IJICTR International Journal of Information & Communication Technology Research

Volume 6- Number 2- Spring 2014 (41-52)

A Cluster-Based Similarity Fusion Approach for Scaling-Up Collaborative Filtering Recommender System

Faezeh Sadat Gohari
Department of Industrial Engineering, IT Group
K.N. Toosi University of Technology
Tehran, Iran
fgohari@mail.kntu.ac.ir

Mohammad Jafar Tarokh
Department of Industrial Engineering, IT Group
K.N. Toosi University of Technology
Tehran, Iran
mjtarokh@kntu.ac.ir

Received: October 30, 2013-Accepted: Janurary 26, 2014

Abstract— **Collaborative Filtering (CF) recommenders work by collecting user ratings for items in a given domain and computing similarities between users or items to produce recommendations. The user-item rating database is extremely sparse. This means the number of ratings obtained is very small compared with the number of ratings that need to be predicted. CF suffers from the sparsity problem, resulting in poor quality recommendations and reduced coverage. Further, a CF algorithm needs calculations that are very expensive and grow non-linearly with the number of users and items in a database. Incited by these challenges, we present Cluster-Based Similarity Fusion (CBSF), a new hybrid collaborative filtering algorithm which can deal with the sparsity and scalability issues simultaneously. By the use of carefully selected clusters of users and items, CBSF reduces the computational cost of traditional CF, while retaining high accuracy. Experimental results demonstrate that apart from being scalable, CBSF leads to a better precision and coverage for the recommendation engine.**

Keywords-component; recommender systems; collaborative filtering; similarity fusion; clustering.

I. INTRODUCTION

Recommender systems aim at helping users search in overloaded information domains (like e-commerce [1, 2], e-learning [3, 4], DigitalTV [5, 6], etc). These systems guide the user in a personalized way to interesting items in a large space of possible options. Recommender systems are widely used in order to overcome the information overload problem in the World Wide Web environment [7, 8]. Such systems enhance e-commerce sales in three ways: converting browsers into buyers; improving cross-sell by suggesting additional products for customers; improving loyalty by creating a value-added relationship between the site and customers [9].

Two basic entities in a recommender system are: users and items. A user, who uses the recommender system is called *active user*. An active user provides his opinion about past items. The goal of the recommender system is to produce suggestions about new items for the active user. This process is based on the input provided, which is usually expressed in the form of ratings from active user, and the filtering algorithm, which is applied on that input [10]. Ratings in a recommendation system can be represented by a matrix which is called user-item rating matrix. In this matrix, rows represent users and columns represent items [11]. One of the most familiar and commonly used recommendation approaches is collaborative filtering (CF). CF uses the known preferences of a

group of users to make recommendations for other users. Two types of algorithms for CF have been studied: memory-based and model-based. Memory-based algorithms make predictions based on the entire collection of previously rated items. Memory-based algorithms are divided into user-based and item-based approaches. Model-based algorithms use the collection of ratings to learn a model, which is then used to make predictions [8]. CF systems have been used fairly successfully in various domains. However, the data sparsity problem is one of the main limitations of CF [12]. Moreover, memory-based methods are not scalable to very large data sets, while model-based methods improve system scalability at the expense of accuracy [13].

In this paper, we present a new Cluster-Based Similarity Fusion (CBSF). CBSF is a hybrid between memory-based and model-based approaches. CBSF approach employs a novel memory-based method, called *Similarity Fusion* (SF) [12]. Similarity fusion method unifies user-based and item-based collaborative filtering approaches in a single framework. In this method, the final rating is estimated by fusing predictions from different sources. The SF model exploits more of the data available in the user-item matrix. Therefore, it is more robust to data sparsity, but it is not scalable for large datasets. In order to reduce the scalability problem of SF, we integrate it with clustering results. Clustering is probably the most popular model-based method which can improve the scalability problem of memory-based methods.

The contributions of this paper are summarized as follows:

- Presentation of a hybrid recommendation approach which combines item-based, user-based and model-based collaborative filtering to reduce sparsity and scalability problems simultaneously.
- Presentation of a cluster-based similarity fusion matrix which combines the SF method with user clustering and item clustering.
- An experimental comparison between the quality of proposed approach with the original similarity fusion and other collaborative filtering algorithms (item-based, user-based and model-based).

Integrating SF model and clustering allows us to take the advantage of accuracy in the memory-based method as well as the scalability of the model-based method.

The rest of this paper is organized as follows. The following section provides a brief description of several major approaches for collaborative filtering. Section 3 describes our proposed CBSF approach and Section 4 demonstrates the experimental evaluation and results. Finally, we present our conclusions and outline future lines of research in Section 5.

II. Related Works

Collaborative Filtering works by collecting user ratings for items in a given domain and calculating similarities between users or items to produce recommendations [8]. Two families of CF algorithms have been studied extensively in the literature: (1) memory-based algorithms, which perform the computation on the entire database to identify the top k most similar users to the active user. (2) Model-based algorithms, which compute a model of predefined classes of users based on their rating patterns [14].

A. Memory-based CF

Memory-based methods are simpler and work reasonably well in practice. Also, in these methods, new data can be added easily and incrementally [15]. Memory-based algorithms are divided into User-Based (UB) and Item-Based (IB) approaches [16].

UB methods [17, 18] first look for some similar users to the active user which is called the neighbors. Then, they employ the ratings from those neighbors to predict the ratings for the active user. IB methods [19], [20] share the same idea with UB methods, but item-based approaches try to find the similar items for each item [16]. The most extensively used similarity measures in UB or IB approaches are Pearson Correlation Coefficient (PCC) [17] and Vector Space Similarity [18]. The computational complexity of UB methods grows linearly with the number of customers, which in typical commercial applications can be several millions—scalability problem [20]. In contrast, IB can quickly recommend a set of items because item-neighborhood matrix is generated off-line. However, there are experiments showing that UB provides more accurate recommendations than IB [21].

One of the main limitations of memory-based algorithms is sparsity of rating matrix. Sparsity reduces accuracy of predictions. With a sparse ratings matrix, a recommender system becomes unable to find similar neighbors and fails to produce proper recommendations [22]. In both cases of UB and IB, only partial information from the data in the user-item matrix is employed to predict unknown ratings. UB uses “similar user ratings” (SUR data) and IB uses “similar item ratings” (SIR data). Because of the sparsity of user profile data, many related ratings will not be available for the prediction [12]. Wang et al. [12] proposed the Similarity Fusion (SF) between the UB and IB methods, using also data from a third source—ratings of similar users on similar items (SUIR data). This model is more robust to data sparsity, because it exploits more of the data available in the user-item matrix.

B. Model-based CF

In this approach, an underlying model of user preferences is constructed at first, then predictions are inferred from this model [15]. In model-based algorithms, predictions can be calculated quickly once the model is generated. However, they have the overhead to build and update the model. Model-based methods improve system scalability at the expense of accuracy [14]. Algorithms within this category include Bayesian network models [23], latent class models [24], regression models [25], clustering models [26], etc.

Hofmann and Puzicha [24] presented the Aspect Model (AM), which models individual preferences as a convex combination of preference factors. The latent

class variable is associated with each observation pair of a user and an item. The aspect model assumes that users and items are independent from each other given the latent class variable. Vucetic and Obradovic [25] proposed a Regression-Based (RB) approach that searches for similarities between items, builds a collection of experts in the form of simple linear models, and combines them to provide predictions. Jin et al. [27] proposed the Decoupled Model (DM), which satisfies all the desired properties that a graphical model is expected to satisfy. The authors showed that DM outperforms the other mixture models for CF.

Clustering is probably the most widely method used in model-based approaches. It finds groups of users or items that have similar preferences [28]. Clustering models have better scalability than typical CF methods, but their recommendation quality is generally low [13]. In the cluster-based CF area, Sarwar et al. [26] scaled up the neighborhood formation of CF by using clustering techniques. Bridge and Kelleher [29] reduced cardinality and sparsity of a collaborative recommender's data set by applying a *K*-means-like clustering technique. In another work, Kelleher and Bridge [30] presented a user-based model for CF which comprises summary information derived from a hierarchical clustering of the users. Gong [31] joined the user clustering and item clustering for CF. This approach smoothes the predictions based on the nearest clusters to the active user. Then, it uses the item clustering to produce the recommendations. Birtolo and Ronca [32] proposed two clustering-based collaborative filtering algorithms which can increase the coverage and provide high-level recommendation for different users. Wang [33] proposed an e-commerce collaborative algorithm based on product clustering. In this approach, the clustering of product is used to search the neighbors in the clustering centers. To overcome the uncertainty of the users neighborhoods, an improved collaborative algorithm based on user clustering is proposed by Deng and Wang [34]. This algorithm filters the users by their features and uses the improved cosine similarity algorithm for computing similarities between items. Bilge and Polat [35] proposed a novel collaborative filtering scheme based on bisecting *K*-means clustering. Based on the results, this scheme relieves scalability and significantly augments accuracy. Tsai and Hung [36] assessed the applicability of cluster ensembles to collaborative recommendations. The authors showed that cluster ensembles can provide better performance than single clustering techniques in terms of recommendation accuracy and precision.

C. Hybrid of memory-based and model-based CF

Memory-based and model-based CF approaches, can be combined to form hybrid CF approaches. The recommendation performances of these algorithms are generally better than pure memory-based and model-based algorithms [13].

Yu et al. [37] presented a probabilistic model for user preferences. The authors used a mixture model built based on a set of stored user profiles; thus the model is clearly linked with memory-based methods. Suryavanshi et al. [38] proposed a new fuzzy hybrid CF approach. In this approach, the fuzzy nearest prototype of the active user is used to find a group of like-minded users. Then, a memory-based search is carried out within this group. Chuan et al. [39] proposed a recommendation algorithm combining the user-based classified regression model and the item-based CF. This approach helps to avoid the following problems of CF. The first problem is that the value of similarity between users who give very different ratings might be high. Another is that the classification information about resources is not used in CF. ClustKnn [40] is a hybrid algorithm which first compresses data tremendously by building a clustering model. Recommendations are then generated quickly using a simple nearest neighbor-based approach. Wang et al. [41] proposed a combination filtering method which firstly constructs a user model off-line. Then, it forms the neighbor set based on the model and makes on-line recommendation using memory-based CF. Chen et al. [42] presented RegionKNN, a novel hybrid collaborative algorithm that is designed for large-scale web service recommendation. RegionKNN is highly scalable and provides considerable improvement on the recommendation accuracy. Zhang et al. [43] proposed an efficient CF approach using smoothing and fusing strategies. Their approach clusters users with a smoothing strategy to eliminate the diversity in user ratings styles. Then, it fuses different rating sources for producing on-line recommendations. Moghaddam and Selamat [44] clustered users based on their demographic information and then partitioned user rates based on the clusters. Finally, they applied user-based CF on each partition separately. Pennock et al. [15] proposed a new hybrid CF method called Personality Diagnosis (PD). Given a user's preferences for some items, PD estimates the probability that a user belongs to the same "personality diagnosis". PD assigns the missing rating as a uniform distribution over all possible ratings. The authors showed that this method can outperform several other approaches for CF. Xue et al. [14] proposed an accurate and scalable CF using Cluster-Based Smoothing (CBS). CBS approach clusters the user data and applies intra-cluster smoothing to reduce sparsity. Experimental results show that CBS outperforms several other approaches for CF, including the PCC method, the personality diagnosis method and the Bayesian network approach.

The aim of this work is to present a new hybrid approach which alleviates sparsity and scalability problems of traditional CF approaches. Our proposed CBSF approach uses the Similarity Fusion method as a memory-based algorithm. The main motivation for applying SF model is that it can cope with the data sparsity problem, and it is more accurate than the other state-of-the-art algorithms such as PCC, AM, PD and CBS [12]. In order to cope with the scalability problem of SF, we integrate it with clustering models. So, CBSF connects memory-based and model-based CF in a single framework. Next Section describes the CBSF approach in detail.

III. THE PROPOSED CBSF APPROACH

To cope with the data sparsity and scalability problems of CF, we propose a Cluster-Based Similarity Fusion approach (CBSF). CBSF combines the Similarity Fusion approach with clustering methods. In this approach, we cluster users and items based on their rating similarities. Then, we select the most similar clusters to the active user and also the most similar ones to the target item. Based on neighboring clusters, we create a cluster-based similarity fusion matrix. Based on this matrix, we predict the unknown rating of active user for the target item. Our proposed approach is shown in Fig. 1.

In the following, we present our proposed approach in detail. Table 1 summarizes the symbols that are used in the sequel.

A. Clustering

In our approach, we cluster the users and items off-line to reduce the on-line computational overload and overcome the scalability problem. There are many algorithms that can be used to create clusters. We choose the *K*-medoids algorithm [45], mostly for its high accuracy. Similar to *K*-means algorithm, *K*-medoids method is a partition based clustering. In contrast to the *K*-means, *K*-medoids algorithm chooses objects as centers (medoids) instead of taking the mean value of the objects. *K*-medoids can work with an arbitrary matrix of distances (or similarity) between objects. It is more robust than *K*-means to the noise and it is invariant to orthogonal transformations of objects. But, the computational complexity of the *K*-medoids is more than *K*-means.

One of the best-known *K*-medoids based algorithms is PAM (Partitioning Around Medoids) [45]. PAM is based on an iterative process of optimization. In each iteration medoid object *i* and non-medoid object *j* are selected that produce the best clustering when their roles are switched. The objective function used is the sum of the distances from each object to the closest medoid [46]. The PAM algorithm results in high quality clusters, as it tries every possible combination [47]. The time complexity of PAM can be reduced by pre-computed similarity matrix.

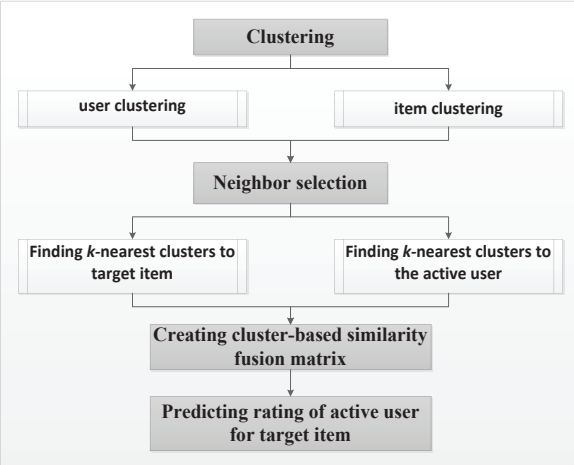

Figure 1. The proposed cluster-based similarity fusion approach

TABLE 1. SYMBOLS AND DEFINITIONS

Symbol	Definition
$U = \{u_1, u_2, ..., u_n\}$	Set of users in the database
$I = \{i_1, i_2, ..., i_m\}$	Set of items in the database
u_a	Active user
i_t	Target item
$r_{u,i}$	Rating of user *u* for item *i*
I_u	Set of items rated by user *u*
U_i	Set of users who rated item *i*
$\bar{r}_u$	Mean rating value for user *u*
$\bar{r}_i$	Mean rating value for item *i*
K	Number of clusters
k	Number of neighbors
$CU = \{cu_1, cu_2, ..., cu_K\}$	Clusters of users
$CI = \{ci_1, ci_2, ..., ci_K\}$	Clusters of items
I_{cu}	Set of items rated by at least one user in cluster *cu*
U_{ci}	Set of users who rated at least one item of cluster *ci*
I_u^{ci}	Set of items in cluster *ci* rated by user *u*
U_i^{cu}	Set of users in cluster *cu* who rated item *i*
$U_{cu,ci}$	Set of users in cluster *cu* who rated at least one item of cluster *ci*
$\Delta r_{cu,i}$	Average deviation in ratings of all users in cluster *cu* for the item *i*
$\Delta r_{u,ci}$	Average deviation in ratings of user *u* for all items belong to cluster *ci*
$\Delta r_{cu,ci}$	Average deviation in ratings of all users in cluster *cu* for all items belong to cluster *ci*

In this work, we employ PAM algorithm with pre-computed similarity matrix. Our clustering process is based on the user-item rating matrix. In order to calculate similarity between two users (or items), we use the Pearson Correlation Coefficient between [48] their vectors of ratings.

1) User clustering

User clustering identifies groups of users who appear to have similar rating vectors. First, the PAM algorithm randomly selects *K* users as the initial medoids. This algorithm proceeds by alternating between two steps. In the first step, each user is assigned to the cluster associated with the most similar medoid. In the next step, each medoid is swapped with each of the non-medoid users and the total cost of the configuration is computed. Then, the configuration with the lowest cost is selected. The algorithm repeats these steps until the medoids become fixed.

In order to compute similarity between two users, we use the Pearson Correlation Coefficient [48] between their vectors of ratings. So, the similarity measure function for *K*-medoids clustering of users is defined as:

$$Sim_{u_1,u_2} = \frac{\sum_{\forall i \in I_{u_1} \cap I_{u_2}} (r_{u_1,i} - \bar{r}_{u_1}) \cdot (r_{u_2,i} - \bar{r}_{u_2})}{\sqrt{\sum_{\forall i \in I_{u_1} \cap I_{u_2}} (r_{u_1,i} - \bar{r}_{u_1})^2} \cdot \sqrt{\sum_{\forall i \in I_{u_1} \cap I_{u_2}} (r_{u_2,i} - \bar{r}_{u_2})^2}} \quad (1)$$

2) Item Clustering

Item clustering is similar to user clustering. First, *K* items are selected as the initial medoids. Each item is assigned to the cluster associated with the most similar medoid. Then, the positions of the medoids are re-calculated and the configuration with the lowest cost is selected. The algorithm repeats these steps until the assignments no longer change.

In order to compute similarity between two items, we use the Pearson Correlation Coefficient [48] between their vectors of ratings. So, the similarity measure function for *K*-medoids clustering of items is defined as:

$$Sim_{i_1,i_2} = \frac{\sum_{\forall u \in U_{i_1} \cap U_{i_2}} (r_{u,i_1} - \bar{r}_{i_1}) \cdot (r_{u,i_2} - \bar{r}_{i_2})}{\sqrt{\sum_{\forall u \in U_{i_1} \cap U_{i_2}} (r_{u,i_1} - \bar{r}_{i_1})^2} \cdot \sqrt{\sum_{\forall u \in U_{i_1} \cap U_{i_2}} (r_{u,i_2} - \bar{r}_{i_2})^2}} \quad (2)$$

B. Neighbor selection

An important step of CF algorithm is the neighborhood formation. In this step, top-k most similar clusters are selected as the nearest neighbors.

Traditional memory-based CF searches the whole database, so it has the scalability problem in large databases. By using the concept of a cluster, we can do this step more efficiently. Instead of each individual object, we calculate the similarity between a target object and each cluster. Therefore, clustering can help speed up the similarity calculation [14].

1) Finding k-nearest clusters to the active user

Xue et al. [14] proposed the following function for calculating the similarity between a user cluster *cu* and an active user u_a:

$$Sim_{u_a,cu} = \frac{\sum_{\forall i \in I_{u_a} \cap I_{cu}} \Delta r_{cu,i} \cdot (r_{u_a,i} - \bar{r}_{u_a})}{\sqrt{\sum_{\forall i \in I_{u_a} \cap I_{cu}} (\Delta r_{cu,i})^2} \sqrt{\sum_{\forall i \in I_{u_a} \cap I_{cu}} (r_{u_a,i} - \bar{r}_{u_a})^2}} \quad (3)$$

where $\Delta r_{cu,i}$ is defined as:

$$\Delta r_{cu,i} = \sum_{\forall u \in U_i^{cu}} \frac{(r_{u,i} - \bar{r}_u)}{\left| U_i^{cu} \right|} \quad (4)$$

After calculating the similarity between each user cluster and the active user, we select the top-*k* most similar clusters as the neighbors.

2) Finding k-nearest clusters to target item

For calculating the similarity between an item cluster *ci* and a target item i_t, we modify equations (3) and (4) and present the following functions:

$$Sim_{i_t,ci} = \frac{\sum_{\forall u \in U_{i_t} \cap U_{ci}} \Delta r_{u,ci} \cdot (r_{u,i_t} - \bar{r}_{i_t})}{\sqrt{\sum_{\forall u \in U_{i_t} \cap U_{ci}} (\Delta r_{u,ci})^2} \sqrt{\sum_{\forall u \in U_{i_t} \cap U_{ci}} (r_{u,i_t} - \bar{r}_{i_t})^2}} \quad (5)$$

where $\Delta r_{u,ci}$ is defined as:

$$\Delta r_{u,ci} = \sum_{\forall i \in I_u^{ci}} \frac{(r_{u,i} - \bar{r}_i)}{\left| I_u^{ci} \right|} \quad (6)$$

After calculating the similarity between each item cluster and the target item, we select the top-*k* most similar clusters as the neighbors.

C. Creating a cluster-based similarity fusion matrix

For predicting r_{u_a,i_t}, we use similarity fusion model introduced by Wang et al. [12]. In this model, *n* users with the highest similarity to the active user are selected as his neighbors. Also, *n* most similar items to the target item are selected as its neighbors. The unknown rating of active user u_a for target item i_t is predicted by fusing three sources: (1) ratings of the item i_t by the neighbors of the user u_a, (2) ratings of the user u_a for neighbors of the item i_t, and, (3) ratings from neighbors of the user u_a towards neighbors of the item i_t.

This model works on a matrix, called "similarity fusion" matrix, which its first row and first column represents the active user and target item, respectively. The remaining rows and columns represent the neighbors of the active user and target item, respectively. The first cell of this matrix is related to the rating of active user for the target item. This model has to predict the value of this cell. The other cells contain ratings of individual users for individual items. Furthermore, each row vector of this matrix is weighted according to the similarity between the active user and his neighbors. Also, each column vector is weighted according to the similarity between target item and its neighbors. These weights are used for computing r_{u_a,i_t} based on the weighted combination of ratings in the matrix. An example of such fusion matrix can be found in [49].

In this article, we modify similarity fusion matrix based on clustering results and present a new model, called "cluster-based similarity fusion (CBSF)" matrix. For creating such a matrix, we select the most similar clusters to the active user and also the most similar ones to the target item. Because of considering user clusters and item clusters, CBSF uses the average deviation in ratings of the cluster objects instead of individual ratings. Fig. 2 shows an example of such a matrix. The main differences between this new matrix and old similarity fusion matrix are as follows:

- In the CBSF matrix, neighbors of the active user (or target item) are clusters of users (or items), while in the SF matrix, neighbors are individual users or items.

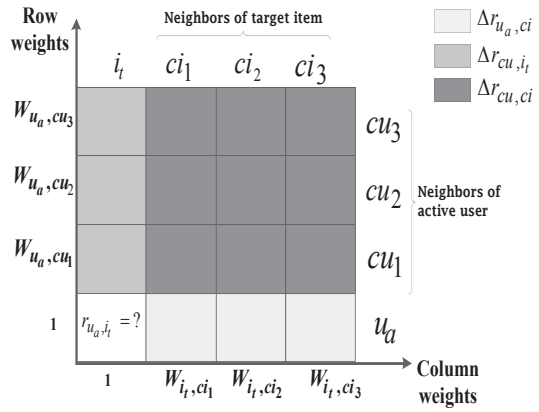

Figure 2. An example of a cluster-based similarity fusion matrix

- In the CBSF matrix, similarity between active user (or target item) and each neighboring cluster is considered as the weight. But, in the SF matrix, similarity between active user (or target item) and each individual neighbor is considered as the weight.

- In the CBSF matrix, each cell contains the average deviation in ratings of cluster objects, while in the SF matrix, individual ratings are used.

As mentioned above, CBSF matrix uses the average deviation in ratings of cluster objects. Accordingly, the available rating sources in CBSF matrix include:

1) $\Delta r_{cu,i_t}$, is the average deviation in the ratings of all users in the neighboring cluster *cu* for the target item i_t.

2) $\Delta r_{u_a,ci}$, is the average deviation in ratings of the active user for all items belong to cluster *ci*.

3) $\Delta r_{cu,ci}$, is the average deviation in the ratings of all users in cluster *cu* for all items in the cluster *ci*.

We defined $\Delta r_{cu,i}$ and $\Delta r_{u,ci}$ in the previous section. Similarly, we define $\Delta r_{cu,ci}$ as follows:

$$\Delta r_{cu,ci} = \sum_{\forall u \in U_{ci}^{cu}} \frac{(\Delta r_{u,ci} - \bar{r}_u)}{\left|U_{ci}^{cu}\right|} \tag{7}$$

Actually, the equation (7) is similar to equation (4), but it is based on a cluster of items instead of an individual item. Therefore, here we consider a set of users in cluster *cu* who rated at least one item in cluster *ci*; and we replace $r_{u,i}$ with $\Delta r_{u,ci}$.

As shown in Fig. 2, a weight is assigned to each neighboring cluster. The weight $W_{u_a,cu}$ is a measure of similarity between the user cluster *cu* and the active user (equation 3). The weight $W_{i_t,ci}$ is a measure of similarity between the item cluster *ci* and the target item (equation 5). We compute the final prediction based on the weighted combination of the ratings in the matrix.

D. Predicting rating of active user for target item

Now we compute r_{u_a,i_t} based on the cluster-based similarity fusion matrix. This rating is estimated by fusing three available data sources in CBSF matrix. The weighted combination of the ratings in this matrix is defined as:

$$r_{u_a,i_t} = \bar{r}_{u_a} + \frac{WR_1 + WR_2 + WR_3}{S_1 + S_2 + S_3}$$

$$WR_1 = \sum_{\forall cu \in CU_N} (\Delta r_{cu,i_t} \times sim_{u_a,cu}) \quad ,$$

$$WR_2 = \sum_{\forall ci \in CI_N} (\Delta r_{u_a,ci} \times sim_{i_t,ci}) \quad ,$$

$$WR_3 = \sum_{\forall cu \in CU_N} \sum_{\forall ci \in CI_N} (\Delta r_{cu,ci} \times sim_{u_a,cu} \times sim_{i_t,ci}) \quad ,$$

$$S_1 = \sum_{\forall cu \in CU_N} sim_{u_a,cu} \quad ,$$

$$S_2 = \sum_{\forall ci \in CI_N} sim_{i_t,ci} \quad ,$$

$$S_3 = \sum_{\forall cu \in CU_N} \sum_{\forall ci \in CI_N} sim_{u_a,cu} \times sim_{i_t,ci} \tag{8}$$

where CU_N is set of clusters which belong to the active user's neighborhood and CI_N is set of clusters which belong to the target item's neighborhood. This model exploits more data and therefore it is more robust to data sparsity. Therefore, our approach takes the advantage of accuracy in the similarity fusion model as well as the scalability of the clustering method.

IV. Experimental Evaluations

In this section, we examine the performance of the proposed approach. We use MovieLens 100k dataset which consists of 100,000 ratings, with the scale of one to five, from 943 users on 1,682 movies. This dataset has some inconsistencies (for example duplicate or unknown movies and duplicate ratings). We correct these inconsistencies and then remove users having less than 20 ratings. We partition the users into test users and train users using 10-fold cross-validation. The ratings withheld in the test set are randomly chosen based on *Given* 5, *Given* 10 and *Given* 20 experimental protocols [18]. Table 2 represents the parameters and their default values in our experiments.

A. Evaluation metrics

In order to evaluate the accuracy of predicted ratings, we use Mean Absolute Error (MAE) metric [50]. MAE measures the average absolute deviation between a predicted rating and the user's true rating. The Mean Absolute Error for each test user *u* is defined as:

$$MAE = \frac{\sum_{\forall i \in I_u} \left|p_{u,i} - r_{u,i}\right|}{\left|I_u\right|} \tag{9}$$

where $p_{u,i}$ is the predicted rating for user *u* on item *i*. In our experiments, we compute the MAE on the

TABLE 2. PARAMETERS AND THEIR DEFAULT VALUES

Parameter	Symbol	Default value
Number of user clusters	K_u	30
Number of item clusters	K_i	30
Active user's neighborhood size	k_{u_a}	4
Target item's neighborhood size	k_{i_t}	4

test set for each user, and then average over the set of test users. The lower the MAE is, the more accurately the recommender system predicts user ratings.

In addition to evaluating the accuracy of the proposed approach, we also examine its suitability. For this purpose, we use coverage metric. Coverage refers to the proportion of items that the recommender system can recommend. A higher coverage means that the system is capable to support decision making in more situations [50].

B. Parameters tuning

In this section, we examine the impact of parameters K_u , K_i , k_{u_a} and k_{i_t} , on the performance of CBSF approach. For this purpose, we conduct experiments over Given5, Given10 and Given20 experimental protocols.

• *Number of user clusters (K_u):* The number of user clusters makes great impact on accuracy of recommendation. In order to examine how much the K_u impact on the results of CBSF, we conduct an experimental analysis on different number of clusters. In this experiment, the accuracy of predictions is examined against the different number of user clusters. We vary the number of user clusters from 10 to 100. Fig. 3 shows the performance of our recommender over 10 folds for different K_u .

With the increment of K_u , MAE decreases first, but performs an increment trend when K_u goes beyond a certain threshold value. The small number of clusters causes the cluster information too general to represent differences among dissimilar users, while larger ones make the cluster information more specific. As shown in Fig. 3, when K_u is between 20 and 30, CBSF gets its lowest MAE. This observation demonstrates the importance of appropriate user clustering number.

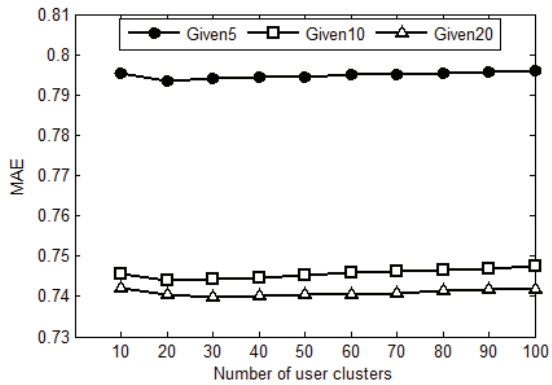

Figure 3. MAE against different number of user clusters

• *Number of item clusters (K_i):* The number of item clusters also impacts the accuracy of the recommendation process. Similar to the previous experiment, we examine the performance of MAE metric versus different values for K_i . In this experiment, we vary the number of item clusters from 10 to 150. Fig. 4 shows the performance of our recommender for different K_i .

As shown, the MAE decreases as we increase K_i from 10 to 40. After that the MAE increases due to the more specific cluster information. This observation confirms that the appropriate number of item clusters can result in the better performance.

• *Active user's neighborhood size (k_{u_a}):* The number of nearest neighbors used for the neighborhood formation is important, because it can affect the system's accuracy. Fig. 5 shows the accuracy of CBSF for various k_{u_a} .

When k_{u_a} is less than 8, the similar user clusters to the active item are not adequate, leading to a high MAE. With the increment of k_{u_a} , MAE decreases,

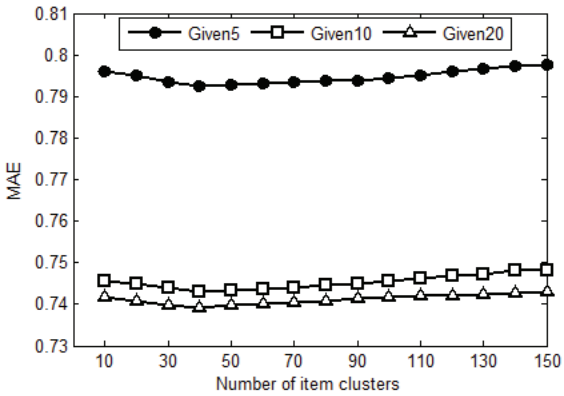

Figure 4. MAE against different number of item clusters

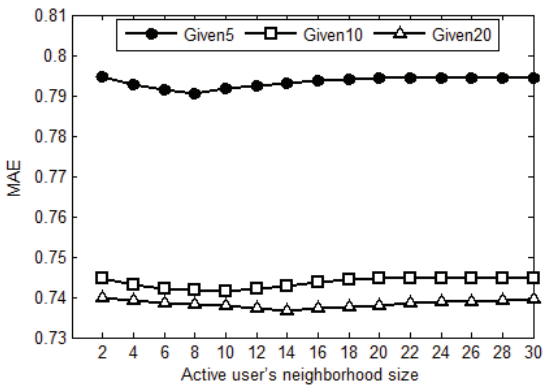

Figure 5. MAE against neighborhood size of the active user

but performs an increment trend when k_{u_a} goes beyond a certain value. This is because the ratings from less similar clusters are considered too much for the recommendations. Based on this observation, active user's neighborhood size affects the accuracy of CBSF.

- *Target item's neighborhood size (k_{i_t}):* Finally, in this step, we examine the effect of target item's neighborhood size on the performance of CBSF. Fig. 6 shows the performance for varying k_{i_t}.

As shown, the performance initially improves as we increase k_{i_t} from 2 to 18. After a threshold value, the performance declines because additional neighbors have a negative effect on rating prediction. When k_{i_t} is between 16 and 18, CBSF gets its lowest MAE. Similar to the previous experiment, we conclude that target item's neighborhood size also affects the accuracy of CBSF.

As we see, the neighborhood size in our experiments is relatively small. The reason is that we select the most similar clusters as the nearest neighbors. Therefore, the neighborhood size in our approach will be smaller than the traditional methods in which individual users (or items) are selected as neighbors.

C. Experimental results

In this section, we conduct experiments to confirm the advantages of our proposed approach over existing CF algorithms. We compare CBSF with memory-based, model-based and hybrid approaches. Table 3 shows the benchmark CF algorithms in our experiments.

Initially, we measure MAE and coverage for all the examined algorithms and compare the overall performance of our approach with other methods. Then, we compare the performance of all the examined algorithms in dealing with the sparsity and scalability problem.

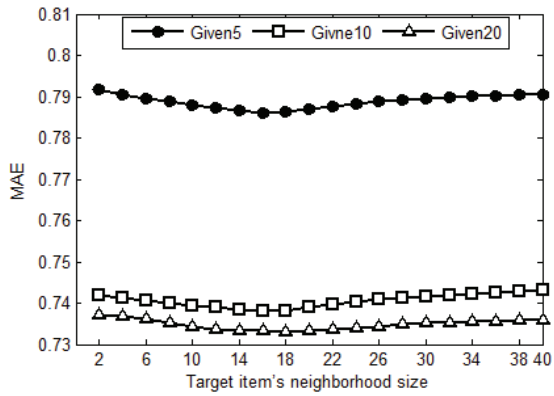

Figure 6. MAE against neighborhood size of the target item

TABLE 3. THE BENCHMARK CF ALGORITHMS

Algorithm	Abbreviation	Type	Refer to reference
User-Based collaborative filtering using PCC	UB-PCC	Memory-based	[17]
Item-Based collaborative filtering using PCC	IB-PCC	Memory-based	[19]
Similarity Fusion	SF	Memory-based	[12]
Personality Diagnosis	PD	Model-based	[15]
Aspect Model	AM	Model-based	[24]
Regression-Based	RB	Model-based	[25]
Cluster-Based Smoothing	CBS	Hybrid	[14]

1) Overall performance

In this section, we evaluate the overall performance of our approach and the benchmark algorithms in terms of MAE and coverage. The results are summarized in Table 4 and Table 5. In all algorithms, as the number of rated items for each user increases from 5 to 20, we see the decrease of MAE and the increase of coverage. This is due to the increase in the training set's size.

As shown, SF and CBSF have the lowest MAE values compared to other approaches. Actually, SF improves the prediction accuracy by fusing different sources of rating. Similarly, taking the advantage of SF accuracy, CBSF outperforms the remaining methods. Also, SF and CBSF have higher coverage than the other benchmarks. In these two approaches, smoothing from a pool of SUIR ratings reduces data sparsity and therefore increases the coverage.

As shown in Table 4, the differences between the accuracy of CBSF and SF are trivial. The differences between their coverage are also trivial (Table 5). These differences arise from using the clustering results in the CBSF. As mentioned before, clustering reduces the quality of recommendations but improves scalability. CBSF alleviate the drawbacks of

TABLE 4. COMPARISON BETWEEN PERFORMANCES OF DIFFERENT ALGORITHMS IN TERMS OF MAE

	MAE		
Algorithm	**Given5**	**Given10**	**Given20**
SF	**0.784**	**0.730**	**0.732**
CBSF	**0.789**	**0.735**	**0.736**
CBS	0.800	0.785	0.751
PD	0.809	0.786	0.760
AM	0.812	0.790	0.774
RB	0.819	0.783	0.770
UB-PCC	0.830	0.806	0.794
IB-PCC	0.852	0.820	0.801

TABLE 5. COMPARISON BETWEEN PERFORMANCES OF DIFFERENT ALGORITHMS IN TERMS OF COVERAGE

	Coverage		
Algorithm	**Given5**	**Given10**	**Given20**
SF	**99.850**	**100**	**100**
CBSF	**99.842**	**99.996**	**100**
CBS	99.801	99.979	99.961
PD	97.911	98.216	99.085
AM	92.689	94.258	97.328
RB	92.742	96.008	96.998
UB-PCC	94.415	95.384	98.125
IB-PCC	97.513	97.997	99.237

clustering model by fusing different sources of data. In the other words, similarity fusion helps to improve the quality of cluster-based recommendations. So, the accuracy and coverage of CBSF are close to the accuracy and coverage of SF. The main advantage of CBSF over SF is its lower computational complexity which will be discussed in the next part.

2) Impact on the scalability problem

For comparing the scalability of different methods, the run time of their on-line parts is measured. Notice that there is an off-line part in our proposed approach, which demands additional computational time. In the off-line part, CBSF finds the item clusters and user clusters. Since these computations are executed off-line, we do not count them in the recommendation time. We measure the average time (ms) that takes to provide recommendations to a test user (runtime per user). The results against the different size of the train set are presented in Fig. 7. The training sets are created using k-fold cross-validation (k = 2, 3, 4, 5, 10).

As expected, with increasing the size of the train set, the runtime increases for all methods. As shown, the efficiency of memory-based methods, UB-PCC and SF, are affected by the train set size. This is due to the on-line computation of the similarity between active user and other training users in the database. In contrast, the runtime of IB is almost stable. The reason is that IB creates the expensive similar-items table off-line. Fig. 7 indicates that CBS algorithm and CBSF are more scalable compared to other model-based approaches. This is due to the fact that these approaches create the clusters off-line and they are almost stable. Since in our approach the similarity is calculated between a target object and each cluster, the on-line similarity computation is speeded up. Fig. 7 clearly shows that CBSF has much less computational complexity and this is the main advantage of CBSF over SF. As shown in Fig. 7, CBSF needs a little more time than CBS algorithm which is negligible. The reason is that CBS only considers the most similar clusters to the active user, but CBSF also finds the most similar clusters to the target item. Although CBS has better execution time, but as we have seen, it has lower accuracy and coverage than CBSF.

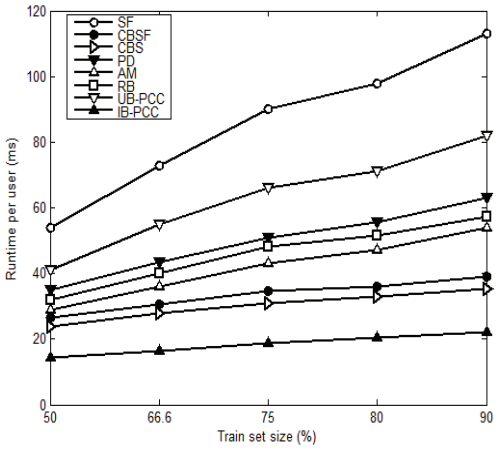

Figure 7. Impact on the scalability problem

3) Impact on the sparsity problem

The sparsity of a rating matrix has a significant impact on the performance of CF. We compare the performance of CBSF against the benchmark algorithms on different levels of sparsity. In each level, 10000 ratings are reduced from train set. Fig. 8 and Fig.9 shows the MAE and coverage against the different sparsity levels. The results show that the sparsity has a great effect on the performance of different algorithms. As expected, with increasing the sparsity level, the MAE increases for all methods (Fig. 8). Also, as shown in Fig. 9, the coverage will decrease when the sparsity level is increased. This is due to the reduction in the training set's size. As seen from Fig. 8 and Fig.9, CBSF and SF have the highest recommendation accuracy and the highest coverage under all levels of sparsity. The reason is that they use more of the data available in the user-item matrix and therefore they are more robust to data sparsity. So, the fusion model is effective at improving the accuracy and coverage of recommendations, even when only sparse data are available.

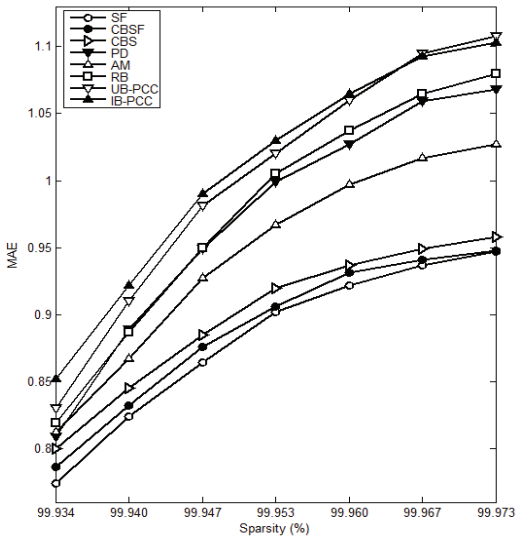

Figure 8. Impact on the sparsity problem in terms of MAE

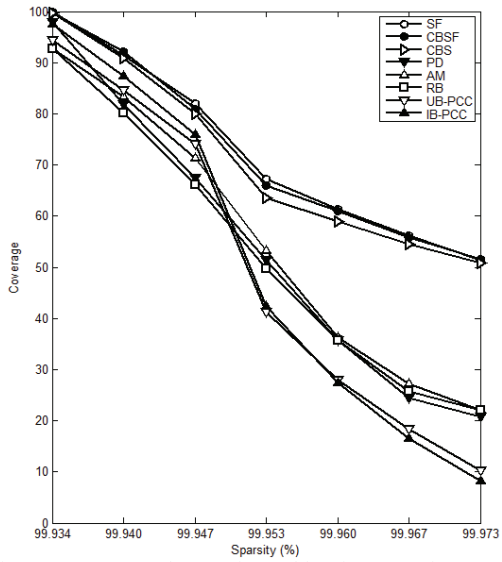

Figure 9. Impact on the sparsity problem in terms of coverage

V. CONCLUSIONS AND FUTURE WORKS

The ability to improve scalability without sacrificing accuracy is an outstanding challenge for collaborative recommender systems. In this paper, we presented a new Cluster-Based Similarity Fusion approach (CBSF) which is capable of scaling-up traditional CF as well as improving its accuracy. Our approach not only outperforms traditional CF in terms of prediction accuracy and coverage but also offers improvements in scalability and sparsity problems.

CBSF can be seen as a hybrid between memory-based and model-based approaches. It uses similarity fusion as a memory-based algorithm because SF is a robust approach to data sparsity. In order to cope with the scalability problem of SF, we integrated it with clustering models.

CBSF consists of four main steps. In the first step, users and items are clustered based on their rating similarities. In the second step, *k*-nearest clusters to the active user and target item are identified. In the third step, the cluster-based similarity fusion matrix is created based on the neighboring clusters. In the last step, the rating of active user for the target item is predicted based on this matrix.

Our experimental evaluations clearly show that CBSF approach produces better recommendation quality than pure model-based techniques. It can produce faster recommendations with the comparable quality to similarity fusion approach. Also, it can deal with the data sparsity problem of traditional collaborative filtering methods.

Our future work will focus on further validating the accuracy and performance of the hybrid CBSF approach on larger data sets. Also, we would like to apply our proposed approach to the real applications to test its performance.

REFERENCES

[1] L. Zhang, S. Yang, and M. Zhang, "E-commerce Website Recommender System Based on Dissimilarity and Association Rule," *TELKOMNIKA Indones. J. Electr. Eng.*, vol. 12, no. 1, pp. 353–360, 2014.

[2] X. Zhao and K. Ji, "Tourism e-commerce recommender system based on web data mining," in *Computer Science & Education (ICCSE), 2013 8th International Conference on*, 2013, pp. 1485–1488.

[3] V. S. Kumaran and A. Sankar, "Recommendation System for Adaptive E-learning using Semantic Net," 2013.

[4] P. Di Bitonto, M. Laterza, T. Roselli, and V. Rossano, "Recommendation of Collaborative Activities in E-learning Environments," in *Human-Computer Interaction. Applications and Services*, Springer, 2013, pp. 484–492.

[5] D. Sánchez-Moreno, A. B. Gil, and M. N. Moreno, "TV-SeriesRec: A Recommender System Based on Fuzzy Associative Classification and Semantic Information," in *Trends in Practical Applications of Agents and Multiagent Systems*, Springer, 2013, pp. 201–208.

[6] J.-H. Chang, C.-F. Lai, M.-S. Wang, and T.-Y. Wu, "A cloud-based intelligent TV program recommendation system," *Comput. Electr. Eng.*, 2013.

[7] R. Burke, "Hybrid recommender systems: Survey and experiments," *User Model. User-Adapt. Interact.*, vol. 12, no. 4, pp. 331–370, 2002.

[8] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *Knowl. Data Eng. IEEE Trans. On*, vol. 17, no. 6, pp. 734–749, 2005.

[9] J. B. Schafer, J. Konstan, and J. Riedi, "Recommender systems in e-commerce," in *Proceedings of the 1st ACM conference on Electronic commerce*, 1999, pp. 158–166.

[10] M. G. Vozalis and K. G. Margaritis, "Using SVD and demographic data for the enhancement of generalized collaborative filtering," *Inf. Sci.*, vol. 177, no. 15, pp. 3017–3037, 2007.

[11] U. Ceylan and A. Birturk, "Combining Feature Weighting and Semantic Similarity Measure for a Hybrid Movie Recommender System," presented at the The fifth International Workshop on Social Network Mining and Analysis (SNAKDD 2011), In conjunction with 17th ACM SIGKDD Conference on Knowledge Discovery and DataMining (KDD-2011), San Diego, CA, USA, 2011.

[12] J. Wang, A. P. De Vries, and M. J. Reinders, "Unifying user-based and item-based collaborative filtering approaches by similarity fusion," in *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, 2006, pp. 501–508.

[13] X. Su and T. M. Khoshgoftaar, "A survey of collaborative filtering techniques," *Adv. Artif. Intell.*, vol. 2009, p. 4, 2009.

[14] G.-R. Xue, C. Lin, Q. Yang, W. Xi, H.-J. Zeng, Y. Yu, and Z. Chen, "Scalable collaborative filtering using cluster-based smoothing," in *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, 2005, pp. 114–121.

[15] D. M. Pennock, E. Horvitz, S. Lawrence, and C. L. Giles, "Collaborative filtering by personality diagnosis: A hybrid memory-and model-based approach," in *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence*, 2000, pp. 473–480.

[16] H. Ma, I. King, and M. R. Lyu, "Effective missing data prediction for collaborative filtering," in *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, 2007, pp. 39–46.

[17] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "GroupLens: an open architecture for collaborative filtering of netnews," in *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, 1994, pp. 175–186.

[18] J. S. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," in *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, 1998, pp. 43–52.

[19] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of the 10th international conference on World Wide Web*, 2001, pp. 285–295.

[20] M. Deshpande and G. Karypis, "Item-based top-n recommendation algorithms," *ACM Trans. Inf. Syst. TOIS*, vol. 22, no. 1, pp. 143–177, 2004.

[21] Y. Li, L. Lu, and L. Xuefeng, "A hybrid collaborative filtering method for multiple-interests and multiple-content recommendation in E-Commerce," *Expert Syst. Appl.*, vol. 28, no. 1, pp. 67–77, Jan. 2005.

[22] Q. Shambour and J. Lu, "A hybrid multi-criteria semantic-enhanced collaborative filtering approach for personalized recommendations," in *Web Intelligence and Intelligent Agent Technology (WI-IAT), 2011 IEEE/WIC/ACM International Conference on*, 2011, vol. 1, pp. 71–78.

[23] J. Xiao, J. Gao, and G. Song, "Adaptive Recommendation Algorithm Based on the Bayesian-Network," in *Informatics and Management Science III*, Springer, 2013, pp. 145–151.

[24] T. Hofmann and J. Puzicha, "Latent class models for collaborative filtering," in *International Joint Conference on Artificial Intelligence*, 1999, vol. 16, pp. 688–693.

[25] S. Vucetic and Z. Obradovic, "A regression-based approach for scaling-up personalized recommender systems in e-commerce," *WEBKDD'00*, 2000.

[26] B. M. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering," in *Proceedings of the fifth international conference on computer and information technology*, 2002, vol. 1.

[27] R. Jin, L. Si, and C. Zhai, "A study of mixture models for collaborative filtering," *Inf. Retr.*, vol. 9, no. 3, pp. 357–382, 2006.

[28] P. Symeonidis, A. Nanopoulos, A. N. Papadopoulos, and Y. Manolopoulos, "Nearest-biclusters collaborative filtering based on constant and coherent values," *Inf. Retr.*, vol. 11, no. 1, pp. 51–75, 2008.

[29] D. Bridge and J. Kelleher, "Experiments in sparsity reduction: Using clustering in collaborative recommenders," in *Artificial Intelligence and Cognitive Science*, Springer, 2002, pp. 144–149.

[30] J. Kelleher and D. Bridge, "An accurate and scalable collaborative recommender," *Artif. Intell. Rev.*, vol. 21, no. 3–4, pp. 193–213, 2004.

[31] S. Gong, "A collaborative filtering recommendation algorithm based on user clustering and item clustering," *J. Softw.*, vol. 5, no. 7, pp. 745–752, 2010.

[32] C. Birtolo and D. Ronca, "Advances in Clustering Collaborative Filtering by means of Fuzzy C-means and trust," *Expert Syst. Appl.*, vol. 40, no. 17, pp. 6997–7009, 2013.

[33] P. Wang, "A Collaborative Filtering Recommendation Algorithm Based on Product Clustering," *Appl. Mech. Mater.*, vol. 267, pp. 87–90, 2013.

[34] Z. Deng and J. Wang, "Collaborative Filtering Algorithm Based on User Clustering," *Appl. Mech. Mater.*, vol. 411, pp. 1044–1048, 2013.

[35] A. Bilge and H. Polat, "A scalable privacy-preserving recommendation scheme via bisecting< i> k</i>-means clustering," *Inf. Process. Manag.*, vol. 49, no. 4, pp. 912–927, 2013.

[36] C.-F. Tsai and C. Hung, "Cluster ensembles in collaborative filtering recommendation," *Appl. Soft Comput.*, vol. 12, no. 4, pp. 1417–1425, 2012.

[37] K. Yu, A. Schwaighofer, V. Tresp, X. Xu, and H.-P. Kriegel, "Probabilistic memory-based collaborative filtering," *Knowl. Data Eng. IEEE Trans. On*, vol. 16, no. 1, pp. 56–69, 2004.

[38] B. S. Suryavanshi, N. Shiri, and S. P. Mudur, "A fuzzy hybrid collaborative filtering technique for web personalization," in *Proc. of 3rd Workshop on Intelligent Techniques for Web Personalisation (ITWP'05)*, 2005.

[39] Y. Chuan, X. Jieping, and D. Xiaoyong, "Recommendation algorithm combining the user-based classified regression and the item-based filtering," in *Proceedings of the 8th international conference on Electronic commerce: The new e-commerce: innovations for conquering current barriers, obstacles and limitations to conducting successful business on the internet*, New York, NY, USA, 2006, pp. 574–578.

[40] S. K. L. Al Mamunur Rashid, G. Karypis, and J. Riedl, "ClustKNN: a highly scalable hybrid model-& memory-based CF algorithm," *Proceeding WebKDD*, 2006.

[41] Q. Wang, X. Yuan, and M. Sun, "Collaborative filtering recommendation algorithm based on hybrid user model," in *Fuzzy Systems and Knowledge Discovery (FSKD), 2010 Seventh International Conference on*, 2010, vol. 4, pp. 1985–1990.

[42] X. Chen, X. Liu, Z. Huang, and H. Sun, "Regionknn: A scalable hybrid collaborative filtering algorithm for personalized web service recommendation," in *Web Services (ICWS), 2010 IEEE International Conference on*, 2010, pp. 9–16.

[43] D. Zhang, J. Cao, J. Zhou, M. Guo, and V. Raychoudhury, "An efficient collaborative filtering approach using smoothing and fusing," in *Parallel Processing, 2009. ICPP'09. International Conference on*, 2009, pp. 558–565.

[44] S. G. Moghaddam and A. Selamat, "A scalable collaborative recommender algorithm based on user density-based clustering," in *2011 3rd International Conference on Data Mining and Intelligent Information Technology Applications (ICMiA)*, 2011, pp. 246–249.

[45] L. Kaufman and P. J. Rousseeuw, *Finding groups in data: an introduction to cluster analysis*, vol. 344. Wiley-Interscience, 2009.

[46] A. P. Reynolds, G. Richards, and V. J. Rayward-Smith, "The application of k-medoids and pam to the clustering of rules," in *Intelligent Data Engineering and Automated Learning–IDEAL 2004*, Springer, 2004, pp. 173–178.

[47] M. C. N. Barioni, H. L. Razente, A. J. Traina, and C. Traina Jr, "An efficient approach to scale up k-medoid based algorithms in large databases," in *Brazilian Symposium on Databases (SBBD), Florianópolis, SC, Brazil. SBC*, 2006, pp. 265–279.

[48] P. Melville and V. Sindhwani, "Recommender systems," *Encycl. Mach. Learn.*, vol. 1, pp. 829–838, 2010.

[49] O. Nouali and A. Belloui, "Using semantic web to reduce the cold-start problems in recommendation systems," in *Applications of Digital Information and Web Technologies, 2009. ICADIWT'09. Second International Conference on the*, 2009, pp. 525–530.

[50] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, "Evaluating collaborative filtering recommender systems," *ACM Trans. Inf. Syst. TOIS*, vol. 22, no. 1, pp. 5–53, 2004.

Faezeh S. Gohari is a M.Sc. student in Electronic Commerce at K.N. Toosi University of Technology, Tehran, Iran. She received her bachelor's degree in computer engineering from Elm O Farhang University, Tehran, Iran. Her current primary research interests include Information Systems Management, Semantic Web, Data Mining and e-Marketing.

Mohammad J. Tarokh is an associate professor in the department of Industrial Engineering at K.N. Toosi University of Technology, Tehran, Iran. He received his B. Sc. from Sharif University of Technology in Tehran, M.Sc. from University of Dundee in UK and Ph.D. from University of Bradford, UK. His main research interests are in knowledge management, business intelligence, customer relationship management and supply chain management.