Neurocomputing 191 (2016) 388–397

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

# An exploration of improving prediction accuracy by constructing a multi-type clustering based recommendation framework

Xiao Ma, Hongwei Lu, Zaobin Gan *, Qian Zhao

*School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China*

ARTICLE INFO

*Article history:*
Received 4 June 2015
Received in revised form
28 December 2015
Accepted 17 January 2016
Communicated by Peng Cui
Available online 18 February 2016

*Keywords:*
Recommender Systems
Multi-type Clustering
Social Trust Relationships
Collaborative Filtering

ABSTRACT

Existing clustering-based recommendation methods generally focus on the clustering of users, items or social trust relationships. Although demonstrated to be efficient and scalable to large-scale datasets, these methods are sensitive to the quality of clustering and still suffer from the problem of low accuracy. In order to solve this issue, in this paper, we propose a multi-type clustering based recommendation framework which systematically considers the trust-based user clustering, similarity-based user clustering and similarity-based item clustering to further improve the recommendation accuracy. A SVD (Singular Value Decomposition) signs-based community mining method is utilized to process the trust and distrust matrix in order to discover the trust-based user clusters. The PLSA (Probabilistic Latent Semantic Analysis)-based model is employed to explore the similarity-based user and item clusters. Then a clustering-based trust regularization term is proposed to incorporate the trust-based user clusters into the matrix factorization model. Comparative experiments on two real-world datasets demonstrate that our approach can better address the issues of data sparsity and cold start, and outperforms other state-of-the-art methods in terms of RMSE and MAE.



## 1. Introduction

Recommender Systems (RSs) as an indispensable type of information filtering technique dealing with information overload have attracted lots of attention in the past decades. Such systems help users explore their interests in various domains, including movies, music, books, and academics. Most of recommender systems are based on Collaborative Filtering (CF), which is a technique that automatically predicts the interests of an active user by collecting rating information from other similar users or items [1].

Typically, collaborative filtering based RSs can be classified into memory-based methods and model-based methods. Memory-based CF methods explore the entire user–item rating matrix to find similar neighbors (also known as nearest neighbors) for a given user or item. Model-based CF methods including matrix factorization [2], latent semantic models [3], Bayesian Inference [4] usually learn parameters of a model offline and make prediction for unrated items with the generated models. These methods do not need to explore the rating matrix and only store the model parameters.

Although traditional CF models have been successful in many areas, they all have to face several critical problems: data sparsity, scalability and cold-start. In order to solve these inherent problems, the clustering-based recommendation approaches have been proposed, which have been proved to improve the quality of recommender systems [5,6]. Different clustering strategies can be performed based on users, items or trust relationships, which results in several sub-matrices of the entire user–item rating matrix and groups the well-connected users or items into the same clusters. Then traditional collaborative filtering approaches can be applied to the sub-matrices, which alleviate the data sparsity and scalability problems to a large extent. Fig. 1 is a toy example of diverse types of clustering. There are six different users and items in this figure. In Fig. 1(a), users are clustered based on the similarity among users, thus users $u_1$, $u_2$, $u_3$ and $u_4$ are clustered into the similarity-based user cluster. In Fig. 1(b), users are clustered based on the trust relationships, thus users $u_3$, $u_4$, $u_5$ and $u_6$ who have trust connections between each other are clustered into the trust-based user cluster. Similarly, two similarity-based item clusters are obtained as shown in Fig. 1(c).

As shown in Fig. 1, existing clustering-based recommendation algorithms are mainly based on single type of clustering information, such as similarity-based user clustering [7], similarity-based item clustering [8], and trust-based user clustering [9,10]. Recently, Guo et al. [6] develop a clustering method through which users are iteratively clustered from the views of both ratings and

* Corresponding author.
*E-mail addresses:* cindyma@hust.edu.cn (X. Ma), luhw@hust.edu.cn (H. Lu), zgan@hust.edu.cn (Z. Gan), zqhuster@163.com (Q. Zhao).

http://dx.doi.org/10.1016/j.neucom.2016.01.040
0925-2312/

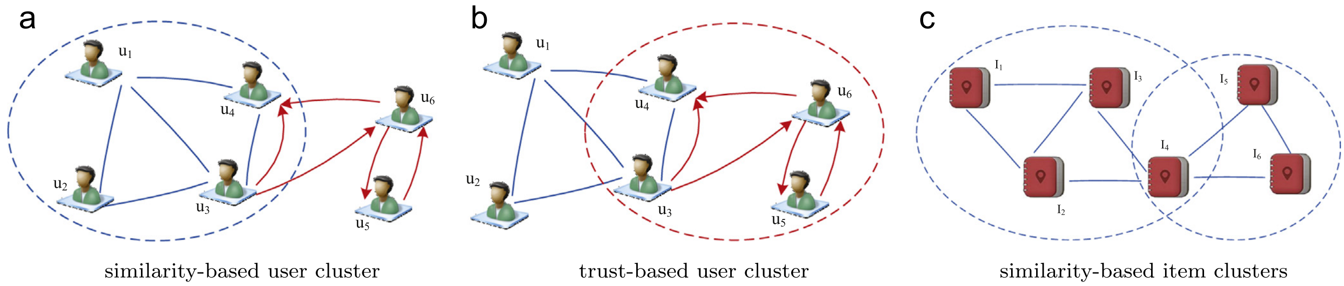



**Fig. 1.** A toy example of diverse types of clustering: Dotted circles denote the formed user/item clusters, undirected blue lines represent the similarity between two users, while directed red line represent the trust relationship between users. (a) User clustering based on the similarity between users; (b) user clustering based on the trust relationships between users; (c) item clustering based on the similarity between items. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

social trust relationships. There are also some works focusing on clustering users and items simultaneously (called co-clustering), which further improve the accuracy of clustering-based recommender systems [11–13]. It is known to us that recommendation performance is sensitive to the clustering results. Similar users can only be selected from the fixed size of cluster members, and in general a fewer number of similar users can be identified compared with the whole space. Therefore clustering-based methods still suffer from relatively low accuracy [6,9], which severely hinders the practical use of clustering-based methods in recommender systems.

In order to solve the issue, in this paper, we design a unified framework named as *MCRec* (*M*ulti-type *C*lustering based *Rec*ommender systems) which systematically combines multi-type clustering sources, namely, the similarity-based user clustering, the similarity-based item clustering, and the trust-based user clustering, to improve the performance of clustering-based recommender systems. The proposed MCRec can be characterized as follows.

Firstly, as co-clustering is extremely important when dealing with large, sparse data matrices [11], MCRec generates submatrices by clustering users and items simultaneously which are so much denser than the original user–item matrix that will greatly alleviate the data sparsity problem. Secondly, for cold start users with only a few user-generated information, traditional clustering-based methods cannot capture their personal tastes accurately. Trust clustering classifies the densely connected trust users into the same trust cluster, and provides an additional data source to enhance the clustering-based recommender systems by analyzing the preferences of their trust neighbors in the same trust community [14]. Thirdly, more information can be used to learn users' preferences by proposing a unified recommendation framework making use of multiple clustering sources. Hence intuitively, the recommendation performance will be improved, as we will demonstrate later.

Our objective is to incorporate the multi-type clustering sources into a unified recommendation framework to improve the quality of clustering-based recommender systems. Thus we will elaborate these two questions in our paper: (1) *How to combine the multi-type clustering information (as shown in Fig. 1) into a unified recommendation framework?*(2) *How can this combination further improve the accuracy of recommender systems?*

In summary, the main contributions of this paper are:

- We propose the multi-type clustering problem in the domain of clustering-based recommender systems and present a method well-suited for this problem.
- A SVD signs based community mining method is utilized to process the trust and distrust relationship matrix aiming at discovering the trust-based user clusters. And the PLSA-based model is employed to explore the similarity-based user clusters and similarity-based item clusters.
- A clustering-based trust regularization term is proposed to incorporate the trust-based user clusters into the matrix factorization model.
- Extensive of comparative experiments are conducted to explore how can this combination improve the performance of recommender systems. Real datasets based experiments demonstrate the effectiveness of our method in comparison with state-of-the-art methods.

The rest of this paper is organized as follows. Section 2 gives an introduction of the related work on trust-based and clustering-based recommender systems. The proposed multi-type clustering framework is discussed in detail in Section 3. Section 4 includes our experimental algorithm in alleviating the data sparsity and cold start problem in rating matrix and generating more accurate recommendations. Finally, Section 5 concludes this study with future work.

## 2. Related work

In this section, we review the approaches to recommender systems, including trust-based recommender systems and clustering-based recommender systems.

### 2.1. Trust-based recommender systems

Trust-based recommender systems exploit trust information explicitly expressed by users to help model user preferences. Generally, trust-based recommender systems can be classified into memory-based methods and model-based methods. Memory-based trust-aware recommender systems use memory-based collaborative filtering methods as their basic models. They search the trust networks to obtain trust neighbors for a given user. And the correlated users for the given user can be the trust neighbors, similar neighbors, or a combination of similar neighbors and trust neighbors. For example, there are a Trust Metric module and a Similarity Metric module in the architecture of trust-aware recommender systems proposed by Massa and Avesani [15]. Therefore, the weights for identifying neighbors can be generated by trust metrics or similar metrics. They find that trust is useful in helping alleviating the issues of traditional collaborative filtering such as data sparsity and cold start. Jamali and Easter [16] propose the TrustWalker, a random walk model to combine trust-based and item-based collaborative filtering approach for recommendation, which clearly outperforms the traditional collaborative filtering approaches in terms of precision.

Model-based trust-aware recommender systems use model based collaborative filtering methods as their basic models. Matrix factorization techniques are widely used in this category. Ma et al. [17] propose a trust-aware recommendation model on the basis of the matrix factorization model. Their method combines a basic matrix factorization approach and a trust-based approach, and naturally fuses the users' tastes and their trusted friends' favors together. Jamali and Easter [18] introduce SocialMF, which is one of the most popular social recommendation algorithms. The authors incorporate the mechanism of trust propagation into their probabilistic matrix factorization based model, which is a crucial phenomenon in social sciences and increases the rating accuracy to a large extent. Jiang et al. [19,20] incorporate the social context information (individual preference and interpersonal influence) into the social recommendation framework to further enhance the performance of recommender systems. Forsati et al. [21] provide a matrix factorization-based model for recommendation in social rating networks that properly incorporates both trust and distrust relationships. Later, they propose PushTrust simultaneously leveraging trust, distrust and neutral relations in their recommendation framework [22]. The experimental results show that their methods are effective in upgrading the quality of recommendations and mitigating the data sparsity and cold-start users issues by exploiting trust and distrust relations.

In conclusion, trust-based recommender systems are effective in alleviating the inherent problems of traditional recommender systems and can further improve the recommendation performance.

### 2.2. Clustering-based recommender systems

Clustering-based recommendation methods group users or items into clusters, which offer a new way to identify the neighborhood and are scalable to large-scale datasets [5]. As a dimension-reduction method, clustering techniques are able to alleviate the sparsity problem of rating data (however, it may also lead to the low accuracy problem).

Most early works focus on clustering users or items separately and are all relied on the clustering of similarity. For instance, Sarwar et al. [7] cluster the complete user set based on the user–user similarity and use the cluster as the neighborhood. However, they uncover that the accuracy is decreased around 5% in comparison with the conventional kNN based CF method. While OConnor and Herlocker [8] use clustering algorithms to partition the set of items based on user rating data, predictions are computed independently within each partition. Xue et al. [23] apply the K-means algorithm to cluster users, and select the top-K most similar users from each cluster as the nearest neighbors. Ji et al. [24] focus on discovering the implicit similarity among users and items. They first cluster user/item latent factor vectors into user/item cluster-level factor vectors. Then they use the cluster-level factor vectors to compress the original approximation into a cluster-level rating-pattern.

Some other works consider clustering users and items simultaneously (called co-clustering). Some typical works are those proposed by [11–13]. George et al. [13] obtain the user and item neighborhoods via co-clustering and map users and items into clusters simultaneously, thus each cluster becomes much denser than the entire rating matrix. Then Xu et al. [11] extend the idea of George by clustering users and items into several clusters at the same time. Pereira et al. [12] provide a hybrid recommendation method to address the cold start problem based on the simultaneous co-clustering and learning of user and item attributes.

Recently, some works are exploring the clustering of the social trust information to further enhance the Recommender Systems [9,10,6,25]. Pham et al. [9] run a density-based clustering algorithm on trust relationships to cluster users into trust-based user clusters. Ma et al. [10] combine trust clustering and collaborative filtering to further improve the recommendation accuracy. Guo et al. [6] develop a multiview clustering method concentrating on clustering users using both ratings and trust information. They use the k-medoids to acquire clusters and employ a support vector regression model to determine a prediction for a given item, based on user-, item- and prediction-related features.

In summary, existing clustering-based recommender systems mainly focus on single type of clustering information and still suffer from the issue of low accuracy. Parapar and Bellogin [26] point out that a better clustering algorithm can further improve the recommendation performance. This motivates us to develop a better approach to improve the performance of existing clustering-based recommender systems.

## 3. The MCRec framework

In this section, we details about our proposed MCRec framework. The notations used in this paper are defined in Section 3.1. A brief description of our framework is presented in Section 3.2. Then we introduce how to get the trust-based user clusters in Section 3.3. The similarity-based user and item clusters are generated in Section 3.4. Section 3.5 discusses about how to make predictions with multi-type clusters.

### 3.1. Preliminary

Let $\mathcal{U}=\{u_1,u_2,\dots,u_M\}$ and $\mathcal{V}=\{v_1,v_2,\dots,v_N\}$ be the sets of users and items respectively, where $M$ is the number of users and $N$ is the number of items. $\mathcal{R}=\{R_{i,j}\in\{1,2,3,4,5\}\,|\,u_i\in\mathcal{U},v_j\in\mathcal{V}\}$ constructs an $M\times N$ user–item rating matrix. $T\in\mathbb{R}^{M\times M}$ denotes the user–user trust relationships where $T_{i,f}=1$ if $u_i$ trusts $u_f(u_f\in\mathcal{U})$ and $T_{i,f}=-1$ if $u_i$ distrusts $u_f$.

### 3.2. Framework description

The brief description of our MCRec framework is shown in Fig. 2. As we can see, the input are the user–item rating matrix and the trust network. Firstly, we cluster the trust network to get the trust-based user clusters, and the similarity-based user and item clusters are generated by performing clustering algorithm on the user–item rating matrix. Thus users and items with similar interest topics are allocated into the same co-clusters. Then by proposing a clustering-based trust regularization term, we construct a multi-type clustering based recommendation model which incorporates three different types of clusters. Finally, predictions are merged for user–item pairs appearing in multiple co-clusters simultaneously.

### 3.3. Trust-based user clustering

In this section, we cluster users using the social trust relationships. In online social networks, there are positive relationships (i.e. trust, like) and negative relationships (i.e. distrust, dislike). The research of signed network mining believes that a signed network is often composed of communities. Users share positive relationships within the same community and negative relationships exist between different communities [27]. Inspired by this work, we argue that trust network will follow the so-called 'balanced' patterns as a special type of signed network [27,28]. That is to say, users in the same cluster trust each other and users between different clusters have distrust relationships with each other. In this paper, we choose to use the SVD signs proposed by [29] to partition users into different clusters according to how they are trusted and distrusted by others. Note that matrix $T$ is different from the adjacency matrix in [29], which describes both the connection and the trust or distrust relationships between two users. By decomposing the matrix with truncated SVD, we can cluster the users based on how they trust and distrust others and how they are trusted and distrusted by others with less dimensions. The decomposed trust relationship matrix with rank $K$ can be represented by:

$$\tilde{T}_{|U|\times|U|}=P_{|U|\times K}S_{K\times K}Q^{\mathrm{T}}_{|U|\times K} \tag{1}$$

where $P$, $Q$, $S$ represent the left singular matrix, right singular matrix and singular values matrix respectively. Matrix $\tilde{T}$ is the most possible rank $K$ approximation to matrix $T$, $K \lll rank(T)$. According to [29], the singular values can be plotted on a line graph in a descending order, and the turning point of the line can be chosen as the best $K$. Therefore, the elements of matrix $S$ are the $K$ dominant singular values. The rows of matrix $P$ and $Q$ can be regarded as the coordinates of the participants in the $K$ dimensional spaces.

Since the matrix $T$ is asymmetric, clustering methods by rows of $P$ and $Q$ have different meanings. If the rows of matrix $P$ with the same sign patterns on the $K$ dimensions are classified into one cluster, this may lead to up to $2^K$ clusters (actually may fewer than it). The sign patterns of the rows of $P$ clusters the users by how they trust and distrust others. Similarly, the sign patterns of the rows of $Q$ are also applicable and this clusters the users by how they are trusted and distrusted by others.

Fig. 3 is a toy example of the result of SVD signs based trust clustering by using sign patterns of the first two left singular vectors. In the instance, the user number is 10, the rank $K$ is set to be 2. According to the sign patterns of $\{(+,+),(+,-),(-,-)\}$ of the rows of $P_{10\times 2}$ (on the right side of Fig. 3), the original trust network is classified into three parts as circled by different colors of dotted line. From the result we can see that each user belongs to a unique cluster, and users in a same cluster trust each other while distrust each other between different clusters.

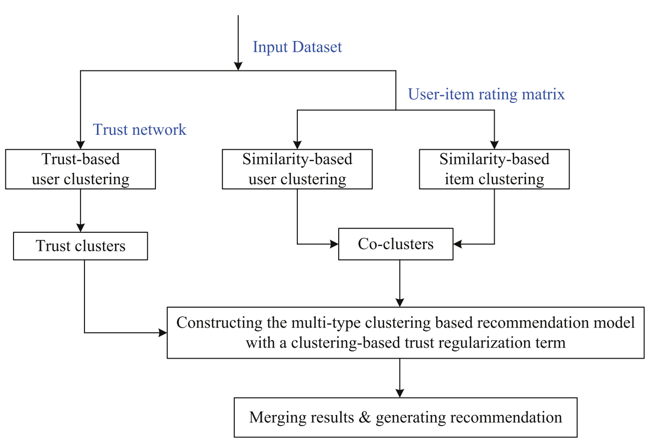



**Fig. 2.** The proposed MCRec framework.

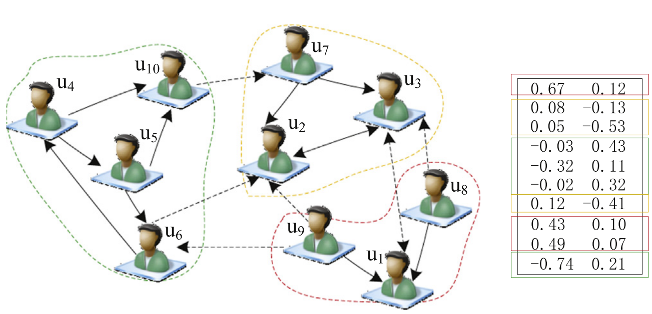



**Fig. 3.** SVD signs based trust clustering. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

### 3.4. Similarity-based user clustering and item clustering

In this section, users and items are clustered based on the rating information. In our framework, we employ the PLSA model [30] to discover similarity-based user and item clusters. The reasons we use the PLSA model are (1) it can mine the latent interest feature of users and items, so that it can get more innate character information from limited datasets. (2) The latent topic of PLSA can be well-aligned with the point of interest in the real word. So it is more interpretable and representative. In our work, the basic PLSA model is adjusted to adapt to the user–item matrix by regarding users as words and items as documents. We use a variable $z_w$ to denote the latent topics where $z_w \in \{z_1, z_2, \ldots, z_W\}$, and $W$ is the number of topics. Then users and items with a similar topic could be mapped into the same subgroup. Parameters in our model are $\{p(u_i|z_w), p(z_w|v_j)\}$. We set all the parameters as $\Theta$ for succinct in the following paragraph. The objective function is defined as follows:

$$\log p(U|\Theta)=\sum_{i=1}^{M}\sum_{j=1}^{N}R_{ij}\cdot\log\left[\sum_{w=1}^{W}p(u_i|z_w)p(z_w|v_j)\right] \tag{2}$$

Note that in Eq. (2), parameters $p(u_i|z_w)$ and $p(z_w|v_j)$ are very useful, which denote the probabilistic distribution of a user $u_i$ and a item $v_j$ on a topic $z_w$, respectively [3,31]. Therefore, we can cluster users and items using these two parameters. In our work, EM algorithm is employed to estimate the parameter $\Theta$. Afterwards, the probability distribution of $p(u_i|z_w)$ and $p(z_w|v_j)$ are produced, see the following equation:

$$P(z_w|u_i,v_j)=\frac{p(u_i|z_w)p(z_w|v_j)}{\sum_{l=1}^{W}p(u_i|z_l)p(z_l|v_j)} \tag{3}$$

Now we take the user clustering as example. In order to group each user into more than one topic cluster, we introduce a clustering threshold as $\varepsilon_u$. Given a user $u_i$, if $p(u_i|z_w) > \varepsilon_u$, $u_i$ is considered to be interested in topic $z_w$. Thus users with similar topics are assigned to the same clusters, and the similarity-based user clusters are produced. Similarly, we adopt the same mechanism of processing users to assign all items to corresponding clusters, and the similarity-based item clusters are generated. Therefore, the observed user–item pairs are allocated into different clusters which can be named as co-clusters [13]. As shown in Fig. 4, each user and item can belong to multiple co-clusters.

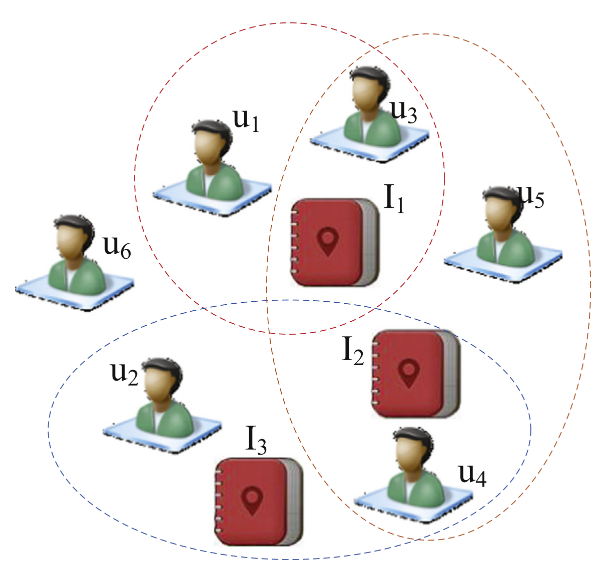



**Fig. 4.** Co-clustering: the dotted circles represent co-clusters which is the result of both similarity-based user clustering and similarity-based item clustering.

### 3.5. Recommendation with multi-type clusters

Till now the trust-based user clusters, similarity-based user and item clusters are obtained. In this section, we will answer our first question proposed in Section 1: *How to combine the multi-type clustering information (as shown in Fig. 1) into a unified recommendation framework*. The problem is resolved in two steps.

Firstly, we detail about how to incorporate the users appearing in trust-based user clusters into the traditional matrix factorization model in Section 3.5.1. Then we introduce how to make recommendations based on the matrix factorization model by combining user clusters and item clusters in Section 3.5.2

#### 3.5.1. Incorporating the trust-based user clusters

Matrix factorization model is one of the most popular and successful models in recommender systems [2]. In this paper, we use matrix factorization as our basic model. As aforementioned in Section 3.4, the user–item pairs are allocated into different co-clusters after performing the PLSA algorithm. Let $R_w \in \mathbb{R}^{M_w \times N_w}$ denotes the rating matrix of the $w$th co-cluster, where $w \in \{1, 2, \ldots, W\}$. $M_w$ and $N_w$ are the number of users and items in each co-cluster, respectively. Let $U_w \in \mathbb{R}^{M_w \times d}$ and $V_w \in \mathbb{R}^{V_w \times d}$ denote the latent feature matrices in the $w$th co-cluster, where $d$ denotes the dimension of latent feature. Adopting the matrix factorization model in different co-clusters, the model is trained on rating data by minimizing the square error:

$$\mathcal{L}_1 = \sum_{i=1}^{M_w} \sum_{j=1}^{N_w} I_{ij}^w (R_{ij}^w - U_i^w (V_j^w)^T)^2 + \lambda (\|U_i^w\|_F^2 + \|V_j^w\|_F^2) \tag{4}$$

where $I$ is an indicator function of which $I_{ij}^w = 1$ means user $i$ rated item $j$ in the $w$th co-cluster, and 0 otherwise. The second term is regularization term and $\|*\|_F$ is the Frobenius norm, which is introduced to avoid overfitting. The parameter $\lambda (\lambda > 0)$ controls the extent of regularization.

Empirical study reveals a correlation between trust and user similarity [25]. In our implementation, we assume that users share more similarity in the same trust cluster than users in different trust clusters. More specifically, if user $u_i$'s trust neighbors in the $g$th trust-based user cluster is $\mathcal{TN}_+^g(i)$, we could make a hypothesis that $u_i$'s taste $U_i$ should be close to the average taste of all the trust neighbors in $\mathcal{TN}_+^g(i)$, which can be represented as $\frac{1}{|\mathcal{TN}_+^g(i)|} \sum_{f \in \mathcal{TN}_+^g(i)} U_f$, where $f \in \{1, 2, \ldots, M\}$, $g \in \{1, 2, \ldots, G\}$ and $G$ represents the total number of trust clusters. Based on this intuition, following the approach described in [32], we propose a clustering-based trust regularization term to incorporate the trust-based user clusters into the matrix factorization model:

$$\alpha \sum_{i=1}^{M_w} \sum_{g=1}^{G} \|X_{g,i} \left( U_i^w - \frac{1}{|\mathcal{TN}_+^g(i)|} \sum_{f \in \mathcal{TN}_+^g(i)} U_f \right) \|_F^2 \tag{5}$$

where $\alpha > 0$ and $\alpha$ is a constant controlling the extent of trust regularization. $X_{g,i}$ is 1 if user $u_i$ has trust neighbors in the $g$-th trust-based user cluster, and 0 otherwise.

Thus, we incorporate the trust-based user clusters into the matrix factorization model. The objective function can be formulated as:

$$\begin{aligned} \mathcal{L}_2 = & \sum_{i=1}^{M_w} \sum_{j=1}^{N_w} I_{ij}^w \left( R_{ij}^w - U_i^w (V_j^w)^T \right)^2 \\ & + \alpha \sum_{i=1}^{M_w} \sum_{g=1}^{G} \|X_{g,i} \left( U_i^w - \frac{1}{|\mathcal{TN}_+^g(i)|} \sum_{f \in \mathcal{TN}_+^g(i)} U_f \right) \|_F^2 \\ & + \lambda (\| U_i^w \|_F^2 + \|V_j^w\|_F^2) \end{aligned} \tag{6}$$

The objective function can be minimized by the gradient descent approach [2]. More formally, the gradients of the objective function with respect to the feature vectors $U_i^w$ and $V_j^w$ are shown as Eqs. (7) and (8) respectively, where $\mathcal{TN}_-^g(i)$ represents $u_i$'s trusted neighbors in the $g$th trust-based user cluster. $X_{g,h}$ is 1 if user $u_h$ has trusted neighbors in the $g$th trust cluster, and 0 otherwise:

$$\begin{aligned} \frac{\partial \mathcal{L}_2}{\partial U_i^w} = & \sum_{j=1}^{N_w} I_{ij}^w ((U_i^w)^T V_j^w - R_{ij}^w) V_j^w + \lambda U_i^w \\ & + \alpha \sum_{g=1}^{G} X_{g,i} \left( U_i^w - \frac{1}{|\mathcal{TN}_+^g(i)|} \sum_{f \in \mathcal{TN}_+^g(i)} U_f \right) \\ & + \alpha \sum_{g=1}^{G} \sum_{h \in \mathcal{TN}_-^g(i)} \frac{X_{g,h}}{|\mathcal{TN}_+^g(i)|} \left( \frac{1}{|\mathcal{TN}_+^g(i)|} \sum_{f \in \mathcal{TN}_+^g(i)} U_f - U_h \right) \end{aligned} \tag{7}$$

$$\frac{\partial \mathcal{L}_2}{\partial V_{j^w}} = \sum_{i=1}^{M_w} I_{ij}^w ((U_i^w)^T V_j^w - R_{ij}^w) U_i^w + \lambda V_j^w \tag{8}$$

#### 3.5.2. Merge the results derived from all clusters

When model training is finished, the latent feature vectors $U_i^w$, $V_j^w$ of each cluster are produced, hence we should solve the problem of how to get the results from all clusters and merge them into a unified user–item matrix. Given a user–item pair $(u_i, v_j)$, if the user and the item are in the same cluster (namely the co-cluster), the prediction can be computed by:

$$\hat{R}_{ij}^w = \begin{cases} \tilde{r} + U_i^w (V_j^w)^T, & \text{if } u_i \in C^w \cap v_j \in C^w \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

where $\tilde{r}$ is the global bias term which can be obtained by the average value of the observed training ratings. $C^w$ represents the $w$th co-cluster. Since users and items may belong to more than one co-cluster as shown in Fig. 4, we have to merge the predicted ratings derived from multiple co-clusters. Therefore, the final results are generated by a linear average as $\hat{R_{ij}} = \frac{\sum_w \hat{R_{ij}^w}}{L}$, where $L$ is the number of co-clusters in which user $u_i$ and $u_j$ appear simultaneously. Otherwise, we do not predict rating for the user–item pair.

### 3.6. Complexity analysis

For our proposed MCRec, the main computation is to identify different types of clusters and compute the gradients of the objective function $\mathcal{L}_2$ as discussed in Sections 3.3, 3.4 and 3.5, respectively. A detailed explanation can be given as follows.

Firstly, since the trust relationships in both the datasets are quite sparse, we use a sparse matrix to store the trust relationships in the experiment. The computational complexity for truncated SVD signs method is $O(|T| * |K|)$, where $|T|$ is the number of trust relationships and $|K|$ is the number of singular values. Generally, $K$ is not big, such as the best $K$ is set to be 2 for Epinions dataset and 1 for Flixster dataset in our work.

Secondly, the time complexity for producing co-clusters is $O(|W| * |R|)$, where $|W|$ is the number of co-clusters and $|R|$ is the total number of ratings. In the experiment, $W$ is always quite small, for instance, our propose method yields the best results when $W$ is set to be 10.

Thirdly, the main computation of gradient methods is evaluating the objective functions of $\mathcal{L}_2$ and their gradients against variable. The computational complexities of evaluating the objective function $\mathcal{L}_2$ are $O(|R^W| * |d| + |U^W| * |\bar{f}| * |G| * |d|)$, where $w \in \{1, 2, \ldots, W\}$ and $W$ is the number of co-clusters. $|G|$ is the number of trust clusters. $|R^W|$ represents the number of nonzero entries in matrix $R^W$. $|d|$ is the number of user features. $|U^W|$ represents the number of users in the $w$th co-cluster. $|\bar{f}|$ denotes the average number of friends from the $g$th trust cluster that a user in the $w$th co-cluster trust. Since online social network graphs always fit the power law distribution, a large long tail of users only has few trusted users. It is indicated that the values of $|\overline{f}|$ is relatively small. Generally, $|U^w|*|\overline{f}|*|G|\ll|R^w|$.

The computational complexities for the gradients $\frac{\partial \mathcal{L}_2}{\partial U}$ and $\frac{\partial \mathcal{L}_2}{\partial V}$ in Eqs. (7) and (8) are $O(|R^w|*|d^2|+|U^w|*(|\overline{f}|+|\overline{h}|)*|G|*|d|)$ and $O(|R^w|*|d^2|)$, respectively. Where $|\overline{h}|$ is the average number of users in the gth trust cluster who trust a user from the $w$th co-cluster. Since $|\overline{f}|$, $|\overline{h}|$ and $|G|$ are small, the total computational complexity in one iteration is around $O(|R^w|*|d^2|+|R^w|*|d|)$. Typically, $R^w$ is much smaller than the original user–item matrix $R$ which is utilized in the BaseMF algorithm.

To sum up, the overall time complexity of the MCRec method is $O(|T|*|K|)+O(|W|*|R|)+O(|R^w|*|d^2|+|R^w|*|d|)$, which indicates that the computational time of our method is linear with respect to the number of observations in the user–item matrix and the number of trust relationships in the trust network. That is to say, our multi-type clustering based recommendation algorithm is efficient in dealing with large scale datasets.[1]

## 4. Experiments

In this section, comparative experiments are conducted on two real-world datasets to study the research question: whether incorporating multi-type clustering information can improve the performance of recommendation in terms of accuracy.

### 4.1. Datasets

Two real-world datasets are used in the experiments, namely Epinions[2] and Flixster.[3] Epinions is a consumer review website in which users can express their opinions about items by assigning numerical ratings and writing article reviews. The extended Epinions data set generated by [15] describes the trust and distrust relationships among users and their ratings on other users' articles. Flixster.com is a movie sharing and discovering website where users can feed back their movie ratings in the range from 0.5 to 5.0 with step 0.5.

In particular, due to limited processing power and memory, we sample a portion of the whole datasets. In order to test the influence of different information on the result, we sample users who have rated at least 10 items and issued at least 1 trust statement. We randomly sample 8051 users from Epinions dataset and 5000 users from Flixster dataset as well as the user ratings and trust statements, respectively. Note that different from the Epinions dataset, the trust information in Flixster is symmetric and no distrust information is available. The statistics of datasets are shown in Table 1, where Epinions has higher rating sparsity (having a larger number of users but issuing a small number of item ratings) and relatively lower trust sparsity (having more trust relationships between users).

### 4.2. Methodology and metrics

In experiments, we randomly pick 80% of the review data as the training set and the rest as the test set. We apply 5-fold cross validation to evaluate the performance of each method, that is to say, we repeat this procedure five times until all folds are tested and average the results. The proposed approach is implemented in MATLAB. All the experiments are conducted on a Windows PC with 4-core 2.3 GHz processors and 16 GB memory.

[1] Accordingly, the complexities of the three variants of MCRec (MCRec-UI, MCRec-T and MCRec-single) which will be discussed later in Section 4.3 are $O(|W|*|R|)+O(|R^w|*|d^2|+|R^w|*|d|)$, $O(|T|*|K|)+O(|R|*|d^2|+|R|*|d|)$, and $O(|W|*|R|)+O(|R^w|*|d^2|+|R^w|*|d|)$, respectively.

[2] www.trustlet.org

[3] www.video.flixster.com/

**Table 1**
Statistics of the datasets.

| | Epinions | Flixster |
|---|---|---|
| # of user | 8051 | 5000 |
| # of item | 22,130 | 23,963 |
| # of rating | 686,306 | 1,037,869 |
| # of trust | 30,217 | 7869 |
| # of distrust | 33,561 | 0 |
| Trust sparsity | 99.48% | 99.97% |
| Rating sparsity | 99.62% | 99.13% |
| Average rating | 4.7649 | 3.6074 |

Since the objective of our approach is to improve the rating prediction accuracy, we use two standard metrics to measure the accuracy of various models: MAE (Mean Absolute Error) and RMSE (Root Mean Square Error).

$$RMSE=\sqrt{\frac{1}{N}\sum_{r=1}^{n}(R-\hat{R})^2}$$

$$MAE=\frac{1}{N}\sum_{r=1}^{N}|R-\hat{R}| \tag{10}$$

where $N$ denotes the number of tested ratings. $R$ is the real rating of an item and $\hat{R}$ is the corresponding predicted rating. In general, smaller RMSE and MAE values indicate better accuracy. The former (MAE) considers every error of equivalent value, while the latter (RMSE) emphasizes larger errors.

### 4.3. Comparisons

In this section, we will answer the second question: *How can this combination further improve the accuracy of recommender systems*. We compare our method with some most related recommendation algorithms. In addition, three variants of the proposed method are given out as comparative methods to investigate the effectiveness of our multi-type clustering based recommendation methods (*MCRec*). The comparative methods are as follows:

- *BaseMF* [2] is the basic matrix factorization model, which is also the basic model of our method. We use it as the baseline in the experiments.
- *RWT/D* [33] is a trust/distrust based recommendation method which is based on the matrix factorization model. The authors propose two regularization terms to incorporate the user trust and distrust relationships into the recommender systems respectively. They do not consider any types of clustering information. In order to provide a fair comparison, we ignore the confidence of trust/distrust in their models and average the results derived from these two models as the final result.
- *CBMF* [31] is the most related work, which partitions users and items into different co-clusters by utilizing the PLSA model and employ the matrix factorization model as their basic model. But the difference lies in tacking the trust relationships. They regard users' trust relationships as an influence factor of acquiring the co-clusters, while we cluster the trust relationships to group the trusted users into the same trust clusters.
- *MCRec-single* considers similarity-based user clustering and item clustering on the assumption that each user and item belongs to a unique cluster.[4] It is a kind of hard division, and non-overlapping exists. We perform the matrix factorization algorithm in each cluster.

[4] Each user or item is assigned to a single cluster in which she/he/it has the highest probability.

- *MCRec-UI* considers similarity-based user clustering and item clustering. In this method, trust clustering is ignored. We just obtain the co-clusters and perform the matrix factorization algorithm in each co-cluster.
- *MCRec-T* only considers the clustering of social relationships. We incorporate the clustering-based trust regularization term into the matrix factorization model and train the model in the whole data space.
- *MCRec* stands for the method proposed in the present work which considers multi-type clustering, i.e. trust-based user clustering, similarity-based user and item clustering.

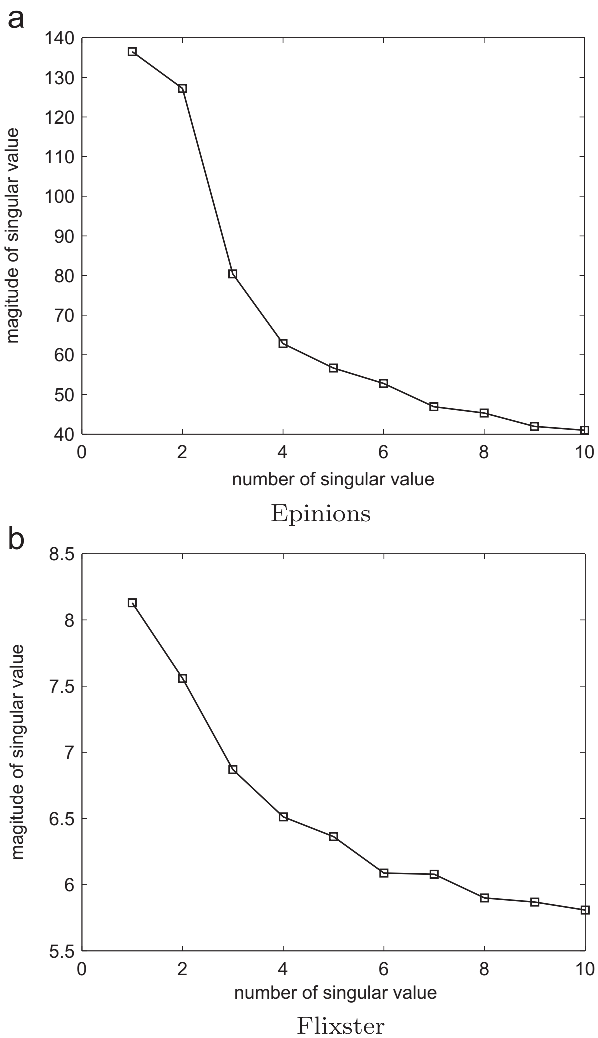



**Fig. 5.** A line plot of the singular values of the trust matrix on Epinions (a) and Flixster (b) datasets, where the *x*-axis represents the number of singular value and the *y*-axis represents the magnitude of a singular value.

### 4.4. The number of trust-based user clusters

Fig. 5 is a line graph which describes the singular values that obtained by the trust-based user clustering on Epinions and Flixster datasets, respectively. According to the method of [29], the graph drops sharply at the number of 2 and 1 as can be seen from Fig. 5(a) and (b). Therefore, it is reasonable to set these two numbers as the best $K$ for Epinions and Flixster datasets. Note that there are only undirected trust relationships in Flixster dataset, thus the trust relationship matrix $T$ is degraded to an adjacency matrix under this condition. In addition, the best $K$ is 1 for Flixster dataset, which means that we do not have to cluster the trust network. One explanation for this phenomenon is the sparsity and symmetry of the trust relationships in Flixster dataset.

### 4.5. Comparison with other approaches

#### 4.5.1. Performance on Epinions

Table 2 shows the experimental results on Epinions dataset with two different evaluation metrics: RMSE and MAE. We set the dimension of matrix factorization $d=10$, the regularization term $\lambda=1$ and the social regularization parameter $\alpha=1$. We vary the number of co-cluster $W=5, 10, 15, 20$ to study the influence of co-clusters. Since BaseMF, RWT/D and MCRec-T have not considered the clustering of users and items, they produce the same results in all cases as shown in Table 2.

From Table 2 we can conclude that our method MCRec yields the best performance under all of the evaluation conditions and gets the best results when $W=10$, which is consistent with the reality. Intuitively, users may be interested in limited topic groups and items may belong to a few categories. If $W$ is too small, i.e. around 5, or too large, i.e. around 20, the results get worse. In our consideration, small number of co-clusters cannot clearly partition different topic groups. When $W$ becomes too large, ratings in each subgroups become sparse, which may largely influence the predicted results. Next we will discuss the comparative results when $W=10$.

Compared with all the other methods, BaseMF gets the worst results mainly because it only utilizes the ratings, thus cannot handle the data sparsity problem efficiently. RWT/D considers the trust and distrust relationships in the models and alleviates the data sparsity problem to some extent. However, we find that as the number of distrust relationships increases, the performance of distrust aspect decreases accordingly, which lower the performance of RWT/D. CBMF considers the clustering of users and items and exploits users' social activeness and dynamic interest to weight the importance of user ratings. A better result is achieved compared with RWT/D. Note that we do not consider the dynamic influence in our work, therefore we neglect this factor in the experiment. On Epinions dataset, MCRec yields 31.15%, 29.44%, 9.41% relative improvement compared to BaseMF, RWT/D, CBMF in terms of RMSE.

**Table 2**
Performance comparisons on Epinions dataset in terms of RMSE and MAE.

| Methods | $W=5$ | | $W=10$ | | $W=15$ | | $W=20$ | |
|---|---|---|---|---|---|---|---|---|
| | RMSE | MAE | RMSE | MAE | RMSE | MAE | RMSE | MAE |
| BaseMF | 0.5817 | 0.3673 | 0.5817 | 0.3673 | 0.5817 | 0.3673 | 0.5817 | 0.3673 |
| RWT/D | 0.5676 | 0.3493 | 0.5676 | 0.3493 | 0.5676 | 0.3493 | 0.5676 | 0.3493 |
| CBMF | 0.4811 | 0.3123 | 0.4421 | 0.2940 | 0.4713 | 0.3109 | 0.5124 | 0.3396 |
| MCRec-single | 0.5637 | 0.3487 | 0.5397 | 0.3277 | 0.5612 | 0.3481 | 0.5823 | 0.3700 |
| MCRec-UI | 0.4821 | 0.3134 | 0.4454 | 0.3079 | 0.4779 | 0.3385 | 0.5061 | 0.3431 |
| MCRec-T | 0.5411 | 0.3591 | 0.5411 | 0.3591 | 0.5411 | 0.3591 | 0.5411 | 0.3591 |
| MCRec | **0.4094** | **0.2562** | **0.4005** | **0.2531** | **0.4297** | **0.2701** | **0.4630** | **0.3164** |

**Table 3**
Performance comparisons on Flixster dataset in terms of RMSE and MAE.

| Methods | $W=5$ | | $W=10$ | | $W=15$ | | $W=20$ | |
|---|---|---|---|---|---|---|---|---|
| | RMSE | MAE | RMSE | MAE | RMSE | MAE | RMSE | MAE |
| BaseMF | 1.1053 | 0.8689 | 1.1053 | 0.8689 | 1.1053 | 0.8689 | 1.1053 | 0.8689 |
| RWT/D | 1.0917 | 0.8439 | 1.0917 | 0.8439 | 1.0917 | 0.8439 | 1.0917 | 0.8439 |
| CBMF | 0.9255 | 0.7279 | 0.9216 | 0.7199 | 0.9298 | 0.7354 | 0.9281 | 0.7289 |
| MCRec-single | 1.0821 | 0.8537 | 1.0716 | 0.8265 | 1.0823 | 0.8518 | 1.0830 | 0.8521 |
| MCRec-UI | 0.9250 | 0.7255 | 0.9238 | 0.7244 | 0.9272 | 0.7310 | 0.9316 | 0.7315 |
| MCRec-T | 1.0917 | 0.8439 | 1.0917 | 0.8439 | 1.0917 | 0.8439 | 1.0917 | 0.8439 |
| MCRec | **0.9107** | **0.7136** | **0.9028** | **0.7101** | **0.9134** | **0.7238** | **0.9257** | **0.7311** |

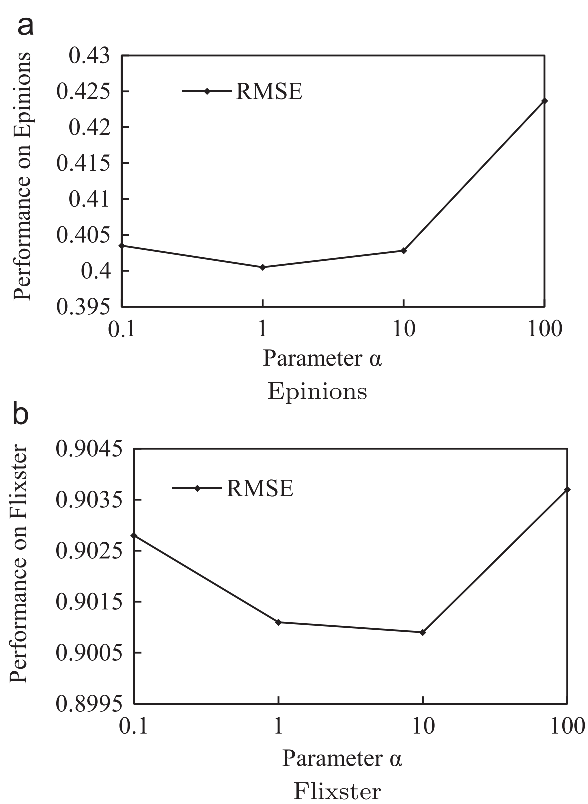



**Fig. 6.** Impact of trust regularization parameter $\alpha$ on Epinions (a) and Flixster (b) datasets.

In order to demonstrate the effectiveness of our proposed method and investigate the contribution of different types of clustering information, we also compare MCRec with its variants MCRec-single, MCRec-UI and MCRec-T. MCRec-UI achieves significantly better performance than MCRec-single, which indicates that overlapping clusters pose great impact on the performance of clustering-based recommender systems. Compared with MCRec-UI and MCRec-T, MCRec decreases RMSE by 10.08% and 25.98%, MAE by 17.80%, 29.52%. All these numbers prove that if the clustering-based recommendation model considers different types of clustering information (the clustering of users, items and trust relationships), it outperforms the version that only considers one of them.

### 4.5.2. Performance on Flixster

Table 3 shows the experimental results on Flixster dataset. We also set the dimension of matrix factorization $d=10$, the regularization term $\lambda=1$ and the social regularization parameter $\alpha=10$. We vary the number of co-cluster $W=5, 10, 15, 20$ to study the influence of co-clusters.

Comparative results show that our method MCRec achieves the best results under all conditions on Flixster dataset. Similar to Epinions, the best results are achieved when $W=10$. Compared with BaseMF, RWT/D, CBMF, MCRec-single, MCRec-UI and MCRec-T, MCRec decreases RMSE by 18.32%, 17.30%, 2.27%, 17.634%, 2.27% and 17.30%, respectively. All the numbers prove that MCRec provides reasonably accurate recommendations that are much better than baselines.

It is important to mention that there are two big differences between the comparative results on Epinions and Flixster datasets. First of all, there are no distrust relationships in Flixster dataset, hence the RWT/D method becomes RWT which ignores the incorporation of distrust aspect. In addition, when $K$ is set to be 1 in Section 4.4, we do not cluster the trust relationships, thus MCRec-T is degenerated into the RWT method. That's the reason why they get the same results on Flixster dataset as shown in Table 3. Secondly, different from the results on Epinions dataset, MCRec has limited improvement compared with its variant MCRec-UI. In our consideration, this is because the available trust relationships in Flixster dataset is quite sparse compared with the Epinions dataset, thus neighborhood obtained by trust clustering may be not accurate for future prediction. Moreover there are friendships only in Flixster dataset, and the trust relationship matrix is degraded into an adjacency matrix when performing SVD signs method, which also influences the final result.

From Tables 2 and 3 we can observe that both the RMSE and MAE on Epinions dataset are smaller than those on Flixster dataset. One explanation for this phenomenon is that ratings in Epinions dataset are quite bias which can be seen from Table 1. The average rating of the sampled Epinions dataset and Flixster dataset is 4.7649 and 3.6074 respectively. There are only 4.59% ratings less than 4 in Epinions dataset, which may influence the prediction results to a large extent.

### 4.6. Effect of parameter $\alpha$

The regularization parameter $\alpha$ is very important in the MCRec model. It controls how much our method should incorporate the information of trust clusters. Fig. 6 shows how the trust regularization parameter $\alpha$ impacts the performance of MCRec, where $\alpha$ is varied as 0.1, 1, 10, 100 respectively. Intuitively, when $\alpha$ is small, the model MCRec behaves like MCRec-UI which does not consider any trust relationships. When $\alpha$ increases, the clustering-based trust regularization term becomes more influential on the model and the contribution of trust clustering information increases. MCRec gets the best results when $\alpha$ is around 1 and 10 on Epinions and Flixster datasets respectively as we can see from Fig. 6. Nevertheless, if $\alpha$ becomes increasingly large, the trust smoothness term would dominate the learning processes, and RMSE increases at the same time. Note that as the trust relationships are quite sparse in Flixster dataset, the influence of parameter $\alpha$ is not as significant as that in Epinions dataset.

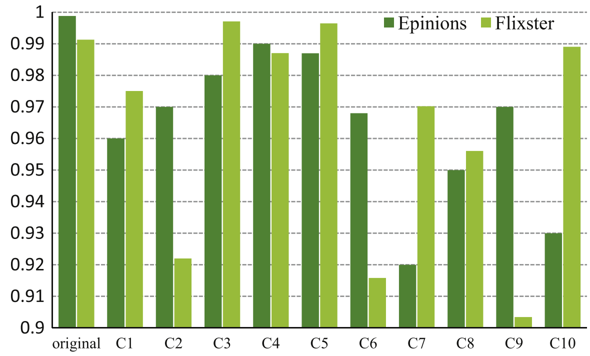



**Fig. 7.** The comparison of density from original matrix to submatrices in Epinions and Flixster.

**Table 4**
RMSE values on cold start users ($W=10$).

| Method | Epinions | Flixster |
|---|---|---|
| BaseMF | 0.7168 | 1.2576 |
| RWT/D | 0.6782 | 1.2149 |
| CBMF | 0.6671 | 1.2036 |
| MCRec-single | 0.6911 | 1.2375 |
| MCRec-UI | 0.6798 | 1.1687 |
| MCRec-T | 0.6618 | 1.1484 |
| MCRec | **0.5731** | **0.9873** |

### 4.7. Data sparsity problem

The sparsity problem occurs when available data are insufficient and it is a major issue that limits the quality of recommendations. In this paper, we employ the clustering technique to allocate users and items into proper co-clusters. The density of all submatrices in co-clusters compared to the original one is recorded in Fig. 7. We observe that almost all the submatrices are denser than the original one(especially for the Epinions dataset where the original rating sparsity is 99.63%), which demonstrates that our method can mitigate the sparsity problem effectively. On the other hand, trust relationships are utilized to regularize users' feature vectors in our model which can alleviate the sparsity problem to some extent.

### 4.8. Performance on cold start users

Cold start is a more specific problem than the sparse data problem, because when a new user has no or too few ratings, it is impossible to provide accurate recommendations. We consider users who have rated less than 5 items and issued at least 1 trust statement as cold start users.[5] In both Epinions and Flixster more than 50% users are cold start users. Therefore, the efficiency of the proposed recommendation algorithm for cold start users becomes very important. In the experiment, $W$ is set to be 10 when MCRec gets the best performance on both Epinions and Flixster datasets. For other comparative methods, we also search for the best configurations while applying to the real datasets. Table 4 shows the comparative results for cold start users on both datasets. From this table we can conclude that our proposed MCRec outperforms other baseline methods for cold start users. The improvement of the RMSE for cold start users compared to MCRec-UI is 15.70% for Epinions and 15.52% for Flixster. The gain for cold start users is more than the gain for all users, which we discussed in Section 4.5. This implies that MCRec is efficient in the cold start situation.

[5] We filter out users who have no ratings or trust statements.

### 4.9. Result analysis

In terms of accuracy of predictions, from both Tables 2 and 3, we can conclude that our MCRec method is very effective compared with other baseline methods. On the one hand, the co-clusters generated by the similarity-based user clustering and item clustering can alleviate the data sparsity problem to a large extent. On the other hand, the clustering of trust and distrust relationships helps to find trust neighbors in different trust communities and provides a rich source of information for the prediction of user preferences, especially for some seriously sparse datasets like Epinions.

## 5. Conclusion and future work

In this paper, we have made progress towards making clustering-based recommender systems more general and practical. We proposed a framework based on matrix factorization which considered multi-type clustering, i.e. trust-based user clustering, similarity-based user clustering and similarity-based item clustering. We detailed about how to obtain different type of clusters and presented a method to combine them into a unified model. Real datasets based experiments demonstrated the efficiency of our MCRec in rating predictions, especially in addressing the data sparsity and cold start problems.

One potential limitation of the proposed framework is that we only consider the situations involving ratings and trust, although it may be straightforward to incorporate other information sources, such as time-series information, dynamic variations information and the place information, to further improve the quality of recommender systems. Another limitation is we made an assumption that users are more similar in the same trust clusters than users in different trust clusters. For future work, we will investigate whether such an assumption is valid in all settings and evaluate the similarity between users from different trust clusters. Moreover, we will also evaluate the performance of the proposed method in item ranking and some real applications, such as personalized e-commerce, e-learning and e-services systems [34,35] which strongly suffer from data sparsity and cold start problems. We hope that our results will positively contribute to the development of recommender systems.

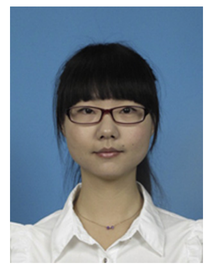
## Acknowledgments

This research is funded by the Fundamental Research Funds for the Central Universities, HUST:2013TS101. Appreciation also goes to anonymous reviewers for their careful work and thoughtful suggestions that have helped improve this paper substantially.

## References

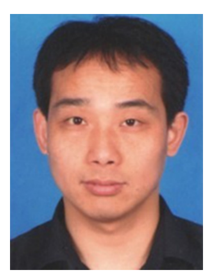
[1] G. Adomavicius, A. Tuzhilin, Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions, IEEE Trans. Knowl. Data Eng. 17 (6) (2005) 734–749.
[2] Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, Computer 42 (8) (2009) 30–37.
[3] T. Hofmann, Latent semantic models for collaborative filtering, ACM Trans. Inf. Syst. 22 (1) (2004) 89–115.
[4] X. Yang, Y. Guo, Y. Liu, Bayesian-inference-based recommendation in online social networks, IEEE Trans. Parallel Distrib. Syst. 24 (4) (2013) 642–651.
[5] L.H. Ungar, D.P. Foster, Clustering methods for collaborative filtering, In: Proceedings of the AAAI Workshop on Recommendation Systems, vol. 1, 1998.
[6] G. Guo, J. Zhang, N. Yorke-Smith, Leveraging multiviews of trust and similarity to enhance clustering-based recommender systems, Knowl. Based Syst. 74 (2015) 14–27.

[7] B.M. Sarwar, G. Karypis, J. Konstan, J. Riedl, Recommender systems for large-scale e-commerce: scalable neighborhood formation using clustering, In: Proceedings of the 5th International Conference on Computer and Information Technology, vol. 1, Citeseer, 2002.
[8] M. OConnor, J. Herlocker, Clustering items for collaborative filtering, In: Proceedings of the ACM SIGIR Workshop on Recommender Systems, vol. 128, Citeseer, 1999.
[9] M.C. Pham, Y. Cao, R. Klamma, M. Jarke, A clustering approach for collaborative filtering recommendation using social network analysis, J. UCS 17 (4) (2011) 583–604.
[10] X. Ma, H. Lu, Z. Gan, Improving recommendation accuracy by combining trust communities and collaborative filtering, In: Proceedings of the 23rd ACM International Conference on Information and Knowledge Management, ACM, Shanghai, China, 2014, pp. 1951–1954.
[11] B. Xu, J. Bu, C. Chen, D. Cai, An exploration of improving collaborative recommender systems via user–item subgroups, In: Proceedings of the 21st International Conference on World Wide Web, ACM, Lyon, France, 2012, pp. 21–30.
[12] A.L.V. Pereira, E.R. Hruschka, Simultaneous co-clustering and learning to address the cold start problem in recommender systems, Knowl. Based Syst. 82 (2015) 11–19.
[13] T. George, S. Merugu, A scalable collaborative filtering framework based on co-clustering, In: Proceedings of the 5th IEEE International Conference on Data Mining, IEEE, Texas, USA, 2005, pp. 625–628.
[14] H. Li, D. Wu, W. Tang, N. Mamoulis, Overlapping community regularization for rating prediction in social recommender systems, In: Proceedings of the 9th ACM Conference on Recommender Systems, ACM, Vienna, Austria, 2015, pp. 27–34.
[15] P. Massa, P. Avesani, Trust-aware recommender systems, In: Proceedings of the 1st ACM Conference on Recommender Systems, ACM, Minneapolis, Minnesota, USA, 2007, pp. 17–24.
[16] M. Jamali, M. Ester, Trustwalker: a random walk model for combining trust-based and item-based recommendation, In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, Paris, France, 2009, pp. 397–406.
[17] H. Ma, I. King, M.R. Lyu, Learning to recommend with social trust ensemble, In: Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, Boston, Massachusetts, USA, 2009, pp. 203–210.
[18] M. Jamali, M. Ester, A matrix factorization technique with trust propagation for recommendation in social networks, In: Proceedings of the 4th ACM Conference on Recommender Systems, ACM, Barcelona, Spain, 2010, pp. 135–142.
[19] M. Jiang, P. Cui, F. Wang, W. Zhu, S. Yang, Scalable recommendation with social contextual information, IEEE Trans. Knowl. Data Eng. 26 (11) (2014) 2789–2802.
[20] M. Jiang, P. Cui, R. Liu, Q. Yang, F. Wang, W. Zhu, S. Yang, Social contextual recommendation, In: Proceedings of the 21st ACM International Conference on Information and Knowledge Management, ACM, Maui, HI, USA, 2012, pp. 45–54.
[21] R. Forsati, M. Mahdavi, M. Shamsfard, M. Sarwat, Matrix factorization with explicit trust and distrust side information for improved social recommendation, ACM Trans. Inf. Syst. 32 (4) (2014) 17.
[22] R. Forsati, I. Barjasteh, F. Masrour, A.-H. Esfahanian, H. Radha, Pushtrust: An efficient recommendation algorithm by leveraging trust and distrust relations, In: Proceedings of the 9th ACM Conference on Recommender Systems, ACM, Vienna, Australia, 2015, pp. 51–58.
[23] G.-R. Xue, C. Lin, Q. Yang, W. Xi, H.-J. Zeng, Y. Yu, 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, ACM, Salvador, Brazil, 2005, pp. 114–121.
[24] K. Ji, R. Sun, X. Li, W. Shu, Improving matrix approximation for recommendation via a clustering-based reconstructive method, Neurocomputing 173 (2016) 912–920.
[25] T. DuBois, J. Golbeck, J. Kleint, A. Srinivasan, Improving recommendation accuracy by clustering social networks with trust, Recomm. Syst. Soc. Web 532 (2009) 1–8.
[26] A. Bellogin, J. Parapar, Using graph partitioning techniques for neighbour selection in user-based collaborative filtering, In: Proceedings of the 6th ACM Conference on Recommender Systems, ACM, Dublin, Ireland, 2012, pp. 213–216.
[27] K.-Y. Chiang, J. J. Whang, I. S. Dhillon, Scalable clustering of signed networks using balance normalized cut, In: Proceeding of the 21th ACM International Conference on Information and Knowledge Management, ACM, Maui, HI, USA, 2012, pp. 615–624.
[28] B. Yang, W.K. Cheung, J. Liu, Community mining from signed social networks, IEEE Trans. Knowl. Data Eng. 19 (10) (2007) 1333–1348.
[29] E.P. Douglas, Clustering datasets with singular value decomposition (Ph.D. thesis), College of Charleston, 2009.
[30] T. Hofmann, Probabilistic latent semantic indexing, In: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, Berkeley, USA, 1999, pp. 50–57.
[31] B. Yin, Y. Yang, W. Liu, Exploring social activeness and dynamic interest in community-based recommender system, In: Proceedings of the 23rd International Conference on World Wide Web Companion, International World Wide Web Conferences Steering Committee, ACM, Seoul, Korea, 2014, pp. 771–776.
[32] H. Ma, D. Zhou, C. Liu, M.R. Lyu, I. King, Recommender systems with social regularization, In: Proceedings of the 4th ACM International Conference on Web Search and Data Mining, ACM, Hong Kong, China, 2011, pp. 287–296.
[33] H. Ma, M.R. Lyu, I. King, Learning to recommend with trust and distrust relationships, In: Proceedings of the Third ACM Conference on Recommender Systems, ACM, New York, USA, 2009, pp. 189–196.
[34] Q. Shambour, J. Lu, A hybrid trust-enhanced collaborative filtering recommendation approach for personalized government-to-business e-services, Int. J. Intell. Syst. 26 (9) (2011) 814–843.
[35] Z. Zhang, H. Lin, K. Liu, D. Wu, G. Zhang, J. Lu, A hybrid fuzzy-based personalized recommender system for telecom products/services, Inf. Sci. 235 (2013) 117–129.



**Xiao Ma** is a doctoral candidate at the School of Computer Science and Technology, Huazhong University of Science and Technology, HUST. Her research interests include recommender systems and social network analysis.

**Hongwei Lu** received the B.Sc., M.Sc. and Ph.D. degrees from HUST, Wuhan, China. Currently, he is a professor at the School of Computer Science and Technology, HUST. His research interests are in security and privacy in ubiquitous computing and electronic commerce, with a focus on security protocol analysis, access control, and trust negotiation.



**Zaobin Gan** received the Ph.D. degree from HUST, Wuhan, China. He worked as a visiting researcher at Macquarie University from July 2004 to June 2006. Currently, he is an associate professor at the School of Computer Science and Technology, HUST. His research interests are in security and privacy in electronic commerce, with a focus on trust computing.

**Qian Zhao** received the M.Sc. degree from HUST, Wuhan, China. Her research interests include social network analysis and data mining.