Clustering Approach for Hybrid Recommender System

Qing Li
Dept. of Computer Engineering
Kumoh National Institute of Technology
Kumi, 730-701, South Korea
liqing@se.kumoh.ac.kr

Byeong Man Kim
Dept. of Computer Engineering
Kumoh National Institute of Technology
Kumi, 730-701, South Korea
bmkim@se.kumoh.ac.kr

Abstract

Recommender system is a kind of web intelligence techniques to make a daily information filtering for people. In this work[1]*, Clustering techniques have been applied to the item-based collaborative filtering framework to solve the cold start problem. It also suggests a way to integrate the content information into the collaborative filtering. Extensive experiments have been conducted on MovieLens data to analyze the characteristics of our technique. The results show that our approach contributes to the improvement of prediction quality of the item-based collaborative filtering, especially for the cold start problem.*

1. Introduction

In our daily life, we make our choices at most cases relying on recommendations from other people either by word of mouth, recommendation letters, movie and book reviews printed in newspapers, or general surveys. Recent years we have seen the explosive growth of the sheer volume of information. The number of movies, books, music, news, and in particular on-line information is staggering. Absolutely, we need information recommender system to help us prioritize information so that we can reduce the searching time and spend much more time in reading the information that we need or favor.

At the initial state, many recommender systems were fairly simple query-based information retrieval system, which can be called as content-based recommender system. For example, search engines recommend web pages with content similar to user queries [15]. Later, Goldberg and his colleagues firstly applied the collaborative filtering technology to recommender systems [6], [19]. It uses free annotations or explicit *like it* or *hate it* annotations to identify like-minded users manually. GroupLens [14] and Ringo [18] developed independently, were the first to automate prediction. Collaborative recommender systems collect ratings of items from many individuals and make recommendations based on those ratings to a given user. The techniques of collaborative filtering have been developed quickly not only in the research area but also in the commercial field. Such as MovieLens system recommends movies, Jeter system recommends jokes [7], Flycasting recommends online radio [9], and GAB recommends web pages based on the bookmarks [21]. A growing number of companies, including Amazon.com, CDNow.com and Levis.com, employ or provide recommender system solutions [17].

Although collaborative filtering has been very successful in both research and practice, it can not recommend new items to users without any history in the system and completely denies any information that can be extracted from contents of items, such as cast list, movie genre and synopsis of movie etc. Further more the quality of recommendation is completely based on the user rating, instead of the information content.

For this reason, hybrid recommender systems have been provided, which can exploit both user preferences and contents. Proposed approaches to hybrid system, which combines content-based and collaborative filters together, can be categorized into two groups.

One group is the linear combination of results of collaborative and content-based filtering as Figure 1 shows, such as systems that are described by Claypool [3] and Wasfi [20]. ProfBuilder [20] recommends web pages using both content-based and collaborative filters, and each creates a recommendation list without combining them to make a combined prediction. Claypool describes a hybrid approach for an online newspaper domain, combining the two predictions using an adaptive weighted average: as the number of users accessing an item increases, the weight of the collaborative component tends to increase. But how to decide the weights of collaborative and content-based components is unclearly given by the author.

[1]This work was supported by Korea Research Foundation Grant (KRF-2002-041-D00459).

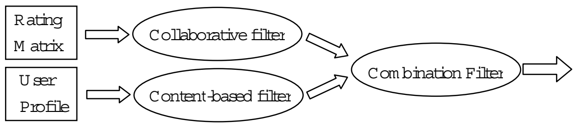

Figure 1. Linear combination

The other group is the sequential combination of content-based filtering and collaborative filtering as Figure 2 shows. In this system, firstly, content-based filtering algorithm is applied to find users, who share similar interests. Secondly, collaborative algorithm is applied to make predictions, such as RAAP [4] and Fab filtering systems [2]. RAAP is a content-based collaborative information filtering for helping the user to classify domain specific information found in the WWW, and also recommends these URLs to other users with similar interests. To decide the similar interests of users, scalable Pearson correlation algorithm based on the web page category is used. Fab system uses content-based techniques instead of user ratings to create profiles of users. So the quality of predictions is fully depended on the content-based techniques, inaccurate profiles result in inaccurate correlations with other users and thus make poor predictions.

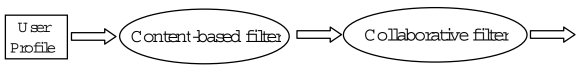

Figure 2. Sequential combination

As we can see, linear combination model just recommends based on the two different recommender system - one is content-based recommender system and the other is collaborative recommender system. As for the sequential combination model, although it considers the useful information from user profiles, it denies the important information from user ratings.

In this paper, we suggest a technique that introduces the contents of items into the item-based collaborative filtering to improve its prediction quality and solve the cold start problem. Shortly, we call the technique ICHM (Item-based Clustering Hybrid Method). Comparing to the sequential combination model, which ignores the useful rating data from users and makes predictions only on the content information, the ICHM can integrate the item information and user ratings to calculate the item-item similarity.

The rest of the paper is organized as follows. The next section provides the detail description of our approach - ICHM, and the section 3 describes our experimental work. The final section provides some conclusion remarks.

2 Our approach

The Figure 3 is an overview of our approach, and the detail procedure is described as follows:

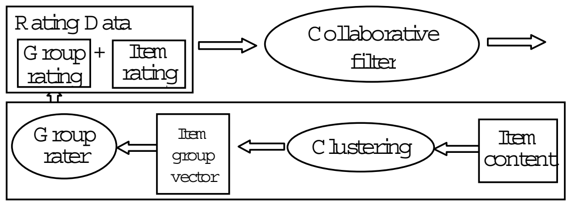

Figure 3. Overview of Our Approach.

1. Apply clustering algorithm to group the items, then use the result, which is represented by the fuzzy set, to create a group-rating matrix.
2. Compute the similarity: firstly, calculate the sub-similarity of group-rating matrix , then calculate the sub-similarity of item-rating matrix . At last, the total similarity is the linear combination of the above two.
3. Make a prediction for an item by performing a weighted average of deviations from the neighbor's mean.

2.1. Group rating

The goal of grouping ratings is to group the items into several cliques and provides content-based information for collaborative similarity calculation. Each item has it's own attributes, such as movie items, which may have actor, actress, director, genre, and synopsis as its attributes. Thus, we can group the items based on those attributes.

K-means Clustering Algorithm is a simple and fast clustering method, which has been popular used [8]. So we apply it to group the items with some adjustments. The difference is that we apply the fuzzy set theory to represent the affiliation between an object and a cluster. As shown in Figure 4 , firstly, items are grouped into a given number of clusters. After completion of grouping, the possibility of one object (here one object means one item) belonging to a certain cluster is calculated as follows.

$$Pro(j,k) = 1 - \frac{CS(j,k)}{MaxCS(i,k)} \tag{1}$$

where $Pro(j,k)$ means the possibility of object j belonging to the cluster k; The $CS(j,k)$ means the counter-similarity between the object j and the cluster k, which is calculated based on the Cosine method; $MaxCS(i,k)$ means the maximum counter-similarity between an object and cluster k.

However, in our adjusted k-means algorithm, the fuzzy membership in a cluster is only assigned at the last step. It seems to unessentially represent the fuzzy memberships of objects. So the fuzzy k-means algorithm [5] is also applied to group the items, in which each object is assigned a fuzzy membership during each iteration as Figure 4 shows.

The global cost function, membership between an object and a cluster, and the mean value of one cluster are calculated as follows.

$$GCF_{fuz} = \sum_{i=1}^{c}(\sum_{j=1}^{n}(Pro_{i,j}^{b} \times Dis_{i,j})) \tag{2}$$

$$Mean_j = \frac{\sum_{j=1}^{n}(Pro_{i,j})^b X_j}{\sum_{j=1}^{n} Pro_{i,j}^{b}} \tag{3}$$

$$Pro_{i,j} = \frac{(\frac{1}{Dis_{i,j}})^{\frac{2}{b-1}}}{\sum_{r=1}^{c}(\frac{1}{Dis_{r,j}})^{\frac{2}{b-1}}} \tag{4}$$

where, GCF_{fuz} means the fuzzy global cost function; c means the cluster number; b is a free parameter chosen to adjust the blending of different clusters; $Dis_{i,j}$ is the Euclidean distance between the mean value of cluster i and the object j; X_j is the vector of object j; $Pro_{i,j}$ means the membership between the cluster i and the object j.

2.2. Similarity

As we can see, after grouping the items, we get a new rating matrix. We can use the item-based collaborative algorithm to calculate the similarity and make the predictions for users. There are lots of different ways to compute the similarity. In our approach, we use two of them, and make a linear combination of their results.

Pearson correlation-based similarity. The most common measure for calculating the similarity is the Pearson correlation algorithm. Pearson correlation measures the degree to which a linear relationship exists between two variables. The Pearson correlation coefficient is derived from a linear regression model, which relies on a set of assumptions regarding the data, namely that the relationship must be linear, and the errors must be independent and have a probability distribution with mean 0 and constant variance for every setting of the independent variable [12].

$$sim(k,l) = \frac{\sum_{u=1}^{m}(R_{u,k} - \overline{R}_k)(R_{u,l} - \overline{R}_l)}{\sqrt{\sum_{u=1}^{m}(R_{u,k} - \overline{R}_k)^2}\sqrt{\sum_{u=1}^{m}(R_{u,l} - \overline{R}_l)^2}} \tag{5}$$

where $sim(k,l)$ means the similarity between the item k and l; m means the total number of users, which rated on both the item k and l; $\overline{R}_k, \overline{R}_l$ are the average ratings of the item k and l, respectively; $R_{u,k}, R_{u,l}$ mean the rating of user u on the item k and l respectively.

```
Algorithm 1: Adjusted K-means Clustering
Input : the number of clusters k and items attribute features.
Output: a set of k clusters that minimizes of the squared-error criterion, and the probability of each item belonging to each cluster center are represented as a fuzzy set.

(1) Arbitrarily choose k objects as the initial cluster centers;
(2) Repeat (a) and (b) until small change;
    (a) (Re) assign each object to the cluster to which the object is the most similar, based on the mean value of the objects in the cluster;
    (b) Update the cluster means, i.e., calculate the mean value of the objects of each cluster;
(3) Compute the possibility between objects and each cluster center.
```

```
Algorithm 2: Fuzzy K-means Clustering
Input: the number of clusters k and items attribute features.
(1) Initialize the parameters, and membership between objects and clusters;
(2) Repeat (a) and (b) until global cost function has small change;
    a) Recompute the mean value of each cluster.
    b) Recompute the membership of each object.
(3) Return the membership.
```

Figure 4. Item clustering.

Adjusted cosine similarity. Cosine similarity once has been used to calculate the similarity of users but it has one shortcoming. The difference in rating scale between different users will result in a quite different similarity. For instance, if Bob only rates score 4 on the best movie, he never rates 5 on any movie; and he rates 1 on the bad movie, instead of the standard level score 2. But Oliver always rates according to the standard level. He rates score 5 on the best movie, and 2 on the bad movie. If we use traditional cosine similarity, both of them are quite different. The adjusted cosine similarity [16] was provided to offset this drawback.

$$sim(k,l) = \frac{\sum_{u=1}^{m}(R_{u,k} - \overline{R}_u)(R_{u,l} - \overline{R}_u)}{\sqrt{\sum_{u=1}^{m}(R_{u,k} - \overline{R}_u)^2}\sqrt{\sum_{u=1}^{m}(R_{u,l} - \overline{R}_u)^2}} \tag{6}$$

where $sim(k,l)$ means the similarity between the item k and l; m means the total number of users, who rates on both the item k and l; $\overline{R}_u$ are the average ratings of the user u; $R_{u,k}$, $R_{u,l}$ mean the rating of the user u on the item k and l respectively.

Linear combination of similarity. Due to the difference in value range between the item-rating matrix and group-rating matrix, we use different methods to calculate the similarity. As for the item-ratings matrix, the rating value is integer; As for the group-rating matrix, it is the real value ranging from 0 to 1. The natural way is to enlarge the continuous data range from [0 1] to [1 5] or reduce the discrete data range from [1 5] to [0 1] and then apply the Pearson correlation-based algorithm or adjusted cosine algorithm to calculate the similarity. We call this enlarged ICHM. We also propose another method: firstly, use the Pearson correlation-based algorithm to calculate the similarity from the item-rating matrix, and then calculate the similarity from the group-rating matrix by the adjusted cosine algorithm, at last, the total user similarity is a linear combination of the above two, we call this combination ICHM.

$$sim(k,l) = sim(k,l)_{item} \times (1-c) + sim(k,l)_{group} \times c \tag{7}$$

where $sim(k,l)$ means the similarity between the item k and l; c means the combination coefficient; $sim(k,l)_{item}$ means that the similarity between the item k and l, which is calculated from the item-rating matrix; $sim(k,l)_{group}$ means that the similarity between the item k and l, which is calculated from the group-rating matrix.

2.3. Collaborative prediction

Prediction for an item is then computed by performing a weighted average of deviations from the neighbor's mean. Here we use top N rule to select the nearest N neighbors based on the similarities of users. The general formula for a prediction on the item i of user k is:

$$P_{u,k} = \overline{R}_k + \frac{\sum_{i=1}^{n} (R_{u,i} - \overline{R}_i) \times sim(k,i)}{\sum_{i=1}^{n} |sim(k,i)|} \tag{8}$$

where $P_{u,k}$ represents the predication for the user u on the item k; n means the total neighbors of item k ; $R_{u,i}$ means the user u rating on the item i; $\overline{R}_k$ is the average ratings on item k; $sim(k,i)$ means the similarity between item k and its' neighbor i; $\overline{R}_i$ means the average ratings on the item i.

2.4. Cold start problem

In traditional collaborative filtering approach, it is hard for pure collaborative filtering to recommend a new item to users since no user make any rating on this new item. However, in our approach, based on the item group information, we can make predictions for the new item. In our experiment, it shows a good recommendation performance for the new items. In Equation 8, $\overline{R}_k$ is the average rating of all ratings on item k. As for the new item, no user makes any rating on items, $\overline{R}_k$ should be zero. Since this standard baseline of user ratings equals to zero, it is unreasonable for us to apply Equation 8 to new items. Therefore, for new item, we use two methods, one is using the $\overline{R}_{neighbors}$, the average rating of all ratings on the new items' nearest neighbors instead of $\overline{R}_k$, which is inferred by the group-rating matrix; the other is using a weighted sum method for prediction computation proposed by [16].

$$P_{u,k} = \frac{\sum_{i=1}^{n} R_{u,i} \times sim(k,i)}{\sum_{i=1}^{n} |sim(k,i)|} \tag{9}$$

where $P_{u,k}$ represents the predication for the user u on the item k; n means the total neighbors of the item k; $R_{u,i}$ means the user u rating on the item i; $sim(k,i)$ means the similarity between the item k and its' neighbor i. We can observe this method avoid calculating the average ratings of the item k. It is quite suitable for cold start problem. In our experiment, we implement both methods and make a comparison of them.

Table 1. Item-rating

	Tom	$Jack$	$Oliver$
Gone with the Wind	5	3	
Swordfish	5	2	4
Pearl Harbor	2	5	
Hero	4	2	
The Sound of Music			

Table 2. Group-rating

	$Cluster1$	$Cluster2$
Gone with the Wind	98%	0.13%
Swordfish	100%	0.02%
Pearl Harbor	1.0%	95%
Hero	95%	1.2%
The Sound of Music	0.12%	98%

2.5. A Scenario of our approach

- **Users name:** *Tom, Jack,and Oliver*
- **Items(movie):**
  - Title of items: *Gone with the Wind, Swordfish, Pearl Harbor, Hero, The sound of Music*
- **Rating:**1 $\sim$ 5 integer score
  - *very bad=1, bad=2, common=3, good=4, very good=5*

The following is a simple procedure of our approach.

1. Based on the item contents, such as movie genre, director, actor, actress, and even synopsis, we apply clustering algorithm to group the items. Here, we

Table 3. Prediction

	Tom	*Jack*	*Oliver*
Gone with the Wind	5	3	4.5
Swordfish	5	2	4
Pearl Harbor	2	5	4
Hero	4	2	3.5
The Sound of Music	0.5	3.5	2.5

use fuzzy set to represent the clustering result. Assume the result is as follows: Cluster 1: *Gone with the Wind (98%), Swordfish (100%), Pearl Harbour (1.0%), Hero (95%), The Sound of Music (0.12%),* Cluster 2: *Gone with the Wind (0.13%), Swordfish (0.02%), Pearl Harbour (95%), Hero (1.2%), The Sound of Music (98%),* the number in the parenthesis following the movie name means the probability of the movie belonging to the cluster.

2. We use group-rating engine to make a group-rating matrix. As Table 2 shows.

3. Combine the group-rating matrix and item-rating matrix to form a new rating matrix. Now, we can calculate the similarity between items based on this new unified rating data matrix. The similarity between items consists of two parts. The first part calculates the similarity based on user ratings, using the Pearson correlation-based algorithm. The second part calculates the similarity based on the clustering result by using adjusted cosine algorithm. The total similarity between items is the linear combination of them. For example, when we calculate the similarity between *Gone with the Wind* and *Swordfish,* firstly, $sim(G,S)_{item}$ and $sim(G,S)_{group}$ are calculated based on Equation 5 and 6 respectively.

   $$sim(G,S)_{item} = \frac{(5-4)\times(5-3.5)+(3-4)\times(2-3.5)}{\sqrt{(5-4)^2+(3-4)^2}\sqrt{(5-3.5)^2+(3.5-2)^2}}$$

   $$sim(G,S)_{group} = \frac{(0.98-0.59)\times(1-0.59)+(0.013-0.39)\times(0.002-0.39)}{\sqrt{(0.98-0.59)^2+(0.013-0.39)^2}\sqrt{(1-0.59)^2+(0.002-0.39)^2}}$$

   Secondly, $sim(G,S)$ is calculated based on Equation 7, here the combination coefficient is 0.4.

   $$sim(G,S) = 1\times(1-0.4)+0.9999\times 0.4 = 0.9999$$

4. Then, predictions for items are calculated by performing a weighted average of deviations from the neighbor's mean. The result of prediction can be observed from Table 3.

In the example, we can observe, the item - *The Sound of Music*, which no one make any rating on, can be treated as a new item. In traditional item-based collaborative method, which makes prediction only based on item-based matrix (Table 1), it is impossible to make predictions on this item. However, in our approach, we can make predictions for users on this item, based on the group-rating matrix (Table 2). From the description of our approach, we can observe that this approach can fully realize the strengths of content-based filtering, mitigating the effects of the new user problem. In addition, when calculating the similarity, our approach considers the information not only from personal tastes but also from the contents, which provides a latent ability for better prediction and makes serendipitous recommendation.

3. Experimental evaluations

3.1. Data Set and Evaluation Metrics

Currently, we perform experiments on movie rating data collected from the MovieLens web-based recommender system. The data set contained 100,000 ratings from 943 users and 1,682 movies, with each user rating at least 20 items. We divide data set into a training set and a test data set.

MAE (Mean Absolute Error) [10] has been widely used in evaluating the accuracy of a recommender system by comparing the numerical recommendation scores against the actual user ratings in the test data. The MAE is calculated by summing these absolute errors of the corresponding rating-prediction pairs and then computing the average.

3.2. Behaviors of our Method

We implement group-rating method described in section 2.1 and test them on MovieLens data with the different number of clusters. Figure 5 shows the experimental results. It can be observed that the number of clusters does affect the quality of prediction. As we have discussed before, the fuzzy k-means algorithm seems more essentially represent the fuzzy membership than the adjusted k-means algorithm. However, in our experiment, it does not show obvious advantages. Since the computation complexity of fuzzy k-means algorithm is heavier than adjusted k-means algorithm, we choose our adjusted k-means algorithm in following parts.

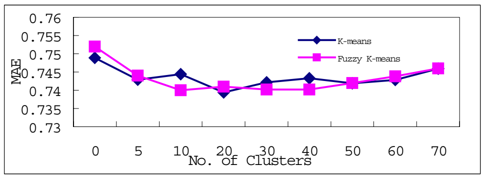

Figure 5. Clustering

In order to find the optimal combination coefficient c in the Equation 7, we conduct a series of experiments by changing combination coefficient from 0 to 1 with a constant step 0.1. Figure 6 shows that when the coefficient arrives at 0.4, an optimal recommendation performance is achieved.

We also apply clustering method to group the user profile instead of the item content information in the user-based collaborative algorithm. We call this method UCHM (user-based clustering hybrid method) [11], in which the impact of combination coefficient is quite similar to the ICHM, as Figure 6 shows.

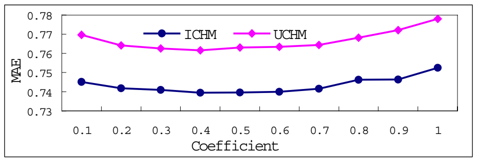

Figure 6. coeffecient

From the Figure 7, it can be observed that the performance of combination ICHM is the best, and the second is the enlarged ICHM, which is followed by the item-based collaborative method. The last is UCHM. We also can observe that the size of neighborhood does affect the quality of prediction. The performance improves as we increase the neighborhood size from 10 to 30, then tend to be flat.

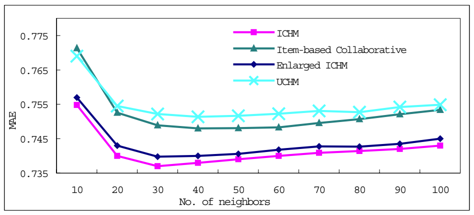

Figure 7. Comparison

When we apply clustering method to movie items, we use the item attribute - movie genre. However, our approach can consider more item attributes, such as actor, actress,director, and even the synopsis. In order to observe the effect of item attributes, we collect the 100 movie synopsis from Internet Movie Database (http://www.imdb.com) to provide attribute information for clustering movies. In our experiment, it shows that the correct attributes of movies can further improve the performance of recommender system, as Figure 8 shows.

As for cold start problem, we choose the items from the training data set and delete all the ratings of those items, thus we can treat them as new items. In section 2.4 we describe two methods for the new item prediction. The first method applies the average rating of ratings from the nearest neighbor, so we call it average method in short. The second method is the weighted sum, so we call it weighted sum method. When a new item coming, no one make any rating on it. According to this, we randomly select one item, and delete all of his ratings and treat him as a new item. In our experiment, the randomly selected item is No.946. In the training data, user made some ratings on item 946, we delete all of those ratings and treat item 946 as a new item. Then, we apply two methods described in section 2.4 to make predictions for the new item. In the test data, item 946 has 11 ratings, which is described by the bar real rating in Figure 9. We can observe that the prediction for the new user can partially reflect the user preference.

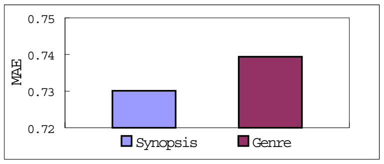

Figure 8. Item attribute

Table 4. MAE of new items

MAE	10	20	30	50
Weighted sum method	0.767	0.875	0.855	1.04
Average method	0.564	0.755	0.834	0.958

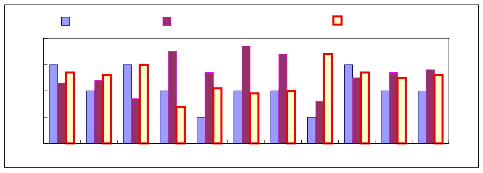

Figure 9. Cold start Problem

To generalize the observation, we randomly select the number of items from 10 to 50 with the step of 10 from the test data, and delete all the ratings of those items and treat them as new items. We can observe from Table 4 that the average method shows a better performance than the weighted sum method.

Since our method has the ability to handle cold start problem, we compare the performance of our approach under the cold-start case and the no-cold start case. In our experiment, we randomly selected 100 items, and delete all the ratings of those items and treat them as new items. The result can be observed from Figure 10. Here, the MAE is the total MAE over all items including new items, while the

MAE in the Table 4 is calculated only based on new items. Although in cold start case we can not achieve the same good performance as in no-cold start case, our approach can treat the new users differently based on their own features, and make predictions for them.

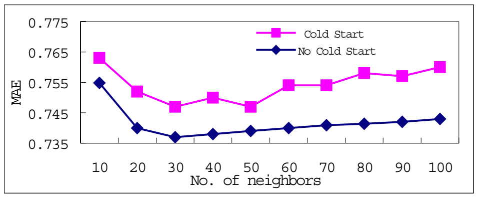

Figure 10. Cold start VS no-cold start

4. Our method versus the classic one

Although some hybrid recommender system has already exited, it is hard to make an evaluation among them. Some systems [4] use Boolean value (relevant or irrelevant) to represent user preferences, while others use numeric value. The same evaluation metrics cannot make a fair comparison. Further more, the quality of some systems depends on the time, in which system parameters are changed with user feedback [3], and Claypool do not clearly describe how to change the weight with time passed. However, we can make a simple concept comparison. In Fab system [2], the similarity for prediction is only based on the user profiles, which is a special case of UCHM that the combination coefficient is 1. As the Figure 7 shows, the performance of UCHM is worse than ICHM, so we can conclude our ICHM shows a better performance than the Fab system.

5. Conclusions

We apply the clustering technique to the item content information to complement the user rating information, which improves the correctness of collaborative predication, and solves the cold start problem. Our work indicates that the correct application of the item information can improve the recommendation performance.

References

[1] R. Baeza-Yates and B. Riberio-Neto. *Modern Information Retrieval*. Addison-Wesley Publishers, New York, 1999.

[2] M. Balabanovic and Y. Shoham. Fab: Content-based, collaborative recommendation. *Communications of the ACM*, 40(3):66–72, 1997.

[3] M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes, and M. Sartin. Combining content-based and collaborative filters in an online newspaper. *In Proc. ACM-SIGIR Workshop on Recommender Systems: Algorithms and Evaluation*, 1999.

[4] J. Delgado, N. Ishii, and T. Ura. Content-based collaborative information filtering: Actively learning to classify and recommend documents. *In Proc. Second Int. Workshop, CIA'98*, pages 206–215, 1998.

[5] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classfication*. Wiley-Interscience Publication, New York, 2000.

[6] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestrye. *Communications of the ACM*, 35(12):61–70, January 1992.

[7] D. Gupta, M. Digiovanni, H. Narita, and K. Goldberg. Jester 2.0: A new linear-time collaborative filtering algorithm applied to jokes. *In Proc. ACM-SIGIR Workshop on Recommender Systems: Algorithms and Evaluation*, 1999.

[8] J. Han and M. Kamber. *Data mining: Concepts and Techniques*. Morgan-Kaufman, New York, 2000.

[9] D. B. Hauver. Flycasting: Using collaborative filtering to generate a play list for online radio. *In Int. Conf. on Web Delivery of Music*, 2001.

[10] J. Herlocker, J. Konstan, B. A., and J. Riedl. An algorithmic framework for performing collaborative filtering. *In Proc. ACM-SIGIR Conf.*, pages 230–237, 1999.

[11] Q. Li and B. Kim. An approach for combining content-based and collaborative filters. *In Proc. of IRAL2003*, 2003.

[12] J. T. McClave and F. H. Dietrich. *Statistics*. Ellen Publishing Company, San Francisco, 1998.

[13] D. Oard and G. Marchionini. A conceptual framework for text filtering. *Technical Report EE-TR-96-25, CAR-TR-830, CS-TR3643*, 1996.

[14] P. Resnick, N. Iacovou, M. Suchak, P. Bergstorm, and J. Riedl. Grouplens: An open architecture for collaborative filtering of netnews. *In Proc. ACM Conf. on Computer-Supported Cooperative Work*, 1994.

[15] G. Salton and McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, New York, 1983.

[16] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. *In Proc. Tenth Int. WWW Conf.*, pages 285–295, 2001.

[17] J. B. Schafer, J. Konstan, and J. Riedl. Recommender systems in e-commerce. *In Proc. 1st ACM. Conf. on Electronic Commerce (EC'99)*, 1999.

[18] U. Shardanand and Maes. Social information filtering: Algorithms for automating.

[19] D. B. Terry. A tour through tapestry. *In Proc. ACM Conf. on Organizational Computing Systems (COOCS)*, pages 21–30, 1993.

[20] A. M. A. Wasfi. Collecting user access patterns for building user profiles and collaborative filtering. *In Int. Conf. on Intelligent User Interfaces*, pages 57– 64, 1999.

[21] K. Wittenburg, D. Das, W. Hill, and L. Stead. Group asynchronous browsing on the world wide web. *In Proc. of Fourth International World Wide Web Conference*, pages 51–62, 1995.