Mobile Netw Appl (2015) 20:380–390
DOI 10.1007/s11036-014-0560-5

CFSF: On Cloud-Based Recommendation for Large-Scale E-commerce

Long Hu · Kai Lin · Mohammad Mehedi Hassan · Atif Alamri · Abdulhameed Alelaiwi

Published online: 30 January 2015

Abstract Recommender systems assist the e-commerce providers for services computing in aggregating user profiles and making suggestions tailored to user interests from large-scale data. This is mainly achieved by two primary schemes, i.e., memory-based collaborative filtering and model-based collaborative filtering. The former scheme predicts user interests over the entire large-scale data records and thus are less scalable. The latter scheme is often unsatisfactory in recommendation accuracy. In this paper, we propose Large-scale E-commerce Recommendation Using Smoothing and Fusion (CFSF) for e-commerce providers. CFSF is divided into an offline phase and an online phase. During the offline phase, CFSF creates a global item similarity matrix (GIS) and user clusters, where user ratings within each cluster is smoothed. In the online phase, when a recommendation needs to be made, CFSF dynamically constructs a locally-reduced item-user matrix for the active user item by selecting the top M similar items from GIS and top the K like-minded users from user clusters. Our empirical study shows that CFSF outperforms existing CF approaches in terms of recommendation accuracy and scalability.

Keywords Large-scale data · Recommender systems · Collaborative filtering · Clustering · Fusing · Smoothing

1 Introduction

The large-scale E-commerce providers such as Amazon[1], Taobao[2] and Netflix[3] have aggregated a huge amount of data about user profiles and their purchase and browse records. Estimated by Piper Jaffray's Gene Munster, Amazon had over 121 million customers and over 5 million prime members by the end of 2011. Most existing E-commerce providers have adopted Recommender Systems (RSs) as a salient part of their websites to deliver automatically generated customized recommendations to their users which supports users in their decision making while interacting with large information spaces. These recommendations are mainly achieved by Collaborative Filtering (CF), which refers to a process of aggregating user profiles and predicting user interests from large-scale purchase and browse records [1]. The underlying principle of CF is that a user may be interested in the items that are preferred by users who share similar tastes and in items that are similar to his favorite items.

L. Hu (✉)
School of Computer Science and Technology,
Huazhong University of Science and Technology, Wuhan, China
e-mail: longhu.cs@gmail.com

K. Lin (✉)
School of Computer Science and Engineering,
Dalian University of Technology, Dalian, China
e-mail: link@dlut.edu.cn

M. M. Hassan · A. Alamri · A. Alelaiwi
College of Computer and Information Sciences, Chair of Pervasive and Mobile Computing, King Saud University,
P.O. Box 51178, Riyadh 11543, Saudi Arabia

M. M. Hassan
e-mail: mmhassan@ksu.edu.sa

A. Alamri
e-mail: atif@ksu.edu.sa

A. Alelaiwi
e-mail: aalelaiwi@ksu.edu.sa

[1] http://www.amazon.com/
[2] http://www.taobao.com/
[3] http://www.netflix.com/

CF has achieved increasing success for service computing and cloud computing, but it often suffers from two primary problems — data sparsity and limited scalability [2, 3]. The former problem is that the rated data made by users are sparse, e.g., less than 1 % in commercial recommender systems [4]. The latter problem denotes that CF approaches are limited by scalability, because the size of item-user matrix in recommender systems is quite large, involving millions of items and users [5].

Generally, CF can be classified into two categories — memory-based and model-based approaches. Memory-based CF approaches often achieve high levels of accuracy by exploiting similarities among items and users, but they are unable to scale up [6, 7]. They predict user interest by identifying similar items or like-minded users over the entire item-user matrix. That is, they require a process to search items and users over the entire item-user matrix. With the increasing number of items and users, memory-based approaches become less and less scalable. Moreover, the universal inconsistencies in the subjectivity of user ratings (also known as the "global effect") significantly affect the accuracy for recommendations in service computing [8]. In contrast, model-based approaches obtain good scalability but relatively poor accuracy. They employ various mechanisms, e.g., social networks [9] and Bayesian networks [10], to cluster items or users into classes. Then, they predict unrated data by selecting a few classes most related to the active user. Model-based approaches narrow down the search space for similar items or like-minded users, thus accelerate the process of predicting user interests with cloud computing technology. As a result, model-based approaches are more scalable than memory-based ones. Nevertheless, the accuracy for model-based approaches is limited by data sparsity. In addition, the ratings that like-minded users made on similar items, the same user made on similar items, and like-minded users made on the same item have diverse impact on the recommendation. But most model-based CF approaches do not differentiate them, which remarkably reduces their recommendation accuracy.

In this paper, we propose CFSF – Large-scale E-commerce Recommendation Using Smoothing and Fusion for e-commerce providers to make services more effective. CFSF formulates the CF problem over the entire item-user matrix in recommender systems as a small-scale prediction problem over a locally-reduced item-user matrix. CFSF divides recommendation into an offline phase and an online phase. In the offline phase, CFSF creates a global item similarity matrix and uses *KNN* clustering algorithm to cluster users. Within each user cluster, CFSF employs a smoothing strategy to smooth user ratings. In the online phase, CFSF identifies the active item and user according to the recommendation request. Then, CFSF dynamically constructs a locally-reduced item-user matrix for the active items and the active users by selecting the top M similar items from the global item similarity matrix and top the K like-minded users from user clusters. Finally, CFSF predicts user interest over the locally-reduced item-user matrix by fusing the ratings of the same user on similar items, and that like-minded users made on the same and similar items. The smoothing and fusing strategies, together with the locally-reduced item-user matrix, contribute to the accuracy and efficacy of CFSF. Empirical studies over the two well-known big data sets also show that CFSF efficiently addresses the two primary problems of CF — data sparsity and scalability.

An earlier version of the approach with a limited evaluation has been presented previously in a symposium paper [11]. We have extended that paper with additional algorithm details, theoretical analysis and a thorough performance evaluation over two well-known big data sets. We have also incorporated the latest related research and distinguish our work from them.

The rest of this paper is organized as follows. Section 2 introduces the background of CF. Section 3 describes the proposed approach in detail. Section 4 reports the experimental results and Section 5 concludes our work with future directions.

2 Background

Figure 1 illustrates user profiles that are represented as a $Q \times P$ item-user matrix X, where Q and P are the sizes of items and users. CF aims to predict the rating of active item i_a made by active user u_b from user profiles. In general, there are two characteristics in such a large number of user profiles. One is that rating data is quite sparse, which raises the data sparsity problem for CF (i.e., how to get a high level of recommendation accuracy from such sparse rating data). The other is that the size of user profiles is large, which causes the scalability problem for CF (i.e., how to efficiently predict user interest over the large-scale user profiles). In order to formally describe the background of CF, we introduce a set of notations. Let

- $\mathcal{I} = \{i_1, i_2, \ldots, i_Q\}$ and $\mathcal{U} = \{u_1, u_2, \ldots, u_P\}$ be the sets of items and users in X,
- $\{C_u^1, C_u^1, \ldots, C_u^L\}$ be L user clusters, and users in each cluster share some similar tastes (i.e., rate similar items in a similar way),
- $I\{u\}$, $I\{C_u\}$ and $U\{i\}$ be the set of items rated by user u, the set of items rated by user cluster C_u, and the set of users who have rated item i,
- r_{u_b,i_a} denote the score that user u_b rates item i_a, $\overline{r_{i_a}}$ and $\overline{r_{u_b}}$ represent the average ratings of item i_a and user u_b,
- SI, SU and SUI be the sets of the similar items, like-minded users, and similar items and like-minded users,

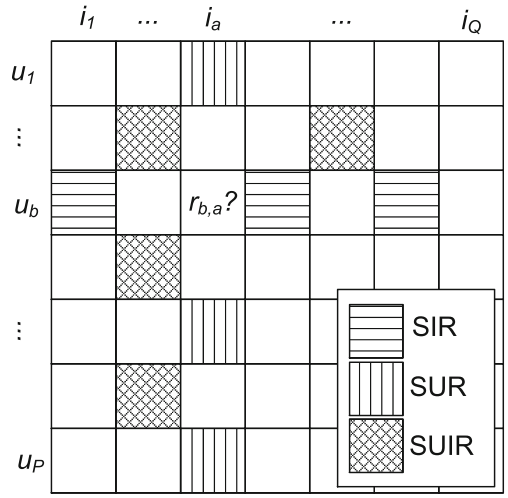

Fig. 1 Collaborative filtering approaches

- SIR, SUR and $SUIR$ denote predicting user interest over the entire item-user matrix from the ratings of the same user make on the similar items, the like-minded users make on the same item, and the like-minded users make on the similar items, i.e., SIR, SUR and $SUIR$ predict unrated items for active users based on SI, SU and $SUIR$, respectively.
- SR represent predicting user interest from all the ratings, i.e., SIR, SUR and $SUIR$,
- SIR', SUR', $SUIR'$ and SR' be the counterparts of SIR, SUR, $SUIR$ and SR, but they are calculated over the locally-reduced item-user matrix.

Then, the item vector of the matrix X is:

$$X_i = [i_1, i_2, \cdots, i_Q], i_q = [r_{1,q}, \cdots, r_{P,q}]^T,$$

where $q \in [1, Q]$. Each column vector i_m corresponds to the ratings of a particular item m by P users. Matrix X can also be represented by user vectors illustrated as:

$$X_u = [u_1, u_2, \cdots, u_P]^T, u_p = [r_{p,1}, \cdots, r_{p,Q}]^T,$$

where $p \in [1, P]$. Each row vector u_p^T indicates a user profile that represents a particular user's item ratings.

Item-based CF approaches, represented as SIR in Fig. 1, find similar items among item vectors and then use their ratingls made by the same user to predict his or her interest. Item-based approaches are intuitive and relatively simple to implement and thus became very popular in early recommender systems. Given an active item i_a and a user u_b, Eq. 1 denotes the mechanism of item-based CF approaches, where sim_{i_a,i_c} is the similarity of items i_a and i_c that is usually computed by Pearson Correlation Coefficient (PCC) or Vector Space Similarity (VSS). PCC-based CF approaches generally achieve higher performance than VSS-based approaches.

$$SIR : \widehat{r_{u_b,i_a}} \longleftarrow \frac{\sum_{i_c \in SI} sim_{i_a,i_c} \cdot r_{u_b,i_c}}{\sum_{i_c \in SI} sim_{i_a,i_c}} \tag{1}$$

An alternative to item-based CF is user-based CF, which takes advantage of the similar motivation to predict user interest. Specifically, the ratings of like-minded users made on the active item are used. Equation 2 shows the mechanism of user-based CF approaches, where sim_{u_b,u_c} is the similarity of users u_b and u_c.

$$SUR : \widehat{r_{u_b,i_a}} \longleftarrow \frac{\sum_{u_c \in SU} sim_{u_b,u_c} \cdot r_{u_c,i_a}}{\sum_{u_c \in SU} sim_{u_b,u_c}} \tag{2}$$

Both item-based and user-based approaches do not consider $SUIR$ that can improve the prediction accuracy. Let i be a similar item to i_a and u be a like-minded user to u_b, $SUIR$ is calculated as Eq. 3.

$$SUIR : \widehat{r_{u_b,i_a}} \longleftarrow \frac{\sum_{u,i \in SUI} sim_{(i,i_a),(u,u_b)} \cdot r_{u,i}}{\sum_{u,i \in SUI} sim_{(i,i_a),(u,u_b)}}, \tag{3}$$

where $sim_{(i,i_a),(u,u_b)}$ is the weight for the rating user u makes on item i, denoting how much the rating $r_{u,i}$ is considered in prediction. In CFSF, this similarity function is defined as Eq. 13. Note that most existing approaches only account for SIR and SUR, thus do not provide the measurement for $sim_{(i,i_a),(u,u_b)}$.

UI-based CF approaches have been proposed to incorporate the ratings that like-minded users made on similar items, which are ignored in item-based and user-based approaches [2, 7]. UI-based approaches are defined as Eq. 4 by combining SIR, SUR and $SUIR$,

$$SR : \widehat{r_{u_b,i_a}} \longleftarrow £\{SIR, SUR, SUIR\}, \tag{4}$$

where $£$ is a fusing function that fuses the ratings from SIR, SUR and $SUIR$, whose mechanisms are illustrated as Fig. 1. Due to the time-consuming search for active items and users over the entire item-user matrix, all memory-based CF approaches achieve limited scalability.

3 CFSF – large-scale E-commerce recommendation using smoothing and fusion for E-commerce providers

In order to solve data sparsity and limited scalability, we propose CFSF – Large-scale E-commerce Recommendation Using Smoothing and Fusion for e-commerce providers. In

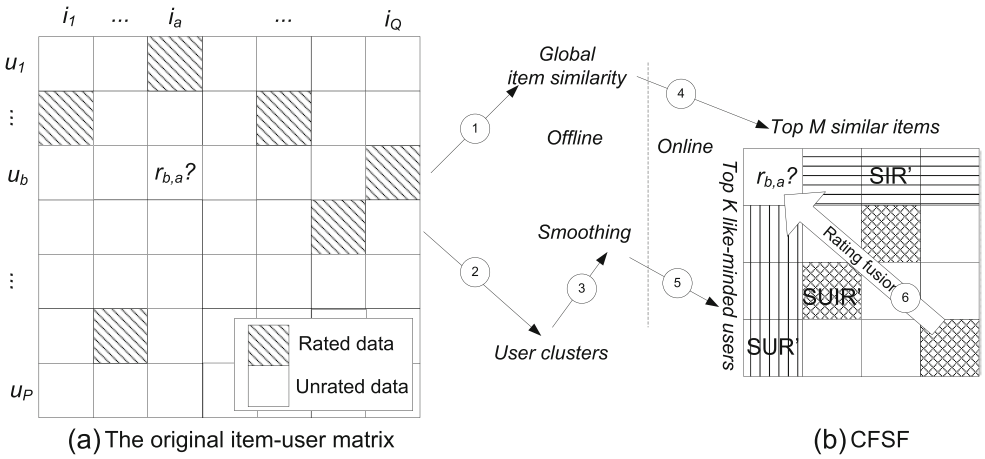

Fig. 2 Derive CFSF from a large-scale item-user matrix to a locally-reduced item-user matrix

this section, we briefly overview it and then introduce it in detail.

3.1 CFSF overview

CFSF aims to calculate the scores for unrated data over a locally-reduced item-user matrix. Given an active item and an active user, CFSF extracts ratings to construct a locally-reduced item-user matrix from most similar items and like-minded users, which significantly reduces the size of the item-user matrix and accelerates the prediction. In order to overcome the data sparsity, CFSF predicts unrated data by a fusing strategy that fuses ratings from the same user made on similar items, and similar users made on the same and similar items. CFSF significantly reduces the size of the item-user matrix used in prediction and thus accelerates prediction.

Figure 2 illustrates the process of deriving CFSF from a large-scale item-user matrix. This process involves six steps. First, CFSF creates a global item similarity matrix (GIS) as performed in the memory-based manner (i.e., search over the entire item-user matrix). Note that CFSF just store a few items that are most related to active users, e.g., may be less than one thousandth of all items in recommender systems. Then, CFSF classifies users into clusters, within each of which unrated ratings are smoothed. These two steps significantly reduce the influences of ratings diversity and accelerate the selection of like-minded users. Based on the request for an active user from the recommender system, CFSF dynamically constructs a locally-reduced item-user matrix to predict the score that the active user possibly makes on the active item. This step is achieved by picking up the top M similar items from GIS, the top K like-minded users from user clusters and extracts related ratings to create the locally-reduced item-user matrix. These similar items and like-minded users are already computed and can be fast extracted. In the end, it predicts user interest by fusing the ratings SIR', SUR' and $SUIR'$ over the locally-reduced item-user matrix. Figure 2 also shows that CFSF significantly reduces the number of users and items involved in prediction.

Given the increasing number of items and users in recommender systems, CFSF divides the CF process into offline and online phases. The offline phase involves the first three steps — creating the GIS, clustering users, and smoothing user ratings. This phase is a time-consuming process. The online phase includes the latter three steps — selecting items and users, constructing a locally-reduced item-user matrix and predicting scores for unrated items. For each user, CFSF requires him or her to rate a certain number of items and then inserts a record in the item-user matrix. Let M be the number of similar items and K be the number of like-minded users. Algorithm 1 illustrates the process of CFSF. Sections 3.2, 3.3 and 3.4 describe the offline phase and Section 3.5 describes the online phase.

Algorithm 1 CFSF algorithm

```
Procedure CFSF
    input: Item set: 𝓘, User set: 𝓤
    output: r_{u_b,i_a}: rating of item i_a by user u_b
    Offline
        Creating GIS ← I
        Clustering users C_i ← U
        Smoothing user ratings within each C_i
    End Offline
    // Constructing a locally-reduced M × K matrix
    Online
        //Neighbor selection
        Selecting top M items ← GIS
        Selecting top K users ← C_i
        //Fusing ratings
        Computing SIR', SUR' and SUIR'
        Calculating SR' ← £{SIR', SUR', SUIR'}
    End Online
End Procedure
```

3.2 Creating $GIS \leftarrow I$

In this step, CFSF creates and saves the item similarity matrix as GIS to eliminate the diversity in item ratings. Popular items tend to get higher ratings than unpopular items. Pearson Correlation Coefficient (PCC), rather than Pure Cosine Similarity (PCS), is selected as the item similarity function, because PCS does not consider the diversity in item ratings. Given items i_a, i_b and $U{=}U\{i_a\} \cap U\{i_b\}$, the similarity between i_a and i_b is defined as Eq. 5:

$$sim_{i_a,i_b} = \frac{\sum_{u \in \mathcal{U}} (r_{u,i_a} - \overline{r_{i_a}}) \cdot (r_{u,i_b} - \overline{r_{i_b}})}{\sqrt{\sum_{u \in \mathcal{U}} (r_{u,i_a} - \overline{r_{i_a}})^2} \cdot \sqrt{\sum_{u \in \mathcal{U}} (r_{u,i_b} - \overline{r_{i_b}})^2}} \tag{5}$$

Given the large number of items, we set thresholds for item similarity to filter the less important items. For instance, an item will not be considered if similarity between it and active item is less than a specific threshold. Consequently, the size of GIS will be greatly reduced.

3.3 Clustering users $C_i \leftarrow U$

In order to eliminate the diversity in user ratings, CFSF uses K-means to cluster users and then smoothes ratings within each user cluster. The K-means method trains the data iteratively and assigns every user to a cluster whose centroid is closest to him or her. The time complexity of each iteration is linear in the size of dataset. Compared with other clustering methods, K-means is simple, fast, and accurate. Its primary objective is minimizing $\sum_{i=1}^{k} \sum_{u_j \in C_i} sim|u_j - \overline{u}|$, where $\overline{u}$ is the centroid of all users that belong to C_i cluster. Similarity $sim|u_j - \overline{u}|$ is defined as Eq. 6 based on the PCC similarity function, where u_a and u_b are users, and $I{=}I(u_a) \cap I(u_b)$, denoting an item set that both users u_a and u_b have rated.

$$sim_{u_a,u_b} = \frac{\sum_{i \in \mathcal{I}} (r_{u_a,i} - \overline{r_{u_a}}) \cdot (r_{u_b,i} - \overline{r_{u_b}})}{\sqrt{\sum_{i \in \mathcal{I}} (r_{u_a,i} - \overline{r_{u_a}})^2} \cdot \sqrt{\sum_{i \in \mathcal{I}} (r_{u_b,i} - \overline{r_{u_b}})^2}} \tag{6}$$

Thus, user clusters are generated and used to smooth user ratings and select the top K like-minded users as was done in Sections 3.4 and 3.5.2.

3.4 Smoothing user ratings within C_i

Global effect negatively affects the recommendation accuracy, which refers to the rating diversity, i.e., users share similar tastes but have dissimilar rating styles so that some users tend to give higher ratings than other users and some items to receive higher ratings than others [6, 8]. On the other side, rating data sparsity should be considered in CF. Users prefer not to rate items and cannot rate all items due to the overwhelming number of items in recommender systems and thus the recommendation quality is low. In order to remove the influence of global effect and data sparsity, CFSF uses a smoothing strategy to smooth unrated data among user clusters. CFSF classifies users into clusters, in each of which users share similar tastes with different rating styles. CFSF smoothes unrated data by the average rating of this cluster. The smoothing function is defined as Eq. 7.

$$r_{u,i} = \begin{cases} r_{u,i}, & if\ u\ rates\ i \\ \overline{r_u} + \Delta r_{C_{u',i}} & otherwise \end{cases} \tag{7}$$

where $\Delta r_{C_{u',i}}$ is the deviation of average rating of item i in $C_{u',i}$ that is a set of users who rate the item i in user cluster $C_{u'}$. This deviation $\Delta r_{C_{u',i}}$ is given as Eq. 8:

$$\Delta r_{C_{u',i}} = \sum_{u \in C_{u',i}} (r_{u,i} - \overline{r_u})/|C_{u',i}|, \tag{8}$$

where $|C_{u',i}|$ is the size of $C_{u',i}$.

After smoothing, CFSF creates $iCluster$ for each user to store its similarity to each user cluster in a descending order. These $iCluster$s are used for selecting the top K like-minded users, which are computed in advance so that the process of selecting like-minded users can be finished quickly. The feature of a user cluster is denoted as a centroid that represents an average rating over all users in the cluster. Given an item set $\mathcal{I} = I\{u_a\} \cap I\{C_{u'}\}$, the similarity between user u_a and cluster $C_{u'}$ is defined as Eq. 9. Thus, we get the $iCluster$ for each user. For instance, the $iCluster$ for user u_a is $\{C_0, C_1, C_7, C_6, C_2, C_3, C_5, C_4\}$. Section 3.5.2 shows the selection process for this case.

$$sim_{u_a,C_{u'}} = \frac{\sum_{i \in \mathcal{I}} \Delta r_{C_{u',i}} \cdot (r_{u_a,i} - \overline{r_{u_a}})}{\sqrt{\sum_{i \in \mathcal{I}} \Delta (r_{C_{u',i}})^2} \cdot \sqrt{\sum_{i \in \mathcal{I}} (r_{u_a,i} - \overline{r_{u_a}})^2}} \tag{9}$$

So far, we have described all the steps in the offline phase that are often computation-intensive; and hence, performed in the backend. The online phase of CFSF focuses on responding to requests, including constructing a locally-reduced item-user matrix and fusing the ratings for prediction.

3.5 Constructing a locally-reduced $M \times K$ item-user matrix

In general, user interest is most likely derived from the most similar items and like-minded users. CFSF creates the locally-reduced item-user matrix containing the most related users and items; and thus, yields significant savings

in CPU, bandwidth, latency and other resources. When a request comes, CFSF will pick up the top M similar items from GIS, the top K like-minded users from user clusters C, and extract related ratings from the original item-user matrix.

3.5.1 Selecting top M similar items

Recall that CFSF computes and sorts the result of item similarity as GIS in descending order when it creates a global item similarity matrix. Consequently, CFSF can directly pick up the top M similar items from GIS.

3.5.2 Selecting top K like-minded users

User interest is often scattered into several user clusters. For instance, user u may like action, fantasy and crime types of movies. To cover user interest as much as possible, CFSF selects a user candidate set and then selects the top K like-minded users. To create a user candidate set, CFSF selects users from top several clusters in $iCluster$ one by one until it gets as many users as defined. CFSF extracts users from C_0 to C_4 (mentioned in Section 3.4) one by one until it gets the specified number of users, e.g., 200 users in experiments. Among these 200 users, CFSF selects the top K like-minded users by user similarity. Note that in this step CFSF involves two types of ratings, i.e., original and smoothed ratings. CFSF differentiates these two types of ratings with a parameter w when calculating the top K like-minded users. Specifically, the similarity between a user u and the active user u_a is defined as

$$sim_{u_a,u} = \frac{\sum_f w_{u,i} \cdot (r_{u,i} - \overline{r_u}) \cdot (r_{u_a,i} - \overline{r_{u_a}})}{\sqrt{\sum_f {w_{u,i}}^2 (r_{u,i} - \overline{r_u})^2} \cdot \sqrt{\sum_f (r_{u_a,i} - \overline{r_{u_a}})^2}}, \tag{10}$$

where f denotes $i \in I\{u_a\}$ and w is the coefficient, defined as Eq. 11. Depending on whether the rating is original or smooth, the weighting coefficient w varies.

$$w : w_{u,i} = \begin{cases} \varepsilon & if\ u\ rates\ i \\ 1-\varepsilon & otherwise \end{cases} \tag{11}$$

Compared with previous methods, CFSF reduces the computation overhead by selecting the like-minded users from iCluster rather than the entire item-user matrix. Moreover, CFSF is capable of setting thresholds for Eq. 10 to filter the less related users, which can further reduce the computation overhead.

3.5.3 Extracting ratings

After the top M similar items and the top K like-minded users are selected, CFSF will extract the related ratings from the original item-user matrix to fill in the locally-reduced item-user matrix.

Up to now, CFSF has constructed a locally-reduced item-user matrix illustrated in Fig. 2, which will be used to predict user interest in the next step.

3.6 Fusing SIR', SUR' and $SUIR'$

There are three ratings in the locally-reduced item-user matrix — ratings from the same user made on the similar items, like-minded users made on the same item, and like-minded users made on similar items. Predicting user interest from these ratings that are defined as SIR', SUR' and $SUIR'$ in CFSF, respectively. SIR', SUR' and $SUIR'$ shares the similar idea with SIR, SUR and $SUIR$, but they are much faster than the latter because they are computed over a locally-reduced $M \times K$ matrix whose size is much less than that of the original item-user $Q \times P$ matrix. For example, the size of the locally-reduced item-user matrix from the EachMovie dataset is 50×100, about 0.03 % of the size of the original EachMovie dataset ($1682 \times 10{,}000$).

Given active item i_a and user u_b, Eq. 12 illustrates these definitions.

$$\begin{aligned} SIR' &= \frac{\sum_{s=1}^{M} w \cdot sim_{i_s,i_a} \cdot r_{u_b,i_s}}{\sum_{s=1}^{M} w \cdot sim_{i_s,i_a}} \\ SUR' &= \frac{\sum_{t=1}^{K} w \cdot sim_{u_t,u_b} \cdot (r_{u_t,i_a} - \overline{r_{u_t}})}{\sum_{t=1}^{K} w \cdot sim_{u_t,u_b}} + \overline{r_{u_b}} \\ SUIR' &= \frac{\sum_{t=1}^{K}\sum_{s=1}^{M} w \cdot sim_{(i_s,i_a),(u_t,u_b)} \cdot r_{u,i}}{\sum_{t=1}^{K}\sum_{s=1}^{M} w \cdot sim_{(i_s,i_a),(u_t,u_b)}} \end{aligned}, \tag{12}$$

where w is defined as Eq. 11 and $sim_{(i,i_a),(u,u_b)}$ is defined by Euclidean distance as Eq. 13, denoting the weight for the rating of the similar item i by the like-minded user u.

$$sim_{(i_s,i_a),(u_t,u_b)} = \frac{sim_{i_s,i_a} \cdot sim_{u_t,u_b}}{\sqrt{sim_{i_s,i_a}^2 + sim_{u_t,u_b}^2}} \tag{13}$$

Figure 3 illustrates the similarity between similar items and users $sim_{(i_s,i_a),(u_t,u_b)}$. Compared with the similarities of similar items with the active item, and like-minded users with the active user, the similarity $sim_{(i_s,i_a),(u_t,u_b)}$ is less than them. Therefore, CFSF reflects such difference by employing Eq. 13 to limit its range between 0 and 0.7071.

CFSF selects SUR' as the major prediction tool and SIR' and $SUIR'$ as supplementary when it predicts user interest. Because SIR', SUR' and $SUIR'$ have different impact on recommendation accuracy, CFSF introduces two

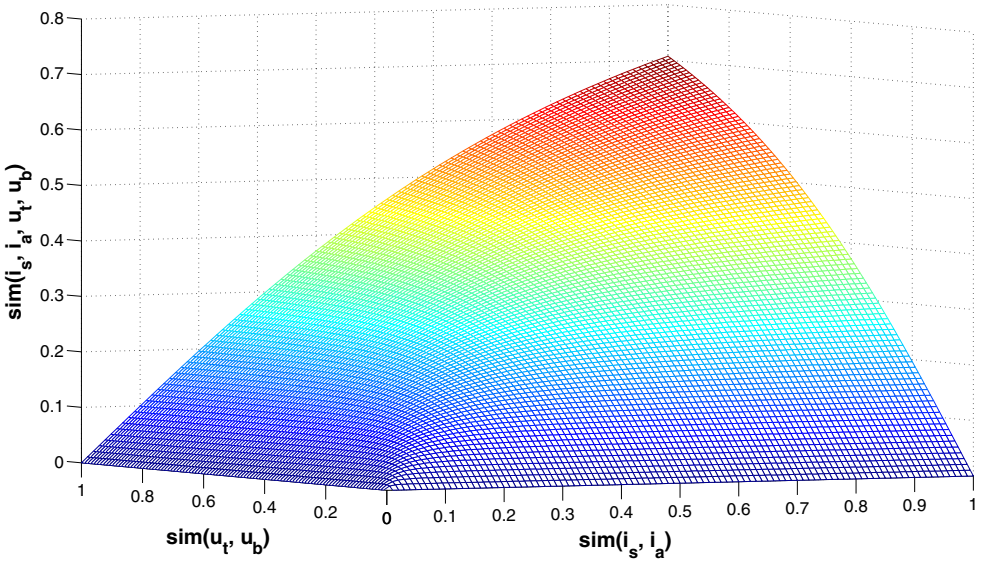

Fig. 3 The similarity between similar items and like-minded users $sim_{(i_s,i_a),(u_t,u_b)}$

parameters λ and δ to balance them. The fusing function of CFSF is defined as Eq. 14:

$$\begin{aligned} SR' : \widehat{r_{u_b,i_a}} &= \pounds\{SIR', SUR', SUIR'\} \\ &= (1-\delta)\cdot(1-\lambda)\cdot SIR' \\ &\quad +(1-\delta)\cdot\lambda\cdot SUR' \\ &\quad +\delta\cdot SUIR', \end{aligned} \tag{14}$$

where $\pounds$ is a function, and λ and δ are between 0 and 1. According to the Eq. 14, we obtain the Lemma 1.

Lemma 1 *Min(SIR′, SUR′, SUIR′) ≤ SR′ ≤ Max(SIR′, SUR′,SUIR′).*

Proof Let *min* and *max* represent $Min(SIR', SUR', SUIR')$ and $Max(SIR', SUR', SUIR')$. Let $\alpha' = (1-\delta)\cdot(1-\lambda)\cdot SIR'$, $\beta' = (1-\delta)\cdot\lambda\cdot SUR'$ and $\gamma' = \delta\cdot SUIR'$. Because parameters λ and δ are between 0 and 1, $(1-\delta)\cdot(1-\lambda)\cdot min \le \alpha' \le (1-\delta)\cdot(1-\lambda)\cdot max$, $(1-\delta)\cdot\lambda\cdot min \le \beta' \le (1-\delta)\cdot\lambda\cdot max$ and $\delta\cdot min \le \gamma' \le \delta\cdot max$. Thus, we get $min \le SR' \le max$. □

Theorem 1 *∃ λ, δ, P(CFSF) can achieve the best accuracy among SIR′, SUR′ and SUIR′ for the locally-reduced M × K item-user matrix.*

Proof According to the Lemma 1, CFSF can get the best ratings that active users made on active items by varying the values of parameters λ and δ. Thus, CFSF can achieve the best recommendation accuracy. □

Theorem 2 *CFSF eventually terminates the selection of a set of items and users from the item-user matrix.*

Proof According to the Algorithm 1, CFSF selects neighbors for the active item and user to construct a locally-reduced item-user matrix. CFSF first selects similar items and then like-minded users. Regardless of what manner (i.e., central or distributed) that CFSF selects items and users, CFSF eventually constructs a locally-reduced item-user matrix. This means that CFSF will terminate in finite time. □

Discussion The locally-reduced item-matrix is small-scale and thus remarkably accelerates the process of prediction. Note that there should never be more or less than M items and K users in the locally-reduced item-user matrix. CFSF constructs a locally-reduced item-user matrix by selecting top M similar items and top K like-minded users from the original item-user matrix. Only when the size of the original item-user matrix is less than $M \times K$, CFSF approach cannot select the M items or K users. This case cannot happen in the large-scale item-user matrix in recommender systems. Meanwhile, CFSF fuses three kinds of ratings across the locally-reduced item-user matrix, which significantly alleviates the influence of data sparsity.

Complexity CFSF contains two phases — offline and online. In the offline phase, its time complexity is very high, and determined by the creation of the global item similarity and K-means. To reduce the computation overhead, CFSF sets thresholds to filter less related items and users. In the online phase, its time complexity is $O(MK)$, where M and K are the number of similar items and like-minded users. Considering that M and K are much less than the original sizes of an item-user matrix, CFSF is scalable.

Table 1 Statistics of the big data sets

	MovieLens	EachMovie
Number of Users	500	10000
Number of Items	1000	1682
Avg. no. of rated Items/User	94.4	38
Density of data	9.44 %	2.37 %
No. of ratings	5	6

4 Evaluation

In order to evaluate CFSF, we carried out a series of experiments. In particular, we try to answer the following questions:

- What is the overall performance of CFSF? Does it work better than traditional item-based, user-based CF approaches, and the state-of-the-art CF approaches?
- How do the two fundamental problems of CF (sparsity and scalability) affect the performance of CFSF?
- How do the parameters influence the performance of CFSF? Several parameters are involved such as similarity fusion parameters λ and δ.

4.1 Dataset

The proposed approach is evaluated over two big data sets: MovieLens[4] and Eachmovie.[5] MovieLens from GroupLens Research at the University of Minnesota is one of the most popular big data for collaborative filtering. EachMovie is a public big data provided by HP/Compaq Research (formerly DEC Research), which contains more items and users than MovieLens. Most existing work is evaluated using one or two of them. We ran our program with Windows XP with 1 GB RAM and 2.4 GHz CPU.

We randomly extracted 500 users from MovieLens, where each user rated at least 40 movies. We changed the size of the training set by selecting the first 100, 200 and 300 users, denoted as ML_100, ML_200, and ML_300. The last 200 users were used as the test set. We varied the number of items rated by active users from 5, 10 to 20, denoted as Given5, Given10 and Given20. Similarly, we extracted 10,000 users from EachMovie with more than 38 ratings per user. We selected the first 500, 2000 and 6000 users for training, denoted as EM_500, EM_2000 and EM_6000. The last 4000 users were selected for testing. Table 1 summarizes the statistical features of the big data sets used in our experiments.

[4]G. Lab. MovieLens. http://www.grouplens.org/

[5]HP. EachMovie. http://www.research.digital.com/

4.2 Metrics

In order to be consistent with experiments reported in the literature [4, 6, 7], we choose the same MAE metric for evaluation, which is defined as:

$$MAE = \frac{\sum_{u \in T} |r_{u,i} - \widehat{r_{u,i}}|}{|T|}, \tag{15}$$

where $r_{u,i}$ denotes the rating that user u rates item i, $\widehat{r_{u,i}}$ denotes the predicted rating, T represents the test set and $|T|$ is the size of test set. The smaller the value of MAE, the better the performance.

4.3 Accuracy

4.3.1 Overall performance

We carried out experiments from two aspects to evaluate the performance of CFSF. One aspect is to compare CFSF with traditional memory-based CF approaches: an item-based approach using PCC (SIR) and a user-based approach using PCC (SUR). For MovieLens, the parameters of CFSF are set as follows: $C = 30, \lambda = 0.8, \delta = 0.1, K = 25, M = 95$ and $w = 0.35$. Table 2 illustrates the results, showing that CFSF

Table 2 MAE on MovieLens for the state-of-the-art CF approaches

Training set	Methods	Given5	Given10	Given20
ML_300	CFSF	**0.743**	**0.721**	**0.705**
	SUR	0.838	0.814	0.802
	SIR	0.870	0.838	0.813
	AM	0.820	0.822	0.796
	EMDP	0.788	0.754	0.746
	SCBPCC	0.822	0.810	0.778
	SF	0.804	0.761	0.769
	PD	0.827	0.815	0.789
ML_200	CFSF	**0.769**	**0.734**	**0.713**
	SUR	0.843	0.822	0.807
	SIR	0.855	0.834	0.812
	AM	0.849	0.837	0.815
	EMDP	0.793	0.760	0.751
	SCBPCC	0.831	0.813	0.784
	SF	0.827	0.773	0.783
	PD	0.836	0.815	0.792
ML_100	CFSF	**0.781**	**0.758**	**0.746**
	SUR	0.876	0.847	0.811
	SIR	0.890	0.801	0.824
	AM	0.963	0.922	0.887
	EMDP	0.807	0.769	0.765
	SCBPCC	0.848	0.819	0.789
	SF	0.847	0.774	0.792
	PD	0.849	0.817	0.808

Table 3 MAE on EachMovie for the state-of-the-art CF approaches

Training set	Methods	Given5	Given10	Given20
EM_6000	CFSF	**1.029**	**0.952**	**0.909**
	AM	1.117	1.069	1.046
	EMDP	1.032	0.975	0.931
	SCBPCC	1.073	1.001	0.956
	SF	1.066	1.004	0.953
	PD	1.101	1.063	1.051
EM_2000	CFSF	**1.062**	**0.979**	**0.932**
	AM	1.125	1.078	1.054
	EMDP	1.071	0.996	0.951
	SCBPCC	1.085	1.014	0.973
	SF	1.079	1.008	0.962
	PD	1.120	1.087	1.043
EM_500	CFSF	**1.084**	**0.973**	**0.948**
	AM	1.157	1.082	1.057
	EMDP	1.094	0.986	0.965
	SCBPCC	1.105	1.041	1.004
	SF	1.102	0.998	0.987
	PD	1.148	1.145	1.140

considerably outperforms the SUR and SIR with respect to recommendation accuracy.

The other aspect is to compare CFSF with the other state-of-the-art CF approaches, i.e., AM [3], EMDP [2], PD [3], SCBPCC [6] and SF [7]. We varied the item number that each user was required to rate on all the test sets for MovieLens and EachMovie big data sets. The results are shown as Tables 2 and 3. When the size of test set increases, the MAEs of all approaches show a downward trend. The same trend can be observed when the number of rated items for each user increases from 5 to 20. Among all approaches, CFSF achieves the best accuracy. This is because CFSF selects the most like-minded users by smoothing strategy and fuses ratings to achieve high levels of accuracy.

4.3.2 Accuracy with M similar items

The similar items M, like-minded users K and user clusters C conspicuously affect the accuracy of CFSF. In order to figure out their influence on CFSF, we conducted separate experiments on Given5, Given10 and Given20 over all the training sets for the MovieLens dataset.

Figure 4 shows the CFSF accuracy with M on ML_100, ML_200 and ML_300 training sets. CFSF achieves higher scalability as M increases. When M is less than 50, the similar items to active item are not many, leading to a high MAE. When M is greater than 60, CFSF collects enough ratings so that it achieves a low MAE. Figures 4a, b and c also show that CFSF improves the recommendation accuracy with the increase of the training set. For example, CFSF experiences heavy fluctuation in recommendation accuracy for the ML_100 training set, but light fluctuation for the ML_200 and ML_300 training sets.

4.3.3 Accuracy with K like-minded users

We did experiments over all the training sets with varying the value of K from 10 to 100 at Given5, Given10 and Given20 for all the big data.

Figure 5 shows the results of CFSF accuracy with K similar items over the ML_300 training set. When K is between 20 and 40, CFSF gets a low MAE. When K is larger than 40, the MAE value increases. This is because the ratings from less related users are overly considered for recommendation. As a result, rating smoothing strategy overly consider the ratings from the less related user clusters so that it negatively affects the recommendation accuracy

4.3.4 Accuracy with C user clusters

CFSF uses a smoothing strategy within user clusters to eliminate the global effect in rating data and the number of user clusters affects the performance of CFSF. We conducted

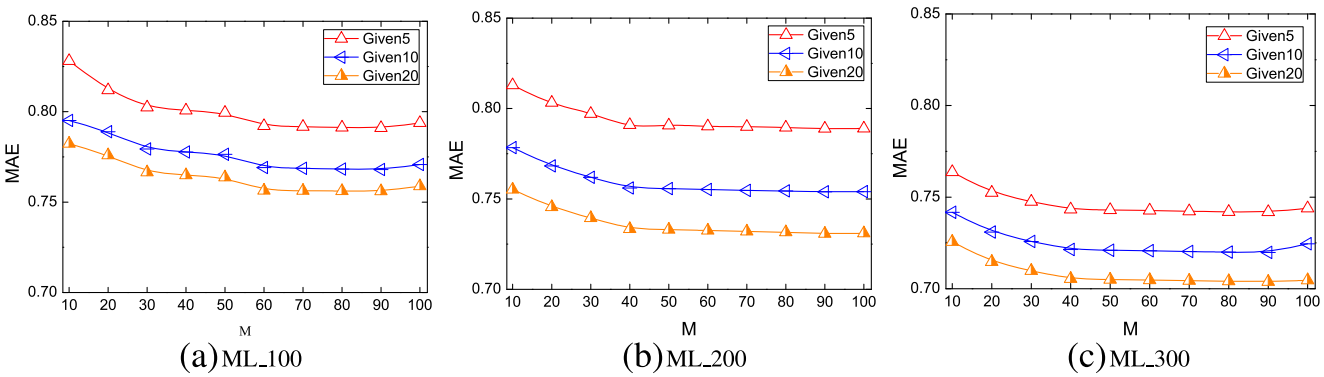

Fig. 4 Accuracy with similar items M over the ML_100, ML_200 and ML_300 (A smaller MAE means a better performance). When the value of parameter m is between 70 and 100, CFSF can achieve high levels of accuracy

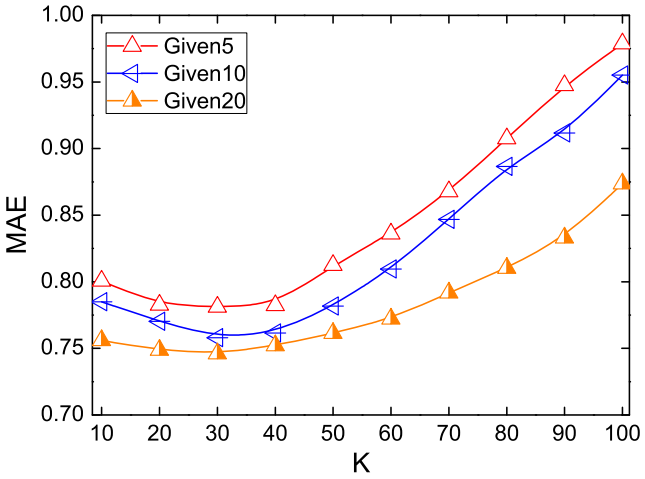

Fig. 5 Accuracy with K like-minded users over the ML_300 (A smaller MAE means a better performance). When the value of like-minded users K is between 30 and 60, CFSF can achieve high levels of accuracy

experiments for all the training sets by varying the values of C from 10 to 100.

Figure 6 illustrates accuracy with the user clusters C for ML_300. When C is less than 30, the user clusters used for selecting like-minded users do not cover most of user interest so that CFSF is incapable of getting a low MAE. When C is larger than 90, the user clusters used are too much and thus CFSF cannot properly eliminate the diversity. This indicates that the value of parameter C is between 30 and 90 is appropriate. In our experiment, we set its value as 30.

4.4 Scalability study

Scalability is extremely important for CF approaches, especially when they are used in larger-scale recommender systems. When the size of item-user matrix grows, CFSF needs to accommodate scale. We evaluate the scalability of the proposed approach by varying the training sets and test sets across MovieLens and EachMovie big data.

Note that CFSF consists of an offline phase and an online phase. The response time in the online phase is less than 2 seconds, because CFSF predicts unrated items over a locally-reduced item-user matrix, which does not take much time. Whereas in the offline phase, CFSF suffers from much computing overhead. Therefore, in the following parts of this section, we focus on evaluating the scalability for CFSF in the offline phase.

4.4.1 Response time

We evaluated the scalability of CFSF from two aspects. First, we checked its performance over the MovieLens dataset. We randomly selected 10 %, 20 % and up to 100 % of the last 200 users as test sets, and selected ML_100, ML_200 and ML_300 as training sets. The values of other parameters are set as the same value as those in Section 4.3.1.

Figure 7 shows the response time of CFSF for online prediction. As the test set grows, the response time increases in a linear fashion, indicating that CFSF is highly scalable. The maximum response time (i.e., the total time from training to testing) for ML_300 with 100 % percentage of the test set is 110 seconds, while SCBPCC spent around 260 seconds. CFSF achieves this by using the locally-reduced item-user matrix and caching intermediate results.

Second, we compared CFSF with other CF approaches. We report the results in EachMovie dataset, which contains more users and items than MovieLens (1,037,794 ratings). In our experiments, we evaluate CFSF with SCBPCC (much

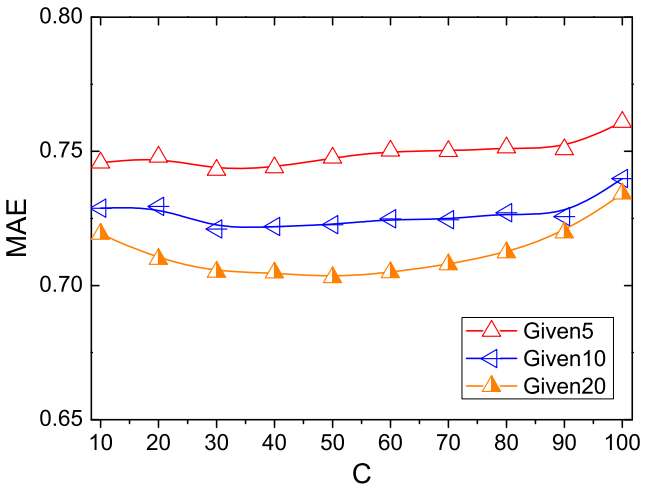

Fig. 6 Accuracy with C similar items over the ML_300 (A smaller MAE means a better performance). When the value of C is between 40 and 90, CFSF can achieve high levels of accuracy

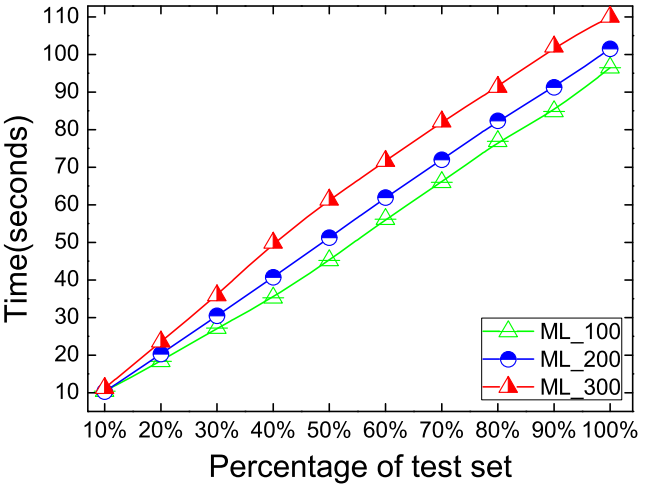

Fig. 7 Response time at Given20 on MovieLens in the offline phase. The response time of CFSF increases lowly, which indicates that CFSF is scalable for MovieLens dataset

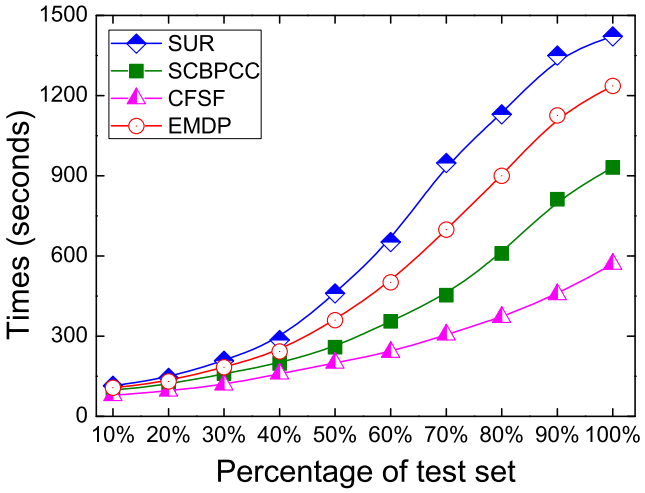

Fig. 8 Response time at Given20 for EM_6000 on EachMovie in the offline phase. The response time of CFSF increases linearly, which indicates that CFSF is scalable for EachMovie dataset that consists of more than 10,000,000 rating data

more scalable than AM, PD and SF [6, 7]), SUR (with similar scalability to SIR) and EMDP [2].

Figure 8 shows the response time at Given20 for EM_6000. The *X*-axis is the percentage of the test set that consists of 4,000 users. We selected 10 %, 20 % and up to 100 % of the test set in experiments to test the scalability of CFSF. With the growth of the test set, the execution times of all the approaches increase quickly. For example, EMDP spends much time in smoothing user ratings over the entire test set when the percentage of test set increases. The growth of the response time of CFSF is approaching linear. From 20 % to 100 % of the test set, CFSF requires less time than the other CF approaches, meaning that CFSF is more scalable than any other CF approaches. This is because the locally-reduced item-user matrix significantly reduces the sizes of similar items and like-minded users involved in prediction.

5 Conclusion

Big data offers e-commerce providers with many opportunities to make better QoS using service computing and cloud computing. One way to using big data is Collaborative Filtering (CF), which enables e-commerce systems to learn from user profiles, and promote purchase behavior by recommending products that might be favor of users. CF seriously suffers from two fundamental problems — data sparsity and limited scalability. To this end and to make better services as services provider, we have proposed CFSF – Large-scale E-commerce Recommendation Using Smoothing and Fusion for e-commerce providers. To summarize, the contributions of CFSF are two-fold. Firstly, it offers a mechanism to significantly reduce the scale of CF problem in recommender systems by mapping CF problem from the entire large-scale item-user matrix to a locally-reduced item-user matrix. This is achieved by the dual reduction in both items and users, which filters a great many of less related items and users. Secondly, CFSF presents smoothing and fusing strategies for the locally-reduced item-user matrix, which enable CFSF to achieve high levels of accuracy and scalability. Experimental results over the well-respected big data sets show that the proposed approach is more desirable than existing CF approaches for large-scale recommender systems.

Acknowledgments The authors would like to extend their sincere appreciation to the Deanship of Scientific Research at King Saud University for its funding of this research through the Research Group Project no RGP-VPP-258.

References

1. Lai C-F, Chang J-H, Hu C-C, Huang Y-M, Chao H-C (2011) Cprs: a cloud-based program recommendation system for digital TV platforms. Futur Gener Comput Syst 27(6):823–835
2. Xia W, He L, Gu J et al. (2009) Effective collaborative filtering approaches based on missing data imputation[C]//INC, IMS and IDC, 2009. NCM'09. Fifth Int Joint Conf IEEE:534–537
3. Zhang Y, Chen M, Mao S, Hu L, Leung V (2014) Cap: crowd activity prediction based on big data analysis. IEEE Netw 28(4):52–57
4. Khabbaz M, Lakshmanan LVS (2011) Toprecs: top-k algorithms for item-based collaborative filtering. In: EDBT
5. Menon AK, Chitrapura KP, Garg S, Agarwal D, Kota N (2011) Response prediction using collaborative filtering with hierarchies and side-information. In: KDD, ACM
6. Gong S (2010) A collaborative filtering recommendation algorithm based on user clustering and item clustering[J]. J Softw 5(7):745–752
7. Zheng VW, Cao B, Zheng Y et al. (2010) Collaborative filtering meets mobile recommendation: a user-centered approach[C]//AAAI 10:236–241
8. Lathia N, Hailes S, Capra L (2009) Temporal collaborative filtering with adaptive neighbourhoods[C]//Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval. ACM:796–797
9. Noel J, Sanner S, Tran K-N, Christen P, Xie L, Bonilla EV, Abbasnejad E, Della Penna N (2012) New objective functions for social collaborative filtering. In: WWW. ACM, New York, pp 859–868
10. Brochu E, Cora VM, De Freitas N (2010) A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning[J]. arXiv preprint arXiv:1012.2599
11. Zhang D, Cao J, Guo M, Zhou J, Raychoudhury V (2009) An efficient collaborative filtering approach using smoothing and fusing. In: Proceedings of the 38th International Conference on Parallel Processing, Vienna, Austria