one of the projects is carried out at the same time. It predicts the value that a user scores other projects, and then takes those items which don't belong to I_U into the recommendation set in descending mode. And finally select the proper project from the recommendation set to make a recommendation.

3.2. Improved Similarity Computing

The similarity among users is weighed with the scores that different users give to the same projects or the same items. If the number of the same projects or items the users score is

so small that it is unable to meet the minimum number of neighbor, the accuracy of its recommendation result will decrease. In order to avoid this occasional influence, some researchers already do some research on it. Herlocker etc. improves the similarity calculation with an increased weight of the relevance; Ma etc. points out its specific settings in document [17]. And this paper controls it by setting a threshold. Supposing that $I' = U_a \bigcap U_b$ presents the items that both user U_a and user U_b have scored, we add the proportion of different users score in the same or similar time period to improve the similarity among users.

$$sim'(U_a, U_b) = \frac{\min(|I'|, \gamma)}{\gamma} \times sim(U_a, U_b) \tag{4}$$

Among them γ is the threshold, and according to its definition its maximum value is the number of the projects that the users both scores. So $\frac{\min(|I'|, \gamma)}{\gamma} \le 1$, and the range of the improved user similarity $sim'(U_a, U_b)$ is still [0,1]. When the number of the same projects the users score is larger than its threshold, $sim'(U_a, U_b) = sim(U_a, U_b)$. In the same way, when the number of the same projects the users score is small, the similarity among users should be decreased to improve its influence on the recommendation accuracy.

3.3. Uncertain Nearest Neighbor Trustworthy Subset

Currently the user-based or product-based collaborative filtering algorithm is common in recommendation system. However due to the variability of the user's demands on the quality of recommendation and the reality, the recommendation always turns out to be unable to meet the user's needs if only using one methods especially when the data is sparseness. That how to consider several factors comprehensively and how to weigh them automatically is what we need to solve. Aiming at the characteristic of a recommendation system making decision with the nearest neighbors.

This paper optimizes the selection of recommendation set, and weighs between the user and the product adaptively to avoid too much human involvement which results in reduced flexibility of recommendation system.

In document [12], it proposes a method to weigh the user's similarity and the product's by setting two similarity threshold μ and ν, and dynamically select the proper neighbor of the recommendation target in user's group and product's group before selecting its neighbors. This method, to some extent, can overcome the shortcoming of traditional algorithm *k*NN and dynamically select the neighbor, but it needs 2 thresholds to weigh it. However, setting a threshold has some influence on neighbor selecting, and its calculation is relevantly complex. If the threshold is set too big, the number of recommendation group will decrease and the recommendation will lack of generality; if the threshold is set too small, it is easy to bring in neighbor with low similarity which affects the accuracy of the recommendation. Therefore, in the absence of prior knowledge, this method is still not well enough to select the user's neighbor. In order to overcome the shortcoming in document [12], this paper brings in a Harmonic parameters to balance the user-based method and the product-based method.

Here, we note the group of the predicted score that user U_a to item I_j as $S'(U_a) = \{U_{a_1}, U_{a_2}, \cdots, U_{a_{m'}}\}$, and note the size of the group as $|S'(U_a)| = m'$. The item group that user U_a has scored in the neighbor of item I_j is noted as $S(I_j) = \{I_{j_1}, I_{j_2}, \cdots, I_{j_{n'}}\}$, and $|S(I_j)| = n'$. The computing procedure of m' and n' is relatively easy so that it can be done off-line. This paper introduces neighbor factors λ and $1-\lambda$ as the balance factor of the user group and that of the item group separately, and adjusts the value of the neighbor factor with harmonic parameters.

$$\lambda = \begin{cases} \dfrac{\phi \times m'}{\phi \times m' + n'} & m' + n' > 0 \\ \dfrac{\phi \times m}{\phi \times m + n} & m + n > 0 \\ 0.5 & \text{其他} \end{cases}$$

$$1 - \lambda = \begin{cases} \dfrac{\phi \times n'}{\phi \times m' + n'} & m' + n' > 0 \\ \dfrac{\phi \times n}{\phi \times m + n} & m + n > 0 \\ 0.5 & \text{其他} \end{cases}$$

Among them, ϕ is the harmonic parameter. In the following, we take λ for example to analyze how ϕ adjusts the neighbor factor. When $m'+n'>0$, λ has the following four possible values:

When $m'=0$ as well as $n'>0$, it means making recommendations by item-based collaborative filtering method. In this case, $\lambda=0$, the value of λ has nothing to do with the value of ϕ.

When $\phi=\frac{n'}{m'}$, $\lambda=1-\lambda=0.5$. In this case, it means half of the recommendation set half comes from item-based recommendation set and half user-based recommendation set.

When $\phi \in (\frac{n'}{m'}, \infty)$, λ is an increasing function, and its range is (0.5,1). $1-\lambda$ is an decreasing function, its range is (0,0.5). It means that with the increasing value of ϕ, the projects in recommendation set comes from the user group more. An extreme case is $n'=0$ and $\lambda=1$, which means the method is user-based collaborative filtering method.

When $\phi \in [0, \frac{n'}{m'})$, λ is an decreasing function, and its range is (0,0.5). It means the recommendation set gradually tends to the item group. When $\lambda=0$, it means making recommendation with item-based collaborative filtering method.

In order to balance the impact of user group and item group on different dimension, this paper introduces a harmonic parameters ϕ to automatically adjust the source of recommendation set, avoid the low quality from recommending by single group. The improved method in this paper reduces the user's intervention to the algorithm by reducing the number and the difficulty of the thresholds the user needs to assign to. Compared to the thresholds in document [12], the one in this paper is more consistent with user habits, reducing the complexity of the user operation. In document [12], μ and ν is sensitive to the selection of neighbors, resulting in big influence on recommendation result, while this paper introduces a harmonic parameters to control the threshold, reducing the sensitivity of neighbor selection and ensuring the accuracy of recommendation result. In the other hand, the threshold's setting in this paper is relatively easy, and the best threshold can be obtained after multiple tests, which also ensuring the algorithm's accuracy and reducing human's intervention.

3.4. Time-Weighted Uncertain Nearest Neighbor Collaborative Filtering Algorithm

According to the analysis above, this paper fully considers the time attribute of the score, and weighs the user-based score and the item-based one based on that. It takes into account many aspects, and proposes a Time Weighted Uncertain Neighborhood Collaboration

Filtering algorithm(TWUNCF) to predict the user's score. If the average score of user U_a, U_x on products is represented separately by $\overline{R_a}, \overline{R_x}$ and the average score of a known user on products I_j, I_y is represented separately by $\overline{R_j}, \overline{R_y}$, then the TWUNCF proposed in this paper can be represented by the formula below.

$$R_{a,j} = \lambda(\overline{R_a} + \frac{\sum_{U_x \in S(U_a)} sim'(U_a, U_x) \times (R_{x,j}, \overline{R_x})}{\sum_{U_x \in S(U_a)} sim'(U_a, U_x)}) + (1-\lambda)(\overline{R_j} + \frac{\sum_{I_y \in S(I_j)} sim'(I_j, I_y) \times (R_{a,y}, \overline{R_y})}{\sum_{I_y \in S(I_j)} sim'(I_j, I_y)})$$

Based on uncertain scenes, the algorithm TWUNCF firstly brings in a time variable by *logistic* function, weigh time against the similarity between the user group and the item group to distinguish the time effectiveness of the similarity and secondly on this basis considers comprehensively the similarity of the user and the item, controls the neighbor factor of different groups by harmonic function indirectly and then considers the impact of the user and the item, and ultimately produce the recommended result set.

Therefore, there are 2 steps. Step 1, select the trustworthy subset of the recommended target; step 2, make a recommendation based on the selected recommendation set and produce the final result. The description of the algorithm is illustrated in Figure 1.

TWUNCF Algorithm

Input: the target user U_a, the item waited to be scored I_j, harmonic parameters ϕ

Output: the score $R_{a,j}$ that the predicted user U_a gives to the item I_j

Step 1 Separately calculate the user similarity matrix and the item similarity matrix based on the score matrix $R(s \times t)$, and save the two matrix separately $Arr\_UsrSim(s,s)$ and $Arr\_ItemSim(t,t)$;

Step 2 Weigh time with logistic function;

Step 3 Determine the validity of the time;

Step 4 Calculate $|S'(U_a)| = m'$ and $|S(I_j)| = n'$ based on the user's score and the item's score, and calculate the trustworthy subset of $S'(I_j)$;

Step 5 Determine a proper harmonic parameter ϕ;

Step 6 Calculate the value of λ;

Step 7 Calculate the predicted score $R_{a,j}$ that user U_a should give to item I_j.

Figure 1. TWUNCF Algorithm

In TWUNCF algorithm, the first thing to do is to determine the score matrix of users and items. This could be done off line to save time complexity of the algorithm. And it is $O(s^2 + t^2)$ in the worst case. Since both $|S'(U_a)| = m'$ and $|S(I_j)| = n'$ are constant, the time complexity of computing the neighbor sand the trustworthy subset is $O(m + n + m' + n') = O(1)$. In the best condition, the time complexity is $O(1)$, effectively avoiding the calculating problem caused by data sparseness and data accumulation. And compared to the DSN algorithm proposed in document [12], the algorithm parameter in this paper is less, which reduces the impact of human operation on neighbor selection and makes it more concise and understandable.

4. Simulation and Analysis

In the following, we testify the effectiveness and accuracy of TWUNCF algorithm with simulations and exploring describe the adaptability to different size of data by using the proposed TWUNCF algorithm against to that by using *k*NN method to verify the proposed idea of dynamic selected neighbor is correct. Meanwhile, we try to analyze that whether the harmonic parameters' setting can lead to a better recommended result from the trustworthy subset. These are the two aspects to be verified in this section.

Similar to the test methods of other recommendation algorithm, this paper uses MovieLens dataset provided by Grouplens to simulate. MovieLens dataset contains 10 records. 943 users score on 1682 movies for five levels in total and the range of the score is 1~5. 1 point presents "poor", 5 is "perfect", and others means middle value. They present user interest in film in varying degrees. The hardware environment of the experiment is Intel (R) Core (TM) i5-25200 2.5GHz quad-core 64-bit CPU and 4GB of memory, the software environment is Windows 7 64bit operating system (professional) and all the codes are implemented with Java(64bit JDK) and Matlab2012.

The density of the score matrix of the user and the item is $\frac{100000}{943\times1682}=1.63\%$, which means the matrix is a sparse matrix. We divide the 943 users from dataset into 3 groups to test; each separately has 100 users, 200 users, 300 users. We select 70% sample data from the whole dataset as the train set, the other 30% as the test set to compare and verify.

4.1. Comparison of Dynamic Selection Method

In the experiments, we need to separately verify the method to select trustworthy subset and the one to recommend. First of all, we compare the DSN method – a method to select trustworthy subset - with the kNN method to test whether the proposed DSN method could successfully pick out the relatively good neighbor objects, and prepare for the following recommending. We set k as the abscissa and compare the performance of the 2 different methods in different neighbors, and its range is 1,2,4,8,10…60. Measure it by the MAE(Mean Absolute Error). The results are showed in Figures 2-4.

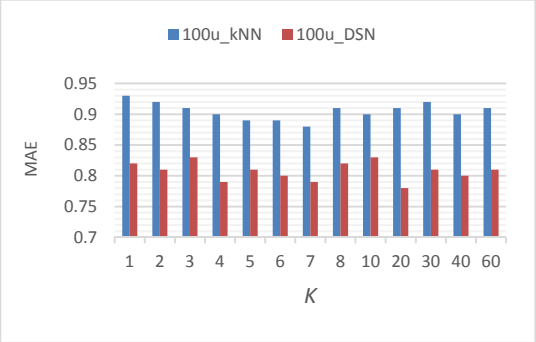

Figure 2. Comparison of kNN Method and DSN Method (100Users)

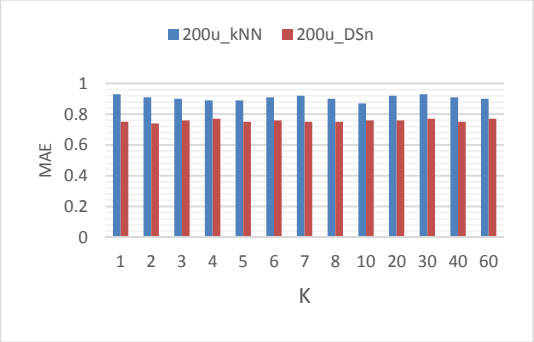

Figure 3. Comparison of kNN Method and DSN Method (200Users)

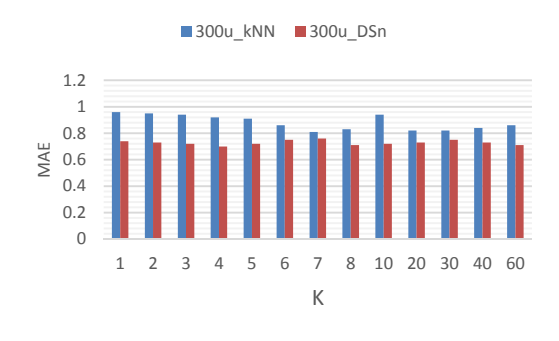

Figure 4. Comparison of kNN Method and DSN Method (300Users)

At first, compare the experimental results horizontally. We can see, when the number of user is 100, kNN has best results when k=7. However DSN has a better result in the same condition than kNN, and when k takes other values, DSN is still able to achieve better performance. Comparing the situation of 200 users to that of 300 users, the DSN has better performance than the kNN under the same conditions.

Secondly, compare the results of each group longitudinally. It's obvious that the bigger the train set is, the more favorable finding the target user's trustworthy subset and making a recommendation is. Through the analysis and comparison, we can find that DSN has better stability and accuracy than kNN under the same conditions.

The experiment shows the DSN method proposed in this paper is of positive significance for improving the method for determining the trust subgroup. Therefore, in the following experiments, we select the target object's trustworthy subset with DSN when recommending for UBCF and IBCF.

4.2. Comparative Experiments between the Traditional Recommendation Algorithm and the TWUNCF Algorithm in this Paper

The experiment compares the traditional collaborative filtering algorithm UBCF and IBCF to the TWUNCF algorithm proposed in this paper. The abscissa indicates the number of the predicted target item's neighbor, and the ordinate uses MAE as the metric.

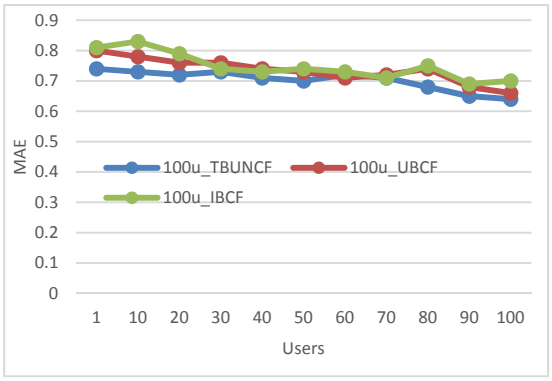

Figure 5. Comparison of IBCF、UBCF and TWUNCF (100Users)

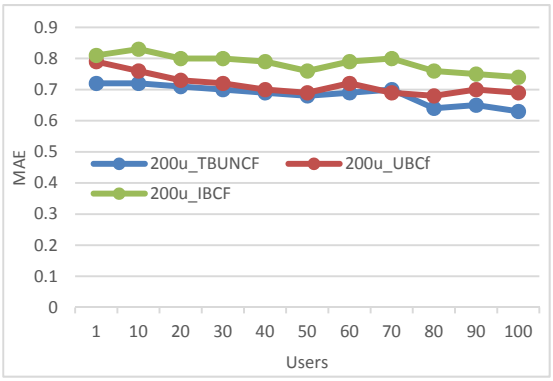

Figure 6. Comparison of IBCF, UBCF and TWUNCF (200Users)

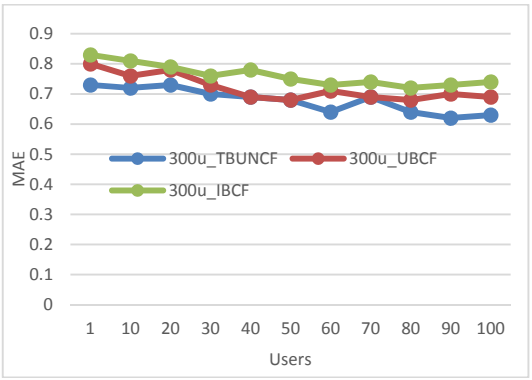

Figure 7. Comparison of IBCF, UBCF and TWUNCF (300Users)

Comparing and analyzing the three experiments above, we can find that TWUNCF can obtains the smaller value of MAE than IBCF and UBCF under the same conditions and it has relatively better recommendation when comparing each experiment horizontally. Comparing the three experiments vertically, it is obvious that the quality of recommendation increases with the

increasing number of the trustworthy subset. Analyzing all the experiments above comprehensively, we can come to an conclusion that TWUNCF has better performance that UBCF and IBCF under the same conditions.

5. Conclusion

Against the product-based and user-based prejudices in the traditional collaborative filtering algorithm and the characteristic of time-data validity, this paper proposed a Time Weighted Uncertain Neighborhood Collaboration Filtering Algorithm (TWUNCF). TWUNCF effectively solves the problem of time-data validity with logistic time function. On this basis, it proposes an idea of uncertain neighbors, and dynamically selects the trustworthy neighbors with both a user-based method and a product-based method, avoiding the uncertain accuracy of recommendation under different occasions caused by simply using product-based recommendation or user-based recommendation. Both the experimental and the theoretical analysis have proved the proposed algorithm TWUNCF has better accuracy and stability than the traditional algorithm TBCF and IBCF.

References

[1] Jelenc, David. Decision making matters: A better way to evaluate trust models. *Knowledge-Based Systems*. 2013; 52: 147-164
[2] Chen MC, Chen LS, Hsu FH, et al. HPRS: A profitability based recommender system. Martin Helander, Min Xie, Roger Jiao, et al. Proceeding of the IEEE International Conference on Industrial Engineering and Engineering Management. Singapore, IEEE Engineering Management Society Singapore Chapter. 2007(12): 219-223
[3] Abeer Mohamed El-korany, Salma Mokhtar Khatab. Ontology-based Social Recommender System. *IAES International Journal of Artificial Intelligence (IJ-AI).* 2012; 1(3): 127-138.
[4] Park ST, Pennock DM. *Applying collaborative filtering techniques to movie search for better ranking and browsing.* Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD 2007, San Jose, California, USA. ACM, New York 2007: 834–843.
[5] Tomoharu I, Kazumi S, Takeshi Y. *Modeling user behavior in recommender system based on maximum entropy.* Proceedings of the 16th International Conference on World Wide Web. Nigel T. Banff, Alberta, Canada, World Scientific Publishing Co.Pte.Ltd. 2007: 1281-1282
[6] Pappageorge, Charlotte. *Recommended verification approach for human rated systems based on lessons learned on the orion launch abort system*. 21st Annual International Symposium of the International Council on Systems Engineering. 2011; 1: 1-13.
[7] Kaklauskas, A. Recommended biometric stress management system. *Expert Systems with Applications*. 2011; 38(11): 14011-14025.
[8] Massa P, Avesani P. Trust-aware collaborative filtering for recommender systems*. Lecture Notes in Computer Science.* 2004; 3290: 492-508.
[9] Vincent SZ, Boi Faltings. *Using hierarchical clustering for learning the ontologies used in recommendation systems.* Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. San Jose, California, United States. 2007: 599-608.
[10] Muhammad Waseem Chughtai, Ali Bin Selamat, Imran Ghani. Goal-based hybrid filtering for user-to-user Personalized Recommendation. *International Journal of Electrical and Computer Engineering (IJECE).* 2013; 3(3): 329-336.
[11] GU Yi-ran, CHEN Min. One Tag Time-weighted Recommend Approach on Tripartite Graphs Networks. *Computer Science.* 2012; 39(8): 96-98,129.
[12] HUANG Chuang-Guang, YIN Jian, WANG Jing, et al. Uncertain Neighbors' Collaborative Filtering Recommendation Algorithm. *Chinese Journal of Computers.* 2010; 33(8): 1369-1377.
[13] CHEN Jian, YIN Jian. A Collaborative Filtering Recommendation Algorithm Based on Influence Sets. Journal of Software. 2007; 18(7): 1685-1694.
[14] Liu X, Datta A, Rzadca K, Lim EP. *Stereo Trust: A Group based per-sonalized trust model.* Proceedings of the 18th ACM conference on In-formation and Knowledge Management. Hong Kong, China. 2009: 7-16.
[15] Jamali M, Ester M, Trust Walker. *A random walk model for combining trust-based and item-based recommendation.* Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Paris, France. 2009: 397-406.
[16] LOU Jun-gang, SHEN Qing, SHEN Zhang-guo. Kernel Methods of Software Reliability Prediction. *Computer Science*. 2012; 39(4): 145-148.
[17] Ma H, King I, Lyu MR. *Effective missing data prediction for collaborative filtering.* Proceedings of the 30th Annual International ACM SIGIR Conference. Amsterdam, the Netherlands. 2007: 39-46.