Journal of Universal Computer Science, vol. 17, no. 4 (2011), 583-604
submitted: 30/10/10, accepted: 15/2/11, appeared: 28/2/11 © *J.UCS*

A Clustering Approach for Collaborative Filtering Recommendation Using Social Network Analysis

Manh Cuong Pham, **Yiwei Cao**, **Ralf Klamma**, **Matthias Jarke**
(Information Systems & Database Technology
RWTH Aachen University, Aachen
Ahornstr. 55, D-52056, Aachen, Germany
{pham, cao, klamma, jarke}@dbis.rwth-aachen.de)

Abstract: Collaborative Filtering(CF) is a well-known technique in recommender systems. CF exploits relationships between users and recommends items to the active user according to the ratings of his/her neighbors. CF suffers from the data sparsity problem, where users only rate a small set of items. That makes the computation of similarity between users imprecise and consequently reduces the accuracy of CF algorithms. In this article, we propose a clustering approach based on the social information of users to derive the recommendations. We study the application of this approach in two application scenarios: academic venue recommendation based on collaboration information and trust-based recommendation. Using the data from DBLP digital library and Epinion, the evaluation shows that our clustering technique based CF performs better than traditional CF algorithms.

Key Words: clustering, collaborative filtering, trust, social network analysis
Category: H.3.3, H.3.7

1 Introduction

Recommender System (RS) is a class of applications dealing with information overload. As more and more information is published on the World Wide Web, it is difficult to find needed information quickly and efficiently. RS helps solve this problem by recommending items to users based on their previous preferences. Many applications have used recommender systems, especially in the e-commerce domains.

Typically in a recommender system, there is a set of users and a set of items. Each user u rates a set of items by some values. The task of a recommender system is to predict the rating of user u on an un-rated item i or recommend some items for user u based on the existing ratings. Techniques in RS [Adomavicius and Tuzhilin, 2005] can be divided into three categories: collaborative filtering, content-based recommendation and hybrid approaches. Collaborative filtering make recommendations based on the ratings of item i by the set of users whose rating profiles are most similar to that of user u. Content-based methods use the features of items, e.g. movie's genres, directors, actors, etc., to generate recommendations. Hybrid approaches [Burke, 2002] make recommendations by combining collaborative filtering and content-based recommendation.

Collaborative Filtering (CF) [Breese et al., 1998, Su and Khoshgoftaar, 2009] is a widely used technique in recommender systems. Methods in CF can be either memory-based and model-based. Memory-based algorithms operate on the entire user-item rating matrix and generates recommendations by identifying the neighborhood of the target user to whom the recommendations will be made, based on the agreement of user's past ratings. Model-based techniques use the rating data to train a model and then the model will be used to derive the recommendations. Well-known model-based techniques include clustering [Ungar and Foster, 1998, Sarwar et al., 2002], pLSA [Hofmann, 2004], matrix factorization (e.g. SVD) [Koren et al., 2009], machine learning on the graph [Wang et al., 2006, Zhou et al., 2008], etc. Memory-based techniques are quite successful in real-world applications because they are easy to understand, easy to implement and work well in many real-world situations. However, there are some problems which limit the application of memory-based techniques, especially in the large-scale applications like Amazon.com. The most serious problem is the sparsity of user-item rating matrix where each user only rates a small set of a large database of items. The similarity between users (or items) is often derived from few overlapping ratings and it is hence a noisy and unreliable value. Another problem of memory-based CF is efficiency. It has to compute the similarity between every pair of users (or items) to determine their neighborhoods. This is not computationally feasible for the ad-hoc recommender systems with millions of users and items.

To overcome the weaknesses of memory-based techniques, a line of research has focussed on model-based clustering techniques with the aim of seeking more accurate, yet more efficient methods. Based on ratings, these techniques group users or items into clusters, thus give a new way to identify the neighborhood. In this article, we are concerned with clustering techniques using social information. Our objective is to identify the communities of similar users based on their social relationships and use these communities as a mechanism to make the recommendations. Currently, some recommender systems allow users to build their social networks and use these network as an additional information to suggest items to users. For example, in Epinion[1] users can state how much they consider every other user trustworthy, meaning that how much they consider the ratings provided by a certain user as valuable and relevant. Epinion then uses the trust network to re-order the items list presented to users. Other kinds of relationships are also useful for recommendation in different domains, for example the scientific collaboration between scholars and citations between publications can be used to recommend research papers, venues (journals, conferences) to researchers. The idea here is not to search for similar users as CF does but to search for users who may "known" each other to form a so-called community.

[1] `http://www10.epinions.com/`

Then the opinion of the community is used to recommend items to the active user.

This article is organized as follows. In Section 2, we make a brief survey of the collaborative filtering techniques. In section 3, some related work is presented. In Section 4, we present our clustering approach. In Section 5 and Section 6, we investigate the approach in two recommendation problems: academic venue recommendation and trust-based recommendation. The article finishes with conclusions and our directions for future work.

2 Memory-based Collaborative Filtering

Memory-based CF (user-based or item-based) is based on the fact that users often like the items which are preferred by others users who have agreed with them in the past. Memory-based CF uses the entire user-item rating database to generate recommendations. A typical CF algorithm proceeds in three steps:

1. Calculating the similarity, $w_{i,j}$, between active user/item i and user/item j.
2. Neighborhood formation: select k similar users/items.
3. Generating top N items by weighted average of all the ratings of users/items in the neighborhood.

2.1 Similarity Computation

Similarity computation is a crucial step of CF algorithms. The basic idea of similarity computation is co-rating. For user-based CF, similarity between user i and user j is computed using the items which have been rated by both users. For item-based CF, similarity between item i and item j is computed by working on the users who have rated both of these items. Among many methods to compute similarity, Pearson correlation and vector Cosine are most popular and widely used. For user-based CF, Pearson correlation between user u and user v is computed as follows:

$$w_{u,v} = \frac{\sum_{i \in I}(r_{u,i} - \overline{r}_u)(r_{v,i} - \overline{r}_v)}{\sqrt{\sum_{i \in I}(r_{u,i} - \overline{r}_u)^2}\sqrt{\sum_{i \in I}(r_{v,i} - \overline{r}_v)^2}} \tag{1}$$

where I is the set of items rated by both user i and user j, $r_{u,i}$ is the rating of user u on item i and $\overline{r}_i$ is the average rating of user i.

For item-based CF, Pearson correlation between user i and user j is computed as follows:

$$w_{i,j} = \frac{\sum_{u \in U} (r_{u,i} - \overline{r}_i)(r_{u,j} - \overline{r}_j)}{\sqrt{\sum_{u \in U} (r_{u,i} - \overline{r}_i)^2} \sqrt{\sum_{u \in U} (r_{u,j} - \overline{r}_j)^2}} \tag{2}$$

where U is the set of users who have rated both item i and item j, $r_{u,i}$ is the rating of user u on item i and $\overline{r}_i$ is the average rating of item i.

Vector Cosine similarity is computed by treating each user as a vector of ratings of the user on items and measuring the cosine of the angle formed by these vectors. Formally, if R is the $m \times n$ user-item rating matrix, then the similarity between two users u and v is defined as the cosine of the m dimensional vectors corresponding to the u^{th} and v^{th} row of the matrix R.

$$w_{u,v} = cos(u, v) = \frac{\overrightarrow{u} \bullet \overrightarrow{v}}{\parallel \overrightarrow{i} \parallel * \parallel \overrightarrow{i} \parallel} \tag{3}$$

where $\bullet$ denotes the dot-product of the two vectors. The cosine similarity between two items is computed analogously where u and v correspond to the u^{th} and v^{th} column of the matrix R.

2.2 Neighborhood Selection

After the similarity computation, CF algorithms have to select the most similar users for the active user. This is the important step since the recommendations are generated using the ratings of neighbors and therefore neighborhood has an impact on the recommendation quality. According to [Zhang and Pu, 2007], there are five strategies for neighbors selection:

1. Baseline strategy: select the top k nearest-neighbors who have rated the given item.

2. Baseline strategy with overlap threshold: select the top k nearest-neighbors who have rated the given item and who have rated at least φ items as the active user has rated (overlapped with the active user).

3. Similarity strategy: select the top nearest-neighbors purely according to their similarities with the active user.

4. Combination strategy: a combination of the strategy 1 and strategy 3.

5. Combination strategy with overlap threshold: a combination of strategies 1, 2 and 3.

A neighborhood selection strategy is chosen depending on the similarity measures and the application domains. In the actual situations, users may use different rating scales, which Cosine similarity cannot take into account. However, using Pearson we also have problems, where some of the neighbors might have only few common rating with the active user. But they could have high similarity value with the active user by chance. The reason is that Pearson correlation does not take into account the degree of overlaps between users. If we use, for example, the first strategy then it could exclude many high similar neighbors just because they have not rated the given items. The combination of these strategies is preferable, but it could decrease the performance of the algorithms.

2.3 Generating Recommendation

In the user-based algorithms, when a subset of nearest neighbors (neighborhood) of the active user are chosen, predictions are generated based on a weighted aggregate of their ratings. Most used aggregating functions are *weighted sum* and *simple weighted average*. To make the prediction for the active user a on an item i, weighted sum is computed using all the ratings of the neighbors on that item by the following formula:

$$P_{a,i} = \overline{r}_a + \frac{\sum\limits_{u \in U}(r_{u,i} - \overline{r}_u) w_{a,u}}{\sum\limits_{u \in U} \parallel w_{a,u} \parallel} \tag{4}$$

Using simple weighted average, the prediction can also be computed by the following formula:

$$P_{a,i} = \frac{\sum\limits_{u \in U} r_{u,i} w_{a,u}}{\sum\limits_{u \in U} \parallel w_{a,u} \parallel} \tag{5}$$

Where $\overline{r}_a$ and $\overline{r}_u$ are the average ratings of user a and user u, $w_{a,u}$ is the weight between user a and user u. The summations are over all the users $u \in U$ who have rated item i. The prediction is generated analogously in item-based algorithms where the summations are over all the ratings of user a on the items in the neighborhood of item i and the weight is the similarity between item i and the items in its neighborhood.

3 Related Work

Clustering methods for CF have been extensively studied by several studies. Ungar [Ungar and Foster, 1998] proposed a repeated K-means and Gibb sampling clustering techniques that group users into clusters with similar items and group items into clusters which tend to be liked by the same users. Kohrs [Kohrs and Merialdo, 1999] used a hierarchical clustering algorithm to independently cluster users and items into two cluster hierarchies. The recommendation is made by the weighted sum of the defined centers of all nodes in the cluster hierarchies on the path from the roots to the particular leaves. Sarwar et al. [Sarwar et al., 2002] evaluated a clustering approach which groups users into clusters based on their similarities. They showed that clustering provides comparable recommendation quality as traditional CF, while significantly improving the online performance. There are also other studies on clustering techniques for collaborative filtering [George and Merugu, 2005, Xue et al., 2005, Truong et al., 2007, Nathanson et al., 2007, Rashid et al., 2006, Zhang and Hurley, 2009].

That reasearch work clusters users and items using the rating data while ignoring the additional information, for example the relationship between users (e.g. trust and friendship) and the relationship between items (e.g. citation between publications). Additional information has been proved to be very useful in certain application domains [Massa and Avesani, 2007, Zhou et al., 2008, O'Donovan and Smyth, 2005, Zhu et al., 2007]. Here we propose to exploit the social relationships for recommendation: users are clustered based on the social network built from various relationships. When we have user clusters, traditional CF algorithms can operate on the clusters instead of the whole user-item matrix. By reducing the dimensions of user-item rating matrix and therefore avoiding the data sparsity problem, this approach can provide better recommendation result in terms of accuracy and can improve the online performance of CF algorithms.

Our study is related to social group recommendation [Backstrom et al., 2006, Anglade et al., 2007, Harper et al., 2007]. Kleinberg [Backstrom et al., 2006] considered the social group formation and community membership in large social networks and their use in recommender systems. Anglade [Anglade et al., 2007] proposed a complex network based approach for music recommendation and shared radio channels in P2P networks. They applied a hub-based clustering technique on the network of peers and showed that the resulting clusters identify the communities of peers that share similar music preferences. These clusters then can be used to provide music recommendations to peers in the groups. Maxwell Harper et al. [Harper et al., 2007] described an algorithm for clustering users of an online community and automatically describing the resulting user groups. They developed an activity-balanced clustering algorithm that considered both user activity and user interests in forming clusters. Users are automatically assigned to groups and have access to group-based social recommendations.

4 Network Clustering for CF

The application of clustering techniques reduces the sparsity and improves the scalability of the systems since the similarity can be calculated only for users in the same clusters. Different clustering strategies can be performed based on users and items, as illustrated in Fig. 1. In general, clustering users (or items) results in creating sub-matrices of the entire user-item rating matrix. Then classical CF algorithms (user-based and item-based) can be used to generate recommendation based on these sub-matrices. In Fig. 1(a), when user-based CF is applied on user clusters, the neighbors of the active user are users in the same user cluster. If item-based CF is used, the ranking of an item is based on the items which are rated by users in the same user cluster. Similarly, in Fig. 1(b), the neighbors of the active user are users who have rated items in the same item cluster (in case of user-based CF) and the ranking of an item is based on the items in the same item cluster (in case of item-based CF).

Fig. 1 depicts the situation where each user and each item are assigned uniquely to one cluster, though users can be assigned to different user clusters (in item clustering) and items can belong to different item clusters (in user clustering). In this case, the prediction for the active user can be made by averaging the opinions of the others in the same user cluster (user-based) and the ranking of an item is based on the items in the same item cluster (item-based). However, in a real-world application, users and items can belong to several clusters, e.g. one user may be interested in the movies of different genres such as horror, war, comedy or drama. One movie could also be assigned to different categories according to genre. The prediction is then made by an average across the clusters.

One simple clustering technique is to classify items based on their content, e.g. movies are categorized by genre. Then the prediction for an item is made by an average of the opinions of users in the clusters that this item belongs to. However, this technique is not feasible in the domains where the features of the items are hidden or hard to extract, as well as where the structure of the categories is complicated, for example when items are hierarchically classified. Consequently, several approaches have been proposed ([Quan et al., 2006, Zhang and Hurley, 2009]) to cluster items based on user rating.

4.1 Algorithm

We argue that social relationships have an impact on user behavior in recommender systems and propose to use clustering technique on social network of users to identify their neighborhood. We present the first case as depicted in Fig. 1(a), where user-based CF is combined with user-based clustering. It differs from the traditional model-based CF clustering in two folds. On the one hand, it exploits the social relationship of users, while others techniques cluster users based on the ratings. On the other hand, the clusters are extracted from the network topology, which are quite different to the explicit communities studied in [Harper et al., 2007] and [Backstrom et al., 2006].

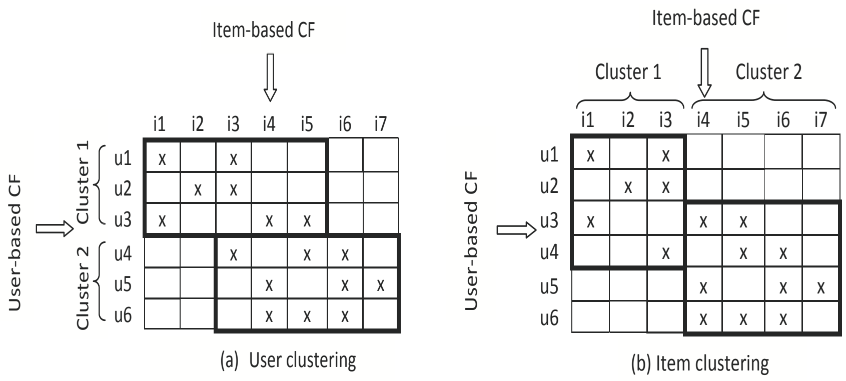

Figure 1: Clustering for collaborative filtering

Traditional CF algorithms proceed in two phases. At the first phase, calculate the similarities between pairs of users and identify their neighborhood. Then recommendations are generated for active user based on the aggregate of the ratings of the neighbors. In our approach, first we cluster users (offline). Then we apply traditional CF process within clusters to generate recommendations. The algorithm is as follows:

1. Formulate the social network of users, $G = (U, E)$, where U is the set of users, $U = \{u_1, u_2, ..., u_n\}$ and E is the set of social relations between users. G might be a weighted or un-weighted network.

2. Perform a clustering algorithm on the network G. The set of users U is divided into clusters $U_1, U_2, ..., U_q$, where $V_i \cap U_j = \emptyset$ and $U = U_1 \cup U_2 ... \cup U_q$.

3. Using clusters as the neighborhoods, the prediction rating for the active user u on item i is computed by either weighted sum or simple weighted average:

$$P_{a,i} = \overline{r}_a + \frac{\sum\limits_{u \in U_a} (r_{u,i} - \overline{r}_u) w_{a,u}}{\sum\limits_{u \in U_a} \parallel w_{a,u} \parallel} \tag{6}$$

$$P_{a,i} = \frac{\sum\limits_{u \in U_a} r_{u,i} w_{a,u}}{\sum\limits_{u \in U_a} \parallel w_{a,u} \parallel} \tag{7}$$

where $P_{a,i}$ is the prediction rating of user a for item i, $\overline{r}_a$ is the average rating of user a, U_a is the cluster of user a and $w(a, u)$ is the weight between user a

and user u. The weight $w(a,u)$ could be the similarity between active user a and the user u in the same cluster, computed using the ratings on the items rated by users in that cluster. That results in a pure memory-based CF with clustering. The weight $w(a,u)$ could also be computed based on the social information, for example the (directed/indirected) trust value that user a gives to user u, resulting in a social recommendation with clustering. When $w(a,u)$ is set to 1, we have group recommendation where every user in a cluster gets the same recommendation for a given item.

4.2 Clustering Techniques

Clustering is the critical and important step in our approach because the clusters gives us the neighborhood of the active user. All the computations following clustering step (weight computation, recommendation generation) depends on the clusters. Methods for clustering networks can be divided into two research categories: graph partitioning and blocking modeling (or hierarchical clustering). We concentrate on the hierarchical clustering since it does not require any assumptions about the cluster structure of a network. The idea is to successively build (agglomerative), or break up (divisive) a hierarchy of clusters (called a dendrogram) with the leaves being the initial network nodes, the root representing the whole graph and the inner nodes corresponding to the clusters at different steps of the algorithm. A hierarchical clustering algorithm can stop at a certain step when the resulting partition is optimal according to some measures. One common measure is called *modularity* [Newman and Girvan, 2004]. The modularity of a given partition of a network measures the quality of that partition. Formally, the modularity Q is defined as

$$Q = \Sigma_{i=1}^{q}(e_{ii} - a_i^2) \tag{8}$$

with

$$a_i = \Sigma_{j=1}^{q} e_{ji} \tag{9}$$

where e_{ji} is the fraction of edges between nodes from group j and i, and q is the number of clusters. The fraction of edges connecting nodes in group i internally is hence e_{ii} and a_i denotes the overall fraction of links connecting to nodes in i. a_i^2 then corresponds to the expected fraction of internal edges given a random assignment of nodes into communities. Thus, if a particular clustering gives no more within-community edges than expected by random chance, Q will be equal to 0 (because then $e_{ii} \approx a_i^2$). Values other than 0 indicate deviations from randomness. Empirical observations indicate that values greater than 0.3 correspond to significant community structures.

In our study, we use the algorithm proposed by Clauset [Clauset et al., 2004], an improved version of the method proposed by Newman [Newman, 2004]. Here, a greedy implementation of hierarchical agglomerative clustering is used: two communities are joined if this join results in the greatest increase (or smallest decrease) of the modularity Q - compared to all other possible joins between pairs of communities. When the computation of the whole dendrogram is finished, the partition with maximal modularity is chosen. Clauset improves this approach by slightly modifying the join-condition so that a computation of the whole dendrogram can be avoided. The algorithm has a worst case time complexity of $O(MdlogN)$ (with d being the depth of the dendrogram), but a complexity of $O(Nlog2N)$ for sparse graphs. Since most complex networks are indeed sparse graphs, the algorithm will run in almost linear time. The algorithm is widely accepted in research community with many studies making use of it.

5 Collaborative Filtering and Venues Recommendation

In this section, we present our first experiment on academic venue recommendation. Our interest is to support researchers, especially young PhD students, to find the right venues or the right communities. Academic venues (journals, conferences etc.) play an important role in computer science. In recent years, the number of venues has increased dramatically, as shown in data from DBWorld[2], DBLP[3] and EventSeer.net[4]. Researchers have to deal with information overload when they want to find the suitable venues to submit papers to.

In computer science, venues are organized into series, e.g ACM International Conference on Knowledge Discovery and Data Mining. Researchers publish their work in annual events or in special issues. Our goal is to recommend the upcoming events (or special issues) in which researchers might be interested. Users are researchers and items are venues. The ratings of users can be explicitly expressed, or can be implicitly inferred from users behavior. Here we approximately infer the attention of researchers to the venues in which they participated by the number of papers they published. This measure is used as the ratings of researchers on venues and is computed as following:

$$R(r, v) = \frac{p(r, v)}{\sum_{i}^{n} p(r, i)} \tag{10}$$

where $p(r, v)$ is the number of papers researcher r published in venue v and n is the number of venues. The rating $R(r, v)$ thus is the fraction of papers which researcher r published in venue v. We note that this measure might be a too strong indicator for researchers' opinion, since publishing a paper in a

[2] `http://www.cs.wisc.edu/dbworld/`
[3] `http://www.informatik.uni-trier.de/ley/db/`
[4] `http://eventseer.net/`

venue or participating in a conference depends on many different aspects. One might favour a particular conference or journal. But because of some problems, they have not published any papers in it. On the one hand, this measure underestimates, and therefore, gives us the low bound of researchers opinion. However, it is an accurate measure since publishing papers in a conference or journal presents the topics of interest and the opinion of researchers on that one.

Using a social network, researchers are clustered to identify their neighborhood. Social relationships between researchers are reflected via research activities such as publishing papers (co-authoring), referencing to other work (citing) and participating in venues. That results in different types of social network, which can be used to group similar researchers: co-authorship network, citation network and venue co-participation. In this article, we cluster researchers based on co-authorship network. We leave the investigation on the quality of the neighborhood identified by other types of network to the future work.

Using clusters as neighborhoods, a traditional CF algorithm is applied to generate recommendations. Venues are ranked according to the ratings of cluster's members and the ranked list is returned to the active researcher.

5.1 Co-authorship Network Clustering

We performed two test cases based on DBLP dataset, each of them uses a snapshot of co-authorship network at a certain year and predicts the venues in which a researcher will participate in the next year. For example, using clusters from the co-authorship network in 2005, we recommend the venues that a target researcher might take part in 2006. DBLP data was downloaded in July, 2009. It contains 788,259 authors, 1,226,412 publications and 3,490 venues. We extract two snapshots of co-authorship network in the years of 2005 and 2006. The snapshots are created based on the publications from the considered year backwards, e.g. 2005 snapshot takes the co-authorship network of publications published in 2005 or earlier. Each snapshot is presented as an weighted un-directed graph, where nodes are researchers and there is an edge between two researchers if they co-authored at least one publication. The edges are weighted by the number of co-authored publications. The 2005 network contains 478,108 nodes and 1,427,196 edges. The 2006 network has 544,601 nodes and 1,686,876 edges.

We run the density-based clustering algorithm [Clauset et al., 2004] on each network. Cluster size distribution is given in Fig. 2. Overall, the algorithm gave us several large clusters with the size of thousand nodes and large number of clusters with the size ranging from 2 to hundred nodes. The modularity Q for the partitions of 2005's and 2006's networks are 0.829 and 0.82, respectively.

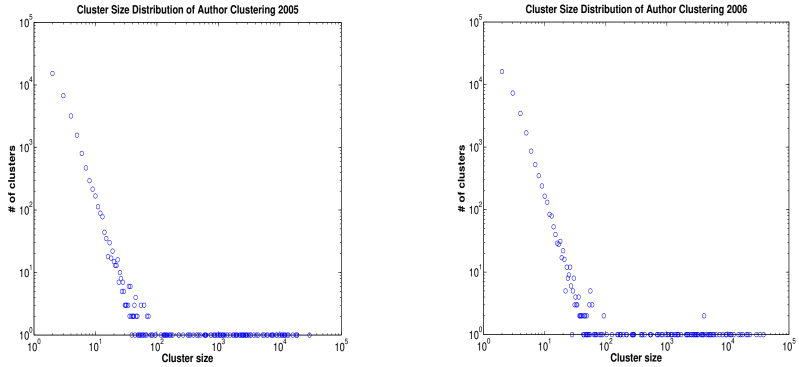

Figure 2: Cluster size distribution

5.2 Evaluation Metrics

We use two standard measures from information retrieval: *precision* and *recall*, defined as follows:

$$Precision = \frac{Relevant\_venues\_recommended}{Venues\_recommended} \tag{11}$$

$$Recall = \frac{Relevant\_venues\_recommended}{Relevant\_venues} \tag{12}$$

where $Relevant\_venues\_recommended$ is the number of venues in the recommended list of venues, in which a researcher participates in the next year; $Venues\_recommended$ is the number of venues recommended; $Relevant\_venues$ is the number of venues a researcher takes part in the next year.

We compare our clustering approach with the traditional Cosine CF algorithm which follows the *top-k* recommendation principle. To make the evaluation fair, the number of cluster members for each researcher was recorded and we force *CF* to use the same number of neighbors for recommendation generation. Similarities between researchers are computed using cosine measure. For each test case, we randomly select one thousand researchers as active users and generate recommendations for them. For each researchers, the precision values are computed at 11 standard recall levels, $0\%, 10\%, ..., 100\%$. Then the average precision value of one thousand researchers at each recall level is computed. Finally, the precision-recall curve of each algorithm is plotted.

The precision-recall curves for 2005 and 2006 test cases are given in Fig. 3. Clearly, clustering method performs better than traditional Cosine CF. This contradicts with the result in [Sarwar et al., 2002], where the prediction quality is

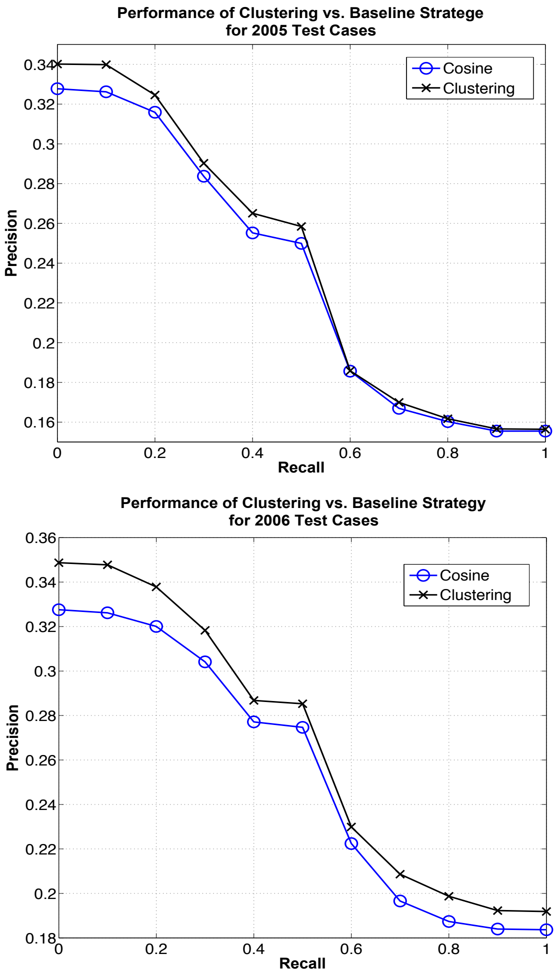

Figure 3: Performance of clustering for test cases 2005 and 2006

worse in case of the clustering algorithm. The reason is that previous clustering approaches group users using rating data and often result in less personal and worse accuracy than classical CF algorithms. Here we cluster users (researchers) based on their social relationships (the co-authorship network), so using additional information for clustering has the benefit. However, (as being observed from the chart) the precision of both algorithms is quite low (about 33% for Cosine CF and 35% for clustering method) and the difference in accuracy is also small. That because we evaluate the approach based on the venues participation of researchers. A researcher may prefer a particular conference, but it may take time to turn it into the action: publishing a paper in this conference. An online evaluation therefore is necessary.

There is also a number of reasons to explain the above results. User clusters have a great impact on the recommendation in our approach. Whether the clusters reflect the true information need of users depends on the network (or the relationship) we use for clustering and also depends on clustering algorithms. Here we examine only the co-authorship network which describes the strongest relation between authors. As mentioned earlier, other types of social relationship (e.g. citation network) could be used to identify better user clusters, which would lead to better result. Also, here we use a clustering approach which assigns users uniquely to one cluster. It would not be the case in the real-world situation where one user might participate in several clusters (communities). For example, if one author is working on different field such as Data Mining, Database and HCI, then he might participate in the communities of these fields. Discovery overlapping communities and integrating them into the recommendation process is challenging and we are still working on this problem.

6 Trust-based Clustering

In the second evaluation, we consider a typical recommendation method using social trust. With the advent of online social networks, the trust-based approach to recommendation has emerged. This approach assumes a trust network among users and makes recommendations based on the rating of the users that are directly or indirectly trusted by the active user [Jamali and Ester, 2009]. Work has been done on trust-based recommendation including trust inference on trust networks [Ziegler, 2005, Golbeck, 2005, O'Donovan and Smyth, 2005], learning to recommend with trust [Ma et al., 2009a, Ma et al., 2009b], etc. In this section, we present our experiment with clustering on trust network.

6.1 Dataset

We use Epinion as the data source for our experiment. Epinion[5] is a well-known knowledge sharing site and review site, which was established in 1999. In Epinion, users can assign products or reviews integer ratings from 1 to 5. Users can also maintain their "trust" lists which present the networks of trust relationships between users. Trust list of a user are people who the user considers their opinion is relevant or valuable. Epinion uses this trust network to order the product reviews such that a user first sees the reviews by users that they trust.

The dataset used in our experiment is collected by [Massa and Avesani, 2007]. It consists of 49,290 users, 139,738 items and 664,824 reviews. The total ratings

[5] http://www10.epinions.com/

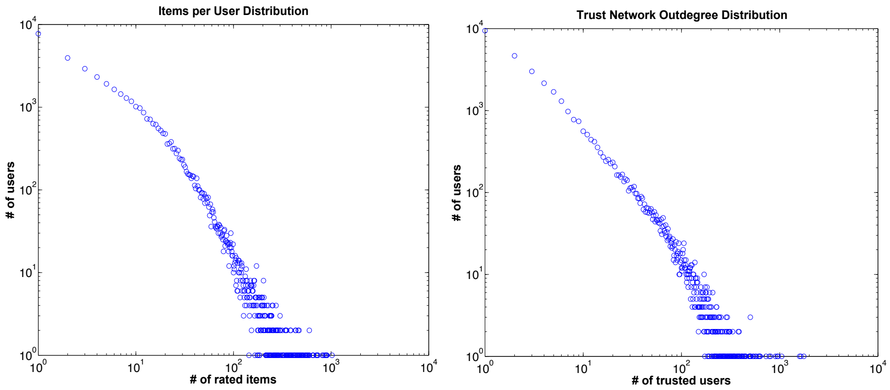

Figure 4: Power-law distributions of Epinion data set

is 664,823. There are 487,182 trust statements in total. The user-item matrix sparsity is 99.99%. The statistics of the dataset and trust network are given in Table 1 and Table 2. We also observe a number of Power-law distributions in the data set, including items per user and outdegree distribution in the trust network (Fig. 4)

Table 1: Dataset statistics

Statistics	User	Item
Min. Num. of Ratings	1	1
Max. Num. of Ratings	1,023	2,026
Avg. Num. of Ratings	16.55	4.76

Table 2: Trust network statistics

Statistics	Trust per User	Be Trusted per User
Max. Num.	1,760	2,589
Avg. Num.	14.35	9.89

6.2 Evaluation Metrics

The widely used technique for evaluating recommender systems is *leave-one-out*. This technique involves hiding one rating from the test set and trying to to predict it with a certain algorithm. Then the predicted rating is compared with the real rating and the difference in absolute value is the prediction error [Herlocker et al., 2004]. We use *leave-one-out* technique and employ the Mean Absolute Error (MAE) to measure the quality of the prediction of our approach in comparison with other collaborative methods and trust-based recommendation methods. Formally, MAE is defined as follows:

$$R(r, v) = \frac{\sum_{i,j} \mid p_{i,j} - r_{i,j} \mid}{n} \qquad (13)$$

where n is the number of ratings over all the test cases, $p_{i,j}$ is the prediction rating of user i on item j and $r_{i,j}$ is the actual rating. We also use the coverage to measure the percentage of the time that an algorithm is able to successfully make a recommendation. Normally, this means that the algorithm was able to make some recommendations. A successful recommendation is defined as the prediction that the algorithm can make for a rating in the test set.

6.3 A Comparison of Various Algorithms

We compare out algorithm with two other baseline methods, traditional user-based Pearson collaborative filtering and directed trust recommendation.

- Pearson: the algorithm is presented in section 2 in which we use Pearson correlation as the similarity measure. For the neighborhood selection, we use the strategy 5 presented in Section 2.2 where we require that a candidate for a neighbor of the active user is the one who rated the given item and have at least four items (overlap threshold equals to 4) in common. So if an user has no such neighbors, we consider that Pearson algorithm is not able to make the recommendation for this user.
- Directed trust: to make the prediction for the active user on an item, the ratings of users in the trust list of the active user are aggregated as in formula 6, where $w_{a,u}$ is the trust value that user a has given to user u. Because in Epinion, trust value is binary (trust or not), $w_{a,u}$ equals to 1.
- Clustering with trust: we run the density-based clustering algorithm by Clauset [Clauset et al., 2004] on trust network to group users into clusters. The trust network is treated as an un-directed network. The prediction for the active user is computed according to the formula 6 based on the ratings

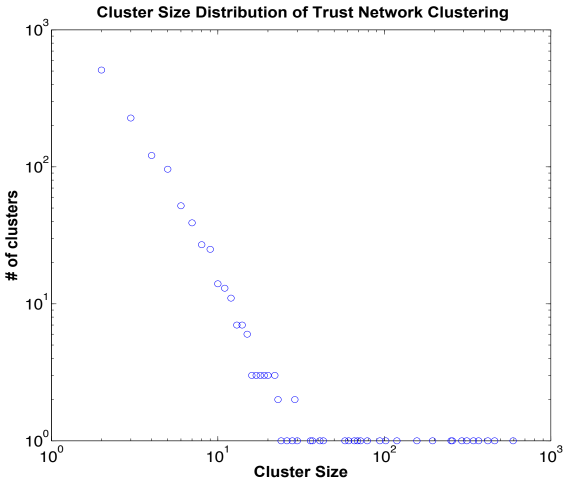

Figure 5: Cluster size distribution of Epinion trust network

of users in the same cluster. The weight $w_{a,u}$ is set to one, resulting in less personalized recommendation (group recommendation). One can employ a trust inference algorithm, for example TidalTrust [Golbeck, 2005], to compute the trust value between user a and user u, and use it as the weight $w_{a,u}$ to have more personalized recommendation.

We divide the dataset into training and test data sets at a 90% to 10% (5,000 ratings) ratio. The clustering is performed on the whole trust network and the predictions are computed for the ratings in the test set. Clustering algorim gives us 1,210 clusters in which there are two large clusters with size over ten thousands users and many clusters with small size (see Fig. 5). To make the evaluation fair, we force Pearson to use the same number of neighbors as the number of cluster members. We perform 5-fold cross validations for the experiment, where in each fold we randomly assign the ratings into either training or test data sets.

The prediction accuracy evaluated by MAE is given in Fig. 6. We can observe that our trust-based clustering approach performs better than traditional Pearson collaborative filtering and directed trust recommendation (0.1373 and 0.0383 respectively). The improvement is even more significant in term of coverage as presented in Fig. 7. Directed trust method performs worst regarding to coverage since the neighborhood of the active user includes only the trust-peers, so if the given item is not rated by any trust-peers, it cannot be recommended. Trust-based clustering outperforms Pearson method (by 24% coverage). As mentioned in Section 2, Pearson method might be not able to recommend the given item to the active user if non of the neighbors have rated that item. Due to the sparsity

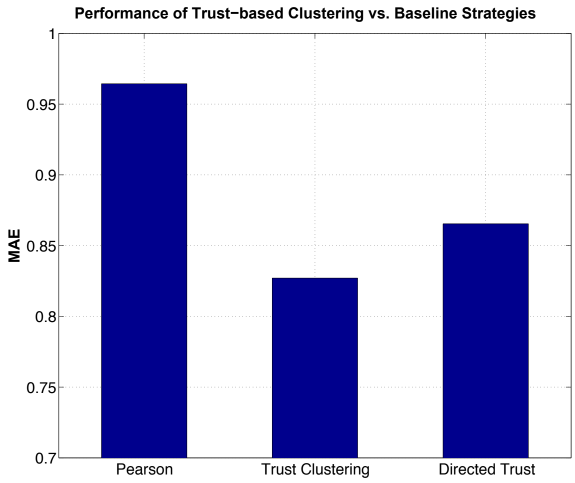

Figure 6: Comparison of MAE for trust-based clustering and baseline methods

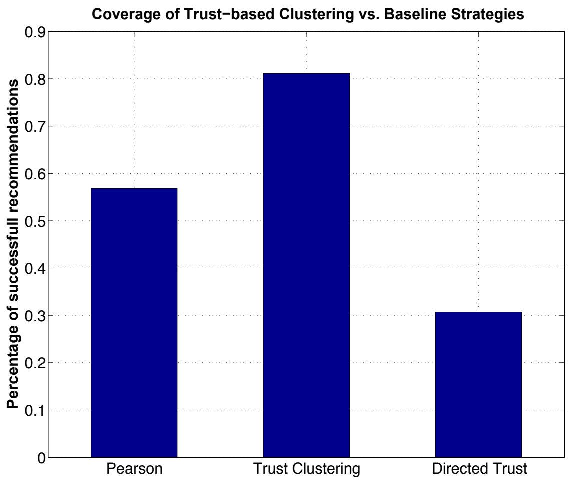

Figure 7: Comparison of coverage for trust-based clustering and baseline methods

of the rating data, the problem becomes serious in which Pearson method cannot make the recommendations for many users. However, with clustering the active user still gets the recommendation as long as he is assigned to a trust community. In summary, trust-based clustering outperforms both baseline methods in term of prediction accuracy and coverage.

7 Conclusions and Future Work

In this article, we have presented a clustering approach to collaborative filtering recommendation technique. Instead of using ratings data, we propose to use social relationship between users to identify their neighborhoods. A complex network clustering technique is applied on the social network of users to find the groups of similar users. After that, the traditional CF algorithms can be used to efficiently generate the recommendations. We presented our experiments with two real-world data sets and show that our clustering method outperforms traditional collaborative filtering algorithms.

The paper draws several issues which need to be further studied. First, the evaluation on different types of social network is necessary. For example, to recommend academic venue to researchers, the citation network and venue co-participation network are promising social networks which can be used to group similar researchers. We address these problems in our development system called AERCS (*http://bosch.informatik.rwth-aachen.de:5080/AERCS/*) which provides useful recommendation tools for researchers in computer science. It is also interesting to see the performance of the approach in other domains like music recommendation on the online music sites such as Last.fm. We also would like to perform the evaluation on item-based clustering: items are clustered using the relation between items and then CF is applied on the clusters of items instead of user clusters. Another extension of the algorithm is to assign users or items to several clusters (overlapping clusters) and use the opinion of these clusters to generate recommendations. Overlapping clusters could depict the real-world situations where users and items participate in different communities. That would further improve our algorithm.

Second, the comparison of our method with the more recently developed methods such as trust inference, pLSA, matrix factorization, PCA, clustering using rating data, etc., is also needed. Clustering can be considered as a dimensional deduction method which can be used to reduce the sparsity of the rating data. pLSA, matrix factorization and PCA are some well-established methods in this line of research. The main difference here is that we use the additional information for clustering, while other methods are based on rating data. Comparison with the above approaches would reveal the benefit of using social information in recommender systems.

Acknowledgments

This work has been supported by the Graduiertenkolleg (GK) "Software for mobile communication systems", the UMIC Research Centre, RWTH Aachen University and the EU FP7 IP ROLE. We would like to thank our colleagues for the fruitful discussions.

References

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

[Anglade et al., 2007] Anglade, A., Tiemann, M., and Vignoli, F. (2007). Complex-network theoretic clustering for identifying groups of similar listeners in p2p systems. In *RecSys '07: Proceedings of the 2007 ACM conference on Recommender systems*, pages 41–48, New York, NY, USA. ACM.

[Backstrom et al., 2006] Backstrom, L., Huttenlocher, D., Kleinberg, J., and Lan, X. (2006). Group formation in large social networks: membership, growth, and evolution. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 44–54, New York, NY, USA. ACM.

[Breese et al., 1998] Breese, J. S., Heckerman, D., and Kadie, C. (1998). Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the14th Annual Conference on Uncertainty in Artificial Intelligence (UAI98)*, pages 43–52.

[Burke, 2002] Burke, R. (2002). Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370.

[Clauset et al., 2004] Clauset, A., Newman, M. E. J., and Moore, C. (2004). Finding community structure in very large networks. *Phys. Rev. E*, 70(6):066111.

[George and Merugu, 2005] George, T. and Merugu, S. (2005). A scalable collaborative filtering framework based on co-clustering. In *ICDM '05: Proceedings of the Fifth IEEE International Conference on Data Mining*, pages 625–628, Washington, DC, USA. IEEE Computer Society.

[Golbeck, 2005] Golbeck, J. A. (2005). *Computing and applying trust in web-based social networks*. PhD thesis, College Park, MD, USA. Chair-Hendler, James.

[Harper et al., 2007] Harper, F. M., Sen, S., and Frankowski, D. (2007). Supporting social recommendations with activity-balanced clustering. In *RecSys '07: Proceedings of the 2007 ACM conference on Recommender systems*, pages 165–168, New York, NY, USA. ACM.

[Herlocker et al., 2004] Herlocker, J. L., Konstan, J. A., Terveen, L. G., and Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1):5–53.

[Hofmann, 2004] Hofmann, T. (2004). Latent semantic models for collaborative filtering. *ACM Trans. Inf. Syst.*, 22(1):89–115.

[Jamali and Ester, 2009] Jamali, M. and Ester, M. (2009). Trustwalker: a random walk model for combining trust-based and item-based recommendation. In *KDD '09: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 397–406, New York, NY, USA. ACM.

[Kohrs and Merialdo, 1999] Kohrs, A. and Merialdo, B. (1999). Clustering for collaborative filtering applications. In *In Proceedings of CIMCA'99*. IOS Press.

[Koren et al., 2009] Koren, Y., Bell, R., and Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37.

[Ma et al., 2009a] Ma, H., King, I., and Lyu, M. R. (2009a). Learning to recommend with social trust ensemble. In *SIGIR '09: Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 203–210, New York, NY, USA. ACM.

[Ma et al., 2009b] Ma, H., Lyu, M. R., and King, I. (2009b). Learning to recommend with trust and distrust relationships. In *RecSys '09: Proceedings of the third ACM conference on Recommender systems*, pages 189–196, New York, NY, USA. ACM.

[Massa and Avesani, 2007] Massa, P. and Avesani, P. (2007). Trust-aware recommender systems. In *RecSys '07: Proceedings of the 2007 ACM conference on Recommender systems*, pages 17–24, New York, NY, USA. ACM.

[Nathanson et al., 2007] Nathanson, T., Bitton, E., and Goldberg, K. (2007). Eigentaste 5.0: constant-time adaptability in a recommender system using item clustering. In *RecSys '07: Proceedings of the 2007 ACM conference on Recommender systems*, pages 149–152, New York, NY, USA. ACM.

[Newman, 2004] Newman, M. E. J. (2004). Fast algorithm for detecting community structure in networks. *Phys. Rev. E*, 69(6):066133.

[Newman and Girvan, 2004] Newman, M. E. J. and Girvan, M. (2004). Finding and evaluating community structure in networks. *Phys. Rev. E*, 69(2):026113.

[O'Donovan and Smyth, 2005] O'Donovan, J. and Smyth, B. (2005). Trust in recommender systems. In *IUI '05: Proceedings of the 10th international conference on intelligent user interfaces*, pages 167–174, New York, NY, USA. ACM.

[Quan et al., 2006] Quan, T. K., Fuyuki, I., and Shinichi, H. (2006). Improving accuracy of recommender system by clustering items based on stability of user similarity. In *CIMCA '06: Proceedings of the International Conference on Computational Inteligence for Modelling Control and Automation and International Conference on Intelligent Agents Web Technologies and International Commerce*, page 61, Washington, DC, USA. IEEE Computer Society.

[Rashid et al., 2006] Rashid, A. M., Shyong, Karypis, G., and Riedl, J. (2006). Clustknn: A highly scalable hybrid model- & memory-based cf algorithm. In *WEBKDD 2006*, Philadelphia, Pennsylvania, USA.

[Sarwar et al., 2002] Sarwar, B., Karypis, G., Konstan, J., and Riedl, J. (2002). Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering. In *Proceedings of the Fifth International Conference on Computer and Information Technology*.

[Su and Khoshgoftaar, 2009] Su, X. and Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. *Adv. in Artif. Intell.*, 2009:1–20.

[Truong et al., 2007] Truong, K., Ishikawa, F., and Honiden, S. (2007). Improving accuracy of recommender system by item clustering. *IEICE - Trans. Inf. Syst.*, E90-D(9):1363–1373.

[Ungar and Foster, 1998] Ungar, L. and Foster, D. (1998). Clustering methods for collaborative filtering. In *Proceedings of the Workshop on Recommendation Systems*. AAAI Press, Menlo Park California.

[Wang et al., 2006] Wang, F., Ma, S., Yang, L., and Li, T. (2006). Recommendation on item graphs. In *ICDM '06: Proceedings of the Sixth International Conference on Data Mining*, pages 1119–1123, Washington, DC, USA. IEEE Computer Society.

[Xue et al., 2005] Xue, G.-R., Lin, C., Yang, Q., Xi, W., Zeng, H.-J., Yu, Y., and Chen, Z. (2005). Scalable collaborative filtering using cluster-based smoothing. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 114–121, New York, NY, USA. ACM.

[Zhang and Pu, 2007] Zhang, J. and Pu, P. (2007). A recursive prediction algorithm for collaborative filtering recommender systems. In *RecSys '07: Proceedings of the 2007 ACM conference on Recommender systems*, pages 57–64, New York, NY, USA. ACM.

[Zhang and Hurley, 2009] Zhang, M. and Hurley, N. (2009). Novel item recommendation by user profile partitioning. In *WI-IAT '09: Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, pages 508–515, Washington, DC, USA. IEEE Computer Society.

[Zhou et al., 2008] Zhou, D., Zhu, S., Yu, K., Song, X., Tseng, B. L., Zha, H., and Giles, C. L. (2008). Learning multiple graphs for document recommendations. In *WWW '08: Proceeding of the 17th international conference on World Wide Web*, pages 141–150, New York, NY, USA. ACM.

[Zhu et al., 2007] Zhu, S., Yu, K., Chi, Y., and Gong, Y. (2007). Combining content and link for classification using matrix factorization. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 487–494, New York, NY, USA. ACM.

[Ziegler, 2005] Ziegler, C.-N. (2005). *Towards Decentralized Recommender Systems.* PhD thesis, Freiburg, Germany.