2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

An Dynamic-weighted Collaborative Filtering Approach to Address Sparsity and Adaptivity Issues

Liang Gu, Peng Yang, Yongqiang Dong

School of Computer Science and Engineering, Southeast University, Nanjing 211189, China

Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, Nanjing 211189, China

pengyang@seu.edu.cn

***Abstract*— Recommendation systems, as efficient measures to handle the information overload and personalized service problems, have attracted considerable attention in research community. Collaborative filtering is one of the most successful techniques based on the user-item matrix in recommendation systems. Usually the matrix is extremely sparse due to the massive number of users and items. And the sparsity of users and items tends to differ significantly in degree. The feature of the matrix changes with the variation of users/items data and hence, leads to poor scalability of the recommendation method. This paper proposes a dynamic-weighted collaborative filtering approach (DWCF) to address sparsity and adaptivity issues. In this approach, the relationship between the distributions of similar users and items is considered to get better recommendation, i.e., the contributions of the user part and the item part to recommendation results depend on their similarity ratios. Moreover, the effect strength of different parts is controlled by an averaging parameter. Experiments on MovieLens dataset illustrate that the DWCF approach proposed in this paper can obtain good recommendation result given different conditions of data sparsity and perform better than a user-based predictor, an item-based predictor and a conventional hybrid approach.**

Keywords—recommendation; collaborative filtering; sparsity; adaptivity

I. INTRODUCTION

Recommendation systems have attracted research attention and become more and more efficient in E-commerce sites e.g. Amazon, IMDB, etc. Collaborative filtering and content-based recommending are two prevalent approaches to perform recommendations. Collaborative filtering recommends items by exploiting the similarity between users or items on the basis of the user's ratings of items. A new rating is predicted by averaging the ratings of similar users on the tested item or ratings of the tested user on similar items. Content-based methods make recommendations based on the features presented in items the user has rated. Collaborative filtering has some distinctive advantages over content-based methods. Firstly，it is more universal and easy to apply in different domains due to its independence of item content. In addition, collaborative filtering can help users to dig their new interests because the recommendation is made according to not only the historical interests of users, but also the relationship between users.

Traditional collaborative filtering exploits the similarity between users or items. Each method has its own advantages and disadvantages. Neighbors (similar users or items) with greater similarity are generally supposed to lead to more reliable and accurate recommendations. Therefore user-based methods are more suitable for a dataset in which the ratio of users with greater similarity is larger than that of items with greater similarity. Conversely, item-based methods perform better on the dataset in which the ratio of more similar items is larger. Both of these two methods suffer from the sparsity and adaptivity problems, i.e. they tend to have poor performance when the dataset changes or becomes extreme sparse. To alleviate these problems, some researchers have recently suggested combination methods based on a simple weighted technique [1]. However, the weighted parameter is generally a constant value. When used in real-world applications, in which the sparsity of dataset does not remain unchanged, this method becomes inefficient and has poor recommendation accuracy. The parameter can be estimated by extracting ratings randomly in order to make the method adaptive [2]. However, this estimation method ignores the sparsity feature of the whole dataset and randomly picking ratings will result in unstable recommendation results.

This paper focuses on the sparsity feature of the user-item matrix and proposes the effective similar ratio (ESR) to describe it as well as the DWCF approach to address sparsity and adaptivity problems using this parameter. Firstly, the similarity matrixes between users and items are constructed by using Pearson correlation coefficient (PCC) method. Then, both user-based recommendation and item-based recommendation are respectively computed according to the matrixes for the unknown tested rating (of a tested user on a tested item). The ESR is the proportion of users/items with more similarity in all similar users/items. And the more similarity part is filtered by setting an appropriate threshold. At the same time, a threshold of all similar users is also set in order to avoid massive data computation and improve the efficiency. After obtaining the ESR of users and items, the overall recommendation is calculated by averaging the user-based recommendation and item-based recommendation weighted by their corresponding ESR. Besides, an effect strength controlling parameter is naturally integrated into the new collaborative filtering approach. Finally, the experiments performed on MovieLens dataset and modified MovieLens dataset (with different ESRs of users and items) show that, DWCF approach outperforms other conventional collaborative filtering methods especially when the dataset is extreme sparse.

The remainder of this paper is organized as follows. Section II provides an overview of the related work. Section III introduces additional background information for the PCC method and two basic approaches, i.e., used-based method and item-based method. The framework of the DWCF approach is introduced detailed in Section IV. Section V evaluates the performance of DWCF via experiments performed on MovieLens dataset, followed by a conclusion in Section VI.

II. RELATED WORK

Collaborative filtering is one of the most successful techniques in recommendation systems based on previous ratings on the items by the users. [3] presents an overview of the field of recommendation systems including collaborative filtering. In this paper, collaborative filtering algorithms are grouped into two general classes: model-based and memory-based.

Model-based methods predict the ratings via training datasets with models like [4] [5] [6].The process to get the parameters in models needs complex computing and shows low efficiency in real-world applications. Unlike model-based methods, the rating to be predicted of is calculated as an aggregate of the ratings of others, e.g., ratings on similar items from similar users. Memory-based methods are generally sorted into the user-based and the item-based. User-based methods recommend the items preferred by other users similar to the tested user. [7] designs a system for collaborative filtering of netnews. All the readers are authorized to rate on the articles they read and the ratings are stored in a server. Then similarities between users are calculated and used to estimate their weights in predicting the unknown rating. To enhance the similarity computation, [8] suggests to optimize the weights of similar users via machine learning method. This approach obtains better recommendation results due to the modified similarity computation. However, [9] holds the opinion that item-based methods are more reliable than user-based methods and tries to reduce the computing complexity by filtering the users that have rated on both the items to be compared in similarity computation. A graph optimization approach to item-based collaborative filtering is proposed in [10]. This method gains improvement in recommendation accuracy than other methods. [11] proposes a hybrid graph method based on social tags to cluster users, webs and tags. The method proposed in this paper is effective in webpage recommendation.

In addition to the above user-based and item-based methods, several researchers attempt to combine these two methods to overcome their own defects. [12] re-analyzes the memory-based method from the perspective of generative probabilistic framework. The work takes into account the effect of user similarity and item similarity and determines the effect arguments via conditional probability. By this method, the data sparsity problem is alleviated to some extent and the recommendation performance gets enhanced. [1] adopts a modified PCC method to compute user and item similarities and predict the missing ratings to mitigate the sparsity problem when making recommendations. Compared to traditional collaborative filtering, this approach avoids unnecessary computations and hence raises recommendation efficiency. Different from the two simple weighted methods mentioned above, [2] tries to make the weight adaptive to different datasets by estimating it based on randomly extracting one rating from previously collected ratings. Whereas the estimation method ignores the sparsity feature of the whole dataset and random picking ratings will result in unstable and gross recommendation results. The approach proposed in our paper deals well with sparse dataset and possesses good adaptivity with consideration of the feature of the whole dataset.

III. BACKGROUND

This section briefly introduces the user and item similarity computation using the PCC method and the user-based and item-based methods. All of this section is based on the user-item matrix. The matrix contains the information of user, item and the user's rating on the item. *M* and *N* represent the number of total users and items respectively. Generally, the row vector represents one user's ratings on different items and the column vector represents all the users' ratings on one item. Every entry in the matrix represents the rating value $g_{m,j}$, from user *m* on item *j*. The entry remains null when the user has not rated on the correspondent item.

A. Similarity Computation Using PCC

User-based methods compute the similarity between the tested user and others based on their previous ratings on all items. According to the user-item matrix above, the computation in user-based methods uses the PCC method between each row, e.g., user *a* and user *b* as follows:

$$S(a,b) = \frac{\sum_{i \in I_a \cap I_b} (g_{ai} - \overline{g_a})(g_{bi} - \overline{g_b})}{\sqrt{\sum_{i \in I_a \cap I_b} (g_{ai} - \overline{g_a})^2} \sqrt{\sum_{i \in I_a \cap I_b} (g_{bi} - \overline{g_b})^2}} \tag{1}$$

$\overline{g_a}$ and $\overline{g_b}$ represent the average ratings of user *a* and *b* on all the items they rated respectively. $I_a \cap I_b$ denotes the intersection of the items that user *a* and *b* have both rated. When $I_a \cap I_b = \Phi$, $S(a,b)$ equals zero.

Item-based methods compute the item similarity based on the user-item matrix. Similar to the user similarity, the item similarity can be computed as follows:

$$S(m,n) = \frac{\sum_{u \in U_m \cap U_n} (g_{um} - \overline{g_m})(g_{un} - \overline{g_n})}{\sqrt{\sum_{u \in U_m \cap U_n} (g_{um} - \overline{g_m})^2} \sqrt{\sum_{u \in U_m \cap U_n} (g_{un} - \overline{g_n})^2}} \tag{2}$$

Average ratings of item *m* and *n* are taken over all the users that have rated on the two items. And $U_m \cap U_n$ represents the intersection of all the users that have ratings on item *m* and *n*.

B. User-based Collaborative Filtering

Existing user-based collaborative filtering methods share the basic similarity computation as the first step to make recommendations. All the other users are then sorted by their similarities to the tested user. Based on the fact that high-similarity users have more positive effect on the recommendation than low-similarity ones, a threshold value is set to filter the low-similarity users in advance. Besides, in user-based collaborative method, the more similarity the users possess, the stronger weight is assigned to them. The detailed weighting strategy is as follows:

$$g_{am,u} = \overline{g_a} + \frac{\sum_{u \in U_{a,\delta}} S(a,u)(g_{um} - \overline{g_u})}{\sum_{u \in U_{a,\delta}} S(a,u)} \quad (3)$$

$U_{a,\delta}$ in (3) is the set of users similar to user *a* filtered with similarity threshold δ. $S(a, u)$ is computed according to (1). Specially, when there are no similar users having similarity greater than δ, the g_{am} equals the average rating of user *a*.

Note that, similarity in (3) is the only consideration in weight computation. Though in fact, the heterogeneity of similarities of selected users have strong effects on recommendation accuracy.

C. Item-based Collaborative Filtering

Different from user-based collaborative filtering, item-based methods utilize the relationship between items. The methods calculate item similarities and attempt to find the ratings on these similar items from the tested user. Similar to user-based methods, item-based methods sort the similarities after filtering and besides, higher similarities contribute more in recommendation. The prediction of ratings based on items is combined from the ratings of similar item as follows:

$$g_{am,i} = \overline{g_m} + \frac{\sum_{i \in I_{m\delta}} S(m,i)(g_{ai} - \overline{g_i})}{\sum_{i \in I_{m\delta}} S(m,i)} \quad (4)$$

$I_{m\delta}$ is the set of items similar to item *i* with similarity larger than δ. $S(m, i)$ is calculated via (2). Likewise, $I_{m\delta}$ is possibly an empty set due to the dataset sparsity and threshold δ. Finally, the rating of user *a* on item *m* is accumulated by two parts: the average rating of item *m* from all the other rated users and the regulation part affected by similar items.

IV. DWCF APPROACH

As mentioned above, collaborative filtering is based on the user-item matrix. In real-world commercial applications, this matrix is fairly sparse due to the massive users and items in general. The similarity computation is completely dependent on the matrix. Therefore, the recommendation accuracy will decay considerably as the density of matrix decreases in traditional collaborative filtering methods. Some work tries to deal with the sparsity problem with several methods. [12] proposes a combinative framework considering the similarities of both users and items. The integration strategy enables the method to obtain better recommendation results than simple user-based and item-based approaches. In [12], however, the weight between users and items is a constant. When the dataset changes, it is almost impossible to maintain an accurate recommendation. Furthermore, [2] proposes a weight parameter estimation method based on randomly extracting one rating from previously collected ratings. Although the randomly picking ignores the sparsity feature of the whole dataset and hence results in unstable and gross recommendation results.

Based on the above discussion and analysis, this paper proposes an adaptive collaborative filtering method which combines user-based and item-based method and adjusts dynamically the weight between them according to the density feature of the matrix. This adaptive combination method bases itself on the assumption that a high percentage of more similar neighbors prevails over a low one. And the assumption will be proven via experiments on the dataset with different density features. In this approach, a novel density estimation method with a pair of parameters to depict the density feature of the dataset more reasonably is also addressed (as shown in Algorithm 1). In this method, the prediction is based on the similarity of users and items. And due to this, the time complexity of this method is O ($M*N$), which is the same with other methods of comparison involved in Section V. Over here, *M* represents the total number of users and *N* represents the total number of items.

Algorithm 1: DWCF Approach

Input : user-item rating matrix($M*N$), user *u*, item *i*

Output : the prediction of the rating on *i* from *u*

1. **For** $k = \{1,\ldots,M\}$, k != u
   Compute the PCC similarity $S(m, k)$ between *u* and *k*
2. **For** $j = \{1,\ldots,N\}$, j != i
   Compute the PCC similarity $S(i, j)$ between *i* and *j*
3. Filter the users and items with low similarity less than δ and μ respectively
4. Compute the proportion of more similar neighbors via dividing the result of (3) with δ by the result with μ
5. Calculate the prediction of *u*'s rating on *i* with user-based CF and item-based CF respectively
6. Compute the weight of the two different CF methods when generating the final prediction with the controlling parameter *s*
7. Generate the final prediction result of *u*'s rating on *i* with the result of (5) and (6)

The remainder of this section gives a detailed formulation of the proposed DWCF approach including similar neighbors selection, density computation, the combination method and parameters discussion.

A. Similar Neighbors Selection and Ratios Computation

1) Similar users selection: By (1), the similar users matrix is created and the result is sorted in descending order constricted by the parameter δ as:

$$U_{a,\delta} = \{x_i \mid S(a, x_i) > \delta,\ S(a, x_i) > S(a, x_{i+1})\} \quad (5)$$

where $U_{a,\delta}$ is the similar users set of user *a* and all the users in this set have the similarity with user *a* bigger than δ. The matrix of $U_{a,\delta}$ is illustrated in Fig. 1.

2) Similar items selection: Like the users selection, items selection is also a critical step in DWCF approach. The similar items are selected with a suitable threshold δ as:

$$I_{m,\delta} = \{y_i \mid S(m, y_i) > \delta,\ S(m,y_i) > S(m,y_{i+1})\} \quad (6)$$

With (6), items information is fully considered to benefit the recommendation result. And Fig.2 shows the result of the computing of $I_{m,\delta}$.

TABLE I. SIMILAR USERS OF THE TESTED USER SORTED BY SIMILARITIES IN TWO DIFFERENT CONDITIONS

User	1	2	3	4	5	6	7	8	9	10
Similarity set 1	0. 89345	0. 86837	0. 85674	0. 83179	0. 80864	0. 78413	0. 75911	0. 70114	0. 65	0. 64102
Similarity set 2	0. 88142	0. 81071	0. 75741	0. 74912	0. 69417	0. 68873	0. 66912	0. 64871	0. 62871	0. 61503
Rating	4	5	4	2	3	2	4	5	3	4

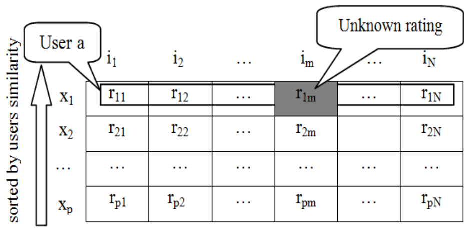


Fig. 1. Matrix of $U_{a,\delta}$ obtained by sorting the similarity of users in ascending order.

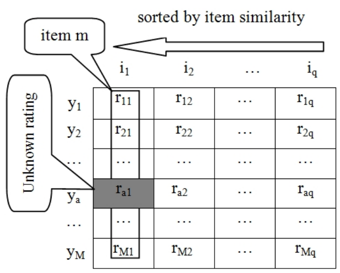


Fig. 2. Matrix of $I_{m,\eta}$ obtained by sorting the similarity of items in descending order.

3) *Ratios computation:* As the like-minded users and the similar items have been preliminary computed and sorted, the next step of DWCF is to calculate the density feature of the matrix. To achieve this, this paper sets another parameter μ to pick the neighbors with more similarities to the tested user or item. Then, it is necessary to figure out their proportion in all the selected neighbors. This idea bases itself on the fact that collaborative filtering with same filtering threshold differs considerably in recommendation results when the proportion of neighbors with more similarities is different. Such condition can be illustrated figuratively by the example extracted from the MovieLens Dataset[1] as Table I illustrates. In this example, the tested user has ten similar users after the similarity computation. The real rating of the tested user on tested item is 4. All the ratings are integers and range from 1 to 5. As the table shows, we suppose the selected users have two sets of similarities with different distributions. The proportion of more similar (e.g., set to 0.74) users of set 1 is larger than set 2. Based on these similarities, the predicted ratings of the test user on the test item are 3.59959 and 3.31453 respectively. Obviously, the results are different and the former is better than the latter one considering the real rating is 4. $U_{a,\delta}$ and $U_{a,\mu}$ can be obtained by applying δ and μ in (5). And with (6) the corresponding similar number of items can also be figured out. Ultimately, the proportion of more similar neighbors can be computed as:

$$p_{u,a} = \frac{|U_{a,\mu}|}{|U_{a,\delta}|} \tag{7}$$

and

$$p_{i,m} = \frac{|I_{m,\mu}|}{|I_{m,\delta}|} \tag{8}$$

where the absolute value symbol represents the calculation of the corresponding member number of the set.

B. Combination Method

The next step is to combine the two basic approaches with the filtering thresholds mentioned above. In this paper, a dynamic weighting strategy is introduced considering user information, item information, and their effects on combination result. And the weight estimation is based on the whole dataset and hence leads to stable and higher accuracy.

$$G_{(a,m)} = \frac{p_{u,a}^{s}}{p_{u,a}^{s} + p_{i,m}^{s}} g_{am,u} + \frac{p_{i,m}^{s}}{p_{u,a}^{s} + p_{i,m}^{s}} g_{am,i} \tag{9}$$

In (9), $g_{am,u}$ represents the recommendation based on the like-mind users of user *a* and $g_{am,i}$ represents the recommendation based on the similar items of item *m*. Moreover, *s* is an adjusting parameter introduced to control the allocation of the recommendations from different sources.

C. Parameters Discussion

1) Filtering thresholds: In user-based part, δ represents the similar users threshold and μ which is larger than δ, represents the more effective users in all the similar users. To get a better recommendation result, it is necessary to set up the two parameters properly. On one hand, if δ is set to a small value, a lot of less similar users will be considered while this will result in unnecessary computations, not mention to that these users may have a negative effect on recommendation accuracy. On the other hand, if δ is set to a large value, many similar users will be overlooked which have a positive effect on recommendation accuracy. To the more effective users parameter μ, no matter its value is too small or too large, it will always have poor influence on the combination process. The condition of the item-based part is analogous to user-based part.

2) Weight controlling parameter: $p_{u,a}$ and $p_{i,m}$ represent the proportions of the more similar users and items respectively in all the selected neighbors. With these two parameters, it is convenient to assess the sparsity feature of the dataset and make the most of the data. The adjusting parameter *s* which is a

[1] http://www.cs.umn.edu/Research/GroupLens/.

positive real number, determines the impacts of user similarity and item similarity on recommendation results, as follows:

- $s = 0$. On this occasion, $p_{u,a}{}^{s}$ and $p_{i,m}{}^{s}$ both equal 1. And it is a simple case that the user similarity equals to the item similarity in the aspect of effects. The combination rating is the average of $g_{am,u}$ and $g_{am,i}$.
- $0 < s < 1$. In this case, the effect of the larger value between $p_{u,a}{}^{s}$ and $p_{i,m}{}^{s}$ on recommendation result is weakened.
- $s = 1$. In this case, the effects of $p_{u,a}{}^{s}$ and $p_{i,m}{}^{s}$ are exactly proportional to their values.
- $s > 1$. When s is bigger than 1, the effect of the larger value between $p_{u,a}{}^{s}$ and $p_{i,m}{}^{s}$ on recommendation result gets stronger as s increases. As a special case, if s tends to positive infinity, the recommendation result totally depends on the larger one between $p_{u,a}$ and $p_{i,m}$.

V. Empirical Analysis

In this section, we conduct several experiments to evaluate our proposed approach, and address the experiments as the following questions: (1) What is the relationship between the matrix density feature and recommendation accuracy? (2) How do the filtering thresholds affect the recommendation accuracy? (3) How does the controlling parameter affect the recommendation accuracy? (4) How does our approach DWCF compare with traditional user-based methods, item-based methods and other combining methods proposed recently?

A. Dataset

The experiments are performed on the MovieLens dataset, which is a famous dataset for collaborative filtering. The dataset contains 100,000 five-grade ratings on 1682 items by 943 users, and each user at least rated 20 movies. We extract Subset a and Subset b with different ESRs of users or items from the dataset by deleting some ratings. Note that, we regard the neighbors with similarity larger than 0.5 as effective considering that the similarity ranges from 0 to 1 certainly. The statistical features of the MovieLens dataset and the subsets are summarized in Table II.

B. Metrics

Mean absolute error (MAE) is the most common method to measure the recommendation accuracy, which is defined as:

$$MAE = \frac{\sum_{u,i} |\ g_{u,i} - g'_{u,i}\ |}{N} \qquad (10)$$

where $g_{u,i}$ denotes the real rating on item i from user u, $g'_{u,i}$ denotes the predicted rating and N represents the total number of all the tested ratings.

C. Accuracy experiments

1) Comparisons under different ESRs: We first conduct predicting experiments of user-based and item-based methods under different ESRs to reveal the relationship between the dataset feature and the recommendation accuracy. With the changes of ESR, the recommendation result shows noticeable

TABLE II. Statistical Features of Different Datasets

Dataset	Statistics	User	Item
Original dataset	Min. Num. of Ratings	20	1
	Avg. Num. of Ratings	106.04	59.45
	Effective Similar Ratio	0.38	0.22
Subset a	Min. Num. of Ratings	17	20
	Avg. Num. of Ratings	100.71	101.12
	Effective Similar Ratio	0.37	0.36
Subset b	Min. Num. of Ratings	11	50
	Avg. Num. of Ratings	88.775	138.83
	Effective Similar Ratio	0.32	0.41

variation, as illustrated in Fig. 3. In the original dataset, the user-based method outperforms than the item-based method in recommendation accuracy. This is chiefly because the ESR of users is larger than that of items i.e. the similar users used to generate the recommendation provide information more reliable than the similar items. Moreover, when the ESR of items gets higher and the ESR of users gets lower, the MAE of user-based method goes up while the MAE of item-based method goes down. Based on the experimental result, it can be inferred that the user-based method and the item-based method perform well under different ESRs and combining the two methods according to this feature properly may result in better recommendation accuracy.

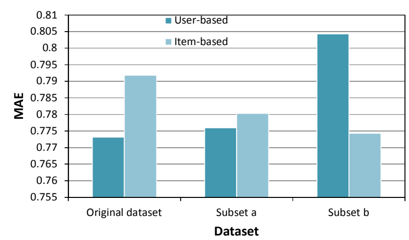


Fig. 3. Plots over MAE of user-based and item-based methods with different datasets

2) Impacts of filtering thresholds: In this paper, a pair of filtering thresholds is introduced to determine the similar neighbor's distribution and balance the information from users and items. The parameter δ is used to filtering the neighbors with low similarities in order to improve the efficiency and accuracy while μ is used to get more similar neighbors in the filtered results. Generally speaking, μ is bigger than δ. In order to cover all the conditions, we choose three representative values for δ and μ: 0.3, 0.5 and 0.7. When the threshold equals 0.3, it means that there are a lot of neighbors to refer to though a considerable portion of them are less similar to the test user or item. While the value 0.7 means that, neighbors number is small but their similarities are reasonably high. Obviously the value 0.5 has an influence on filtering neighbors between 0.3 and 0.7. We perform the experiments on MovieLens dataset with the weight controlling parameter s set at different values and obtain similar curves. One of the results (s set at 1.7) is showed in Table III and Table IV.

TABLE III. IMPACTS OF μ ON MAE WHEN δ IS FIXED AT 0.3, 0.5 AND 0.7 RESPECTIVELY. S IS SET AT 1.7.

δ =0.3	μ	0.35	0.4	0.45	0.5	0.55	0.6	0.65	0.7	0.75	0. 8
	MAE	0.78735	0.78682	0.78503	0.78611	0.78633	0.78652	0.78682	0.78711	0.78727	0.78742
δ =0.5	μ	0.53	0.56	0.59	0.62	0.65	0.68	0.71	0.74	0.77	0.8
	MAE	0.74727	0.74729	0.7472	0.74715	0.74703	0.74697	0.74567	0.7483	0.74933	0.75141
δ =0.7	μ	0.72	0.74	0.76	0.78	0.8	0.82	0.84	0.86	0.88	0.9
	MAE	0.77842	0.77837	0.77847	0.77835	0.7783	0.77843	0.77848	0.78012	0.78013	0.78087

TABLE IV. IMPACTS OF δ ON MAE WHEN μ IS FIXED AT 0.3, 0.5 AND 0.7 RESPECTIVELY. S IS SET AT 1.7.

μ =0.3	δ	0.1	0.12	0.14	0.16	0.18	0.2	0.22	0.24	0.26	0. 28
	MAE	0.84753	0.84741	0.84749	0.84751	0.84735	0.84723	0.84741	0.84735	0.84762	0.84754
μ =0.5	δ	0.2	0.23	0.26	0.29	0.32	0.35	0.38	0.41	0.44	0.47
	MAE	0.78712	0.78711	0.78691	0.78611	0.78604	0.78599	0.78681	0.78697	0.78738	0.78761
μ =0.7	δ	0.3	0.34	0.38	0.42	0.46	0.5	0.54	0.58	0.62	0.66
	MAE	0.78711	0.77595	0.76742	0.75209	0.74673	0.74689	0.74715	0.74726	0.74801	0.74835

Firstly, we fix δ at 0.3, 0.5, 0.7 respectively and test the influence of μ on recommendation accuracy as Table III illustrates. When δ is fixed at 0.3, the MAE of recommendation goes down with the increase of μ and reaches the minimum when μ equals 0.46. After that it gets larger again as μ grows. This surprising result is caused by the mixing effect of the number and similarity of similar neighbors. When μ is small, lots of less similar neighbors are considered into recommendation although they tend to have somewhat negative effects on recommendation accuracy and obviously this will result in a relatively high MAE (poor accuracy). Conversely, as μ get bigger, the neighbors used in recommendation will have positive effects on recommendation accuracy and lead to a lower MAE. However, the number of used neighbors will also reduce as μ becomes larger and make the recommendation unreliable. That's why the MAE gets larger when μ is big enough. Furthermore, at the point where μ equals 0.46, the recommendation obtains balance between the number and similarities of neighbors and thus produces an optimal MAE.

When δ is fixed at 0.5, as Table III shows, the recommendation accuracy does not change too much as μ grows until it reaches 0.72. While μ is larger than 0.72, the MAE stays at a lower level for a little while and gets higher later. Analogous to the condition when δ equals 0.3, the balance between the number and similarities of neighbors is acquired at 0.72.

Table III also plots the influence of μ when δ is fixed at 0.7, a relatively high threshold. Under this circumstance, the accuracy gets poorer after keeping on a plateau for some time. The result further suggests that when the threshold to filter less similar neighbors is set too high, the neighbors available for recommendation reduce distinctly and therefore leads to poor recommendation accuracy.

Next, we get μ fixed at 0.3, 0.5 and 0.7 separately. The relationship of different values of δ and recommendation accuracy is illustrated in Table IV. According to Table IV, we can firstly observe that when δ and μ are both less than or equal 0.3, the recommendation accuracy always stays at a very low level. Also we can obtain that when μ is fixed at 0.7, low value of δ will lead to low accuracy and the accuracy accelerates and tends towards stability as δ increases. Besides that Table IV shows the condition that the parameter μ is fixed at 0.5. In this condition, the balance between the numbers and similarities of neighbors is obtained when δ arrives at 0.36 and the MAE of the recommendation is lower than other values.

We can conclude from the experiments in this section that, filtering thresholds δ and μ both have strong effects on recommendation accuracy. On one side, if they are set too low, a lot of neighbors to be used in recommendation can be obtained. However, they tend to have low similarities to the predicted user or item and hence result in poor recommendation performance. On the other side, if they are set too high, the available neighbors will reduce dramatically and we also cannot obtain satisfied results. Therefore, to get the expected recommendation accuracy, the thresholds should be adjusted at proper value to get the balance between the two influence factors mentioned above. On the dataset used in this article, the recommendation accuracy reaches the optimum value when δ is fixed at 0.5 and μ is fixed at 0.72.

3) *Impact of s:* As introduced in Section IV, s also plays a comparably important role in our collaborative filtering approach. s directly determines the impacts of the user similarity and the item similarity on recommendation results. In this section, we perform experiments to discover the impact of s on recommendation accuracy with δ fixed at 0.5 and μ fixed at 0.72. To get a comprehensive result, we perform experiments on three datasets: Original Dataset, Subset a and Subset b. In Original Dataset, the ESR of users is larger than the ESR of items. While in Subset b, the condition is just the opposite.

The two ESRs rival with each other in Subset a. Besides that, Subset b has the most average of ratings while Original dataset possesses the least ones as Table II shows. The results are showed in Fig. 4. Observed from Fig. 4, we can draw the conclusion that the value of s has a considerable impact on

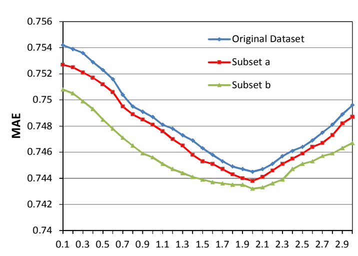
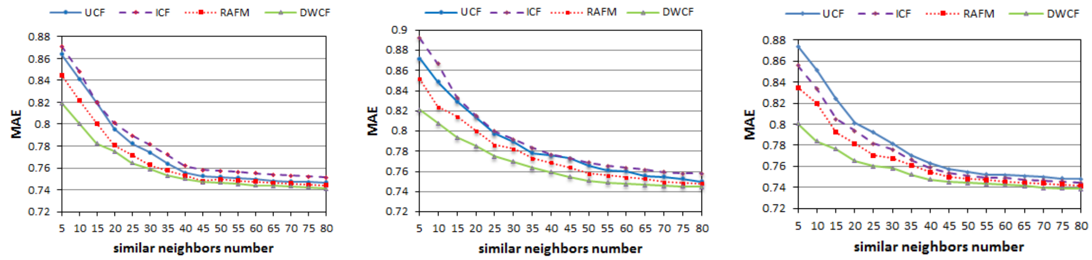


Fig. 4. Plots of MAE in different datasets with different s



(a) Original Dataset (b) Subset a (c) Subset b

Fig. 5. Plots of MAE with different methods in different datasets. The left plot is the result of the original dataset. The middle plot is the result of Subset a and the right one is the result of Subset b.

recommendation results. The accurate recommendation can be obtained around the value 2 on all the datasets. This is mainly because around here, the approach gives a proper weight to both of the user part and the item part considering their similarities with the tested user or item. Another noticeable observation is that the recommendation method works better in Subset b than the other two datasets. This is mainly due to the fact that there are more average available neighbors in Subset b as table II describes.

4) *Comparisons with other methods:* In this section, we compare DWCF method with some traditional methods (the user-based method [12] and the item-based method [5]) and another hybrid method (the random adaptive fusion method [2]). To show the superiority of our proposed method over other methods in alleviating sparsity and adaptivity problems, we make comparisons under different sparsity (given different numbers of similar neighbors) and different ESRs (Original Dataset, Subset a and Subset b). Note that the given similar neighbors are obtained via removing some ones randomly from all the similar neighbors. Fig. 5 presents the experimental results.

According to Fig. 5(a), Fig. 5(b) and Fig. 5(c), we can draw conclusions as follows. When the ESR of users is larger than that of items, the user-based method (UCF) performs better than item-based method (ICF). And ICF performs better in the converse condition. The random adaptive fusion method (RAFM) always performs better than UCF and ICF due to its adaptive strategy. What's more, the DWCF method proposed in this article obtains the most accurate result in all the datasets with different sparsity features. Last but not least, the superiority becomes outstanding when the data is sparse.

VI. CONCLUSIONS

We propose a dynamic-weighted collaborative filtering approach DWCF to address sparsity and adaptivity problems. DWCF evaluates the data feature of the dataset via setting a pair of filtering thresholds and makes the best of the data according to the feature. Furthermore, a weight controlling parameter is introduced to determine the impacts of user similarity and item similarity. Experiments show that, DWCF outperforms other methods in accuracy and adaptivity especially in the sparse condition. In the future, we plan to analyze the approach more formally and attempt to obtain better results.

ACKNOWLEDGMENT

This work was supported by National High Technology Research and Development Program (863 Program) of China under grant no. 2013AA013503, CAE Consulting Project under grant 2013-XY-6, and National Natural Science Foundation of China under grant no. 61370206.

REFERENCES

[1] Ma H, King I, Lyu M R. Effective missing data prediction for collaborative filtering[C]//Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, 2007: 39-46.

[2] Yamashita A, Kawamura H, Suzuki K. Adaptive fusion method for user-based and item-based collaborative filtering[J]. Advances in Complex Systems, 2011, 14(02): 133-149.

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

[4] Yang X, Guo Y, Liu Y. Bayesian-Inference-Based Recommendation in Online Social Networks[J]. IEEE Transactions on Parallel and Distributed Systems, 2013, 24(4): 642-651.

[5] Sarwar B, Karypis G, Konstan J, et al. Item-based collaborative filtering recommendation algorithms[C]//Proceedings of the 10th International Conference on World Wide Web. ACM, 2001: 285-295.

[6] Sahoo N, Singh P V, Mukhopadhyay T. A Hidden Markov Model for Collaborative Filtering[J]. MIS Quarterly, 2012, 36(4).

[7] Resnick P, Iacovou N, Suchak M, et al. GroupLens: an open architecture for collaborative filtering of netnews[C]//Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work. ACM, 1994: 175-186.

[8] Jin R, Chai J Y, Si L. An automatic weighting scheme for collaborative filtering[C]//Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, 2004: 337-344.

[9] Sarwar B, Karypis G, Konstan J, et al. Item-based collaborative filtering recommendation algorithms[C]//Proceedings of the 10th International Conference on World Wide Web. ACM, 2001: 285-295.

[10] Rostami B, Cremonesi P, Malucelli F. A Graph Optimization Approach to Item-Based Collaborative Filtering[M]//Recent Advances in Computational Optimization. Springer International Publishing, 2013: 15-30.

[11] Li H Q, Xia F, Zeng D, et al. Exploring social annotations with the application to web page recommendation[J]. Journal of Computer Science and Technology, 2009, 24(6): 1028-1034.

[12] Wang J, De Vries A P, Reinders M J T. Unifying user-based and item-based collaborative filtering approaches by similarity fusion[C]//Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, 2006: 501-508.