J Intell Inf Syst (2006) 27: 79–91
DOI 10.1007/s10844-006-8771-2

A new approach for combining content-based and collaborative filters

Byeong Man Kim · Qing Li · Chang Seok Park · Si Gwan Kim · Ju Yeon Kim

Received: 18 September 2002 / Revised: 12 October 2004 /
Accepted: 13 May 2005

Abstract With the development of e-commerce and the proliferation of easily accessible information, recommender systems have become a popular technique to prune large information spaces so that users are directed toward those items that best meet their needs and preferences. A variety of techniques have been proposed for performing recommendations, including content-based and collaborative techniques. Content-based filtering selects information based on semantic content, whereas collaborative filtering combines the opinions of other users to make a prediction for a target user. In this paper, we describe a new filtering approach that combines the content-based filter and collaborative filter to capitalize on their respective strengths, and thereby achieves a good performance. We present a series of recommendations on the selection of the appropriate factors and also look into different techniques for calculating user-user similarities based on the integrated information extracted from user profiles and user ratings. Finally, we experimentally evaluate our approach and compare it with classic filters, the result of which demonstrate the effectiveness of our approach.

Keywords Information filtering · Collaborative filtering · Content-based filtering · Recommendation system

B. M. Kim · Q. Li (✉) · C. S. Park · S. G. Kim
Department of Software Engineering, Kumoh National Institute of Technology, 1, Yangho-dong, Gumi, 730-701 Gyeongbuk, Republic of Korea
e-mail: liqing@icu.ac.kr, Kooliqing@gmail.com

Q. Li
Department of Engineering, Information & Communications University, 119 Munji-ro, Yuseong-gu, 305-732 Daejeon, Republic of Korea

J. Y. Kim
Bucheon College, 424 Simkok-dong, Wonmi-gu, Buchon, Gveonggi-do 421-735, Republic of Korea

1 Introduction

Recently we are often overwhelmed by the massive volume of data available on the Internet, and in this environment we must make decisions regarding the consumption of information. In our daily lives, critics do much of our information filtering. For instance, we check ranking lists for bestsellers and listen to movie critics. However, these utilities cannot sufficiently and quickly filter the information to which we are exposed. Fortunately, information filtering technology offers a potential solution to this problem.

Most information filtering methods fall into one of the following categories, content-based filtering (CBF) or collaborative filtering (CF) (Oard & Marchionini, 1996). CBF selects the right information for users by comparing representations of searching information to representations of contents of user profiles that express the interests of users. For example, search engines recommend web pages with contents similar to user queries (Salton & McGill, 1983). CBF has proven to be effective in recommending textual items relevant to a topic using techniques such as Boolean queries (Verhoeff, Goffman, & Belzer, 1961; Anick et al., 1990; Lee, Kim, & Lee, 1993), vector-space queries (Salton & Buckley, 1988), probabilistic models (Robertson & Sparck Jones, 1976), neural networks (Wilkinson & Hingston, 1991; Kim & Raghavan, 2000), and fuzzy set models (Ogawa, Morita, & Kobayashi, 1991). However, CBF also has the following limitations:

- It cannot easily provide serendipitous recommendations, because all the information is selected and recommended based on content.
- It is hard for novices to use content-based systems effectively.

CF is a technology wherein peer opinions are employed to predict the interests of others. Goldberg and his colleagues first applied CF technology to recommender systems (Goldberg, Nichols, Oki, & Terry, 1992; Terry, 1993). Free annotations or explicit *like it* or *hate it* annotations were applied to identify like-minded users manually. GroupLens (Resnick, Iacovou, Suchak, Bergstorm, & Riedl, 1994) and Ringo (Upendra & Patti, 1995) systems, developed independently, were the first to automate prediction. The techniques of CF have been developed quickly not only in the research area but also in the commercial field. There are plenty of collaborative recommender systems, which have been designed for different purposes. These include MovieLens system, which recommends movies, Jeter system, which recommends jokes (Gupta, Digiovanni, Narita, & Goldberg, 1999), Flycasting, which recommends online radio (Hauver & French, 2001), and GAB, which recommends web pages based on bookmarks (Wittenburg, Das, Hill, & Stead, 1995). Recently, more and more companies employ or provide recommender system solutions.

Although CF can improve the quality of recommendations based on user ratings, it completely ignores any information that can be extracted from semantic content. Thus the quality of recommendation completely depends on the user ratings without semantic content. Clearly, CBF does not suffer from the above problems. As such, combining both types of filters in order to achieve better filtering performance will capitalize on the strengths offered by each approach.

In this paper, we introduce a method using a clustering algorithm to combine the content-based and collaborative filters. First, we group the user profiles into clusters in order to provide the semantic content information. We then treat those clusters as

items to form a new user-item matrix for recommendations. Finally, we make predictions for users by using the classic collaborative algorithm based on this new user-item matrix.

2 Related works

The hybrid recommendation system, which combines content-based filters and collaborative filters, capitalizes on the strengths of each method. Most hybrid recommender systems fall into one of the following three models.

The first is the linear combination model, which combines the results of collaborative and content-based filters as shown in Figure 1. For example, ProfBuilder (Wasfi, 1999) recommends web pages using both content-based and collaborative filters, and each filter creates a recommendation list instead of a combined list for users. Claypool et al. (1999) 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. However, the authors do not clearly describe how to decide the weights of collaborative and content-based components.

The second is the sequential combination model, shown in Figure 2. In this model, first, user profiles are constructed by a CBF algorithm based on the items. Second, a collaborative algorithm is applied to make predictions based on those user profiles, such as that employed in RAAP (Delgado, Ishii, & Ura, 1998) and Fab filtering systems (Balabanovic & Shoham, 1997). RAAP is a content-based collaborative information filtering technique that helps the user to classify domain specific information found on the WWW, and also recommends those URLs to other users with similar interests. To determine the similarity of interests among users, a scalable Pearson correlation algorithm based on the web page category is used. The Fab system uses content-based techniques instead of user ratings to create user profiles. Hence the quality of predictions is fully dependent on the content-based techniques, and inaccurate profiles result in inaccurate correlations with other users, thus yielding poor predictions.

The last is the mixed combination model, in which both semantic content and ratings are applied to make recommendations; these include the probabilistic model (Popescul, Ungar, Pennock, & Lawrence, 2001) and Ripper system for recommendation (Basu & Cohen, 1998). Basu and Cohen (1998) trained a Ripper machine learning system with a combination of content data and rating data in an effort to produce better recommendations. Good et al. (1999) combined personal IF agents and the ratings of users to make recommendations. Popescul et al. (2001) provided a probabilistic model for unified collaborative and content-based recommendations.

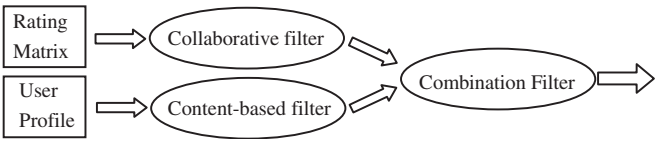

Fig. 1 Linear combination

Fig. 2 Sequential combination

In this paper, we apply clustering techniques to integrate the semantic contents of user profiles into CF to improve its recommendation. In the following section, we describe our approach in detail.

3 Our approach

In our approach, we integrate the semantic contents from user profiles and user ratings to calculate user-user similarity as shown in Figure 3. The procedure of our approach can be described as follows:

1. Gather the information to create user profiles.
2. Apply a clustering algorithm to group the user profiles, then use the clustering result to form a group-rating matrix.
3. Calculate the user-user similarity: first, calculate the sub user-user similarity of the group-rating matrix by using an adjusted-cosine algorithm, and then calculate the sub user-user similarity of the user-rating matrix by using a Pearson correlation-based algorithm. Finally, the total user-user similarity is the linear combination of the above two.
4. Make a prediction for an item by performing a weighted average of deviations from the neighbor's mean.

For example, let us assume that there are four users. First, we group the four users into two clusters based on the semantic contents from the user profiles. Then, we treat those two clusters as items to form a new user-item matrix for the CF. Finally, recommendations for users are made based on this new user-item matrix by applying the CF algorithm.

3.1 User profile

The goal of applying clustering techniques is to group users into several cliques and provides semantic content information for a collaborative similarity calculation. To accomplish this, we need to define representations of the user profiles.

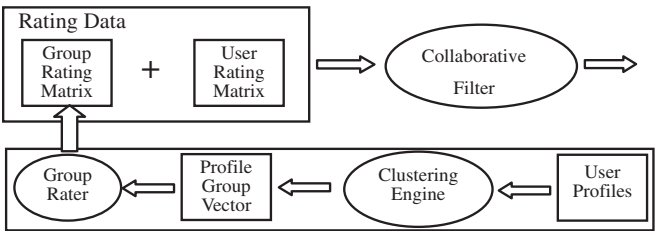

Fig. 3 Overview of our approach

We use two methods to create the user profile, a manual weighted method for creating the user profile and an auto-weighted method.

User profiles indicate the information needs or preferences related to items of interest to the users . A user profile consists of several profile vectors and each profile vector represents an aspect (or dimension of his or her preferences); each aspect contains its own attributes and co-pair weights. For example, in a movie recommender system, a movie item contains five aspects—actor, actress, director, genre, and synopsis, and a profile vector consists of several attribute-value pairs.

For the manual weighted method, the user not only expresses what kinds of item attributes that he or she favors, but also expresses the degree to which he or she likes those items.

$$\{Aspect_1 : (a_{1,1}(w_{1,1}), ...a_{1,m}(w_{1,m})), ...Aspect_i : (a_{i,1}(w_{i,1}), ...a_{i,m}(w_{i,m}))\} \quad (1)$$

where $a_{i,m}$ denotes the item attribute m in aspect i of the user profile, and $w_{i,m}$ is the weight of item attribute m in aspect i of the user profile, which is assigned by the user.

At the beginning stage, users should specify their preferences; however, as the user ratings on items increase, the weights of item attributes in a certain aspect of the user profile can be automatically constructed by the following simple equation in order to reduce the burden of users. We call this the auto-weighted method.

$$W_{n,m} = \frac{Num_{\text{item} \subseteq \text{attribute}_m \text{ of } \text{aspect}_n \mid \text{item} > \text{threshold}}}{Num_{item > threshold}} \quad (2)$$

where $Num_{\text{item} > \text{threshold}}$ is the number of items, in which the ranking value is larger than the threshold; $Num_{\text{item} \subseteq \text{attribute}_m \text{ of } \text{aspect}_n}$ is the number of items containing *attribute m* in *aspect n* of the user profile and its rating is larger than the threshold. In our present experiment, we set the value of the threshold as 3. For example, let us assume that user Tom makes ratings on three movie *Gone with the Wind, Hero, and Swordfish*. And all of the ratings are larger than the threshold 3. From the genre information, we know *Gone with the Wind* belongs to romance genre, while *Swordfish* and *Hero* belong to the action genre. Thus Tom's profile is as follows. *Tom = {Genre: (romance (1/3), action (2/3))}*. The user also can change the automatic weight by explicitly specifying a weight to overwrite the original one.

The characteristics of the synopsis aspect are somewhat different from other aspects. Hence, we take a different approach for this aspect. A weighted keyword vector is constructed for each user. First, keywords are extracted from the synopsis field of the movie description record for which the user expresses interests, that is, the user rating is larger than the threshold. Second, the weight of each keyword is calculated. The widely used $tf \times idf$ formula (Baeza-Yates & Riberio-Neto, 1999) in information retrieval literature is applied for calculating the keyword weight.

3.2 Group rating

After representing the profiles, the next is to group the user profiles to form a group-rating matrix. To this end, we build cliques over all users.

The popularly used K-means Algorithm (MacQueen, 1967), a simple and fast clustering method, is applied with some adjustments in order to group the users. For our purposes, we apply a fuzzy set theory to represent the affiliation between an

Algorithm 1: Adjusted K-means Clustering

Input : the number of clusters k and item 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 that is represented as a fuzzy set.

(1) 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 for each cluster;
(3) Compute the possibility between each object and each

Fig. 4 Adjusted k-means clustering algorithm

object and a cluster. As shown in Figure 4, first, user profiles are grouped into a given number of clusters. After completion of grouping, the possibility of one object (here one object means one user profile) belonging to a certain cluster is calculated as follows.

$$Pro(j,k) = 1 - \frac{CS(j,k)}{MaxCS(i,k)} \tag{3}$$

where $Pro(j,k)$ is the possibility of object j belonging to cluster k; $CS(j,k)$ is the counter-similarity between the object j and cluster k, which is calculated based on the Euclidean distance; and $MaxCS(i,k)$ is 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. Hence the fuzzy k-means algorithm (Duda, Hart, & Stork, 2000) is also applied to group the items, wherein each object is assigned a fuzzy membership during each iteration.

Regardless of the algorithms that are employed, the method of choosing the initial cluster center is a critical issue. In this regard, we recommend the refinement algorithm suggested by Bradley and Fayyad (1998).

3.3 Similarity computation

After grouping the user profiles, we obtain a new rating matrix. We can then use the classic collaborative algorithms to calculate the similarity between users and make predictions. There are numerous different ways to calculate the similarity between users. The most common measure for calculating similarity is the Pearson correlation algorithm (Terry, 1993). Another one is cosine correlation algorithm. Considering the fact that users with similar interests in items may have very different rating patterns: some users tend to assign a higher rating to all items than other users. The adjusted cosine correlation algorithm (Sarwar, Karypis, Konstan, & Riedl, 2001) is therefore used to offset this drawback.

Due to the difference in the value scale between the user-rating matrix and the group-rating matrix, we use different methods to calculate the similarity. For the user-rating matrix, the value is an integer; for the group-rating matrix, it is a real value ranging from 0 to 1. The natural way to normalize the value is to enlarge the continuous data scale from [0 1] to [1 5] or decrease the discrete data scale from [1 5] to [0 1] and then calculate the user-user similarity. In our experiment, we found that this method improved the precision. We also propose another method: first, we use the Pearson correlation-based algorithm to calculate the user-user similarity from the user-rating matrix, and then calculate the user-user similarity from the group-rating matrix by the adjusted cosine algorithm. Finally, the total user similarity is the linear combination of the above two, as given by Eq. 4. In our experiment, the latter approach displays superior performance.

$$sim(k,l) = sim(k,l)_{user} \times (1-c) + sim(k,l)_{group} \times c \tag{4}$$

where $sim(k,l)$ denotes the similarity between user k and his or her neighbor l; c is the combination coefficient; $sim(k,l)_{user}$ is the similarity between user k and his neighbor l, which is calculated from the user-rating matrix; and $sim(k,l)_{group}$ is the similarity between user k and his or her neighbor l, which is calculated from the group-rating matrix. Here we should point out that as the strengths of the group information and item ratings change, the optimum value of c will also change. For example, at the initial stage of a recommender system, due to lack of user ratings, the group-rating matrix plays a major role. Therefore, the weight of the group-rating matrix is relatively higher at that time. Later, with an increase of user ratings, the weight of the group-rating matrix will decrease. Hence, we suggest a simple adaptive solution to automatically find of the optimal value as outlined in Figure 5.

3.4 Collaborative prediction

Prediction for an item is then calculated by performing a weighted average of deviations from the neighbor's mean. Here we use the top N rule to select the nearest N neighbors based on the similarities of users. The general formula (Resnick et al., 1994) for a prediction on item i by user k is:

$$P_{k,i} = \overline{R}_k + \frac{\sum_{u=1}^{n}(R_{u,i} - \overline{R}_u) \times sim(k,u)}{\sum_{u=1}^{n}|sim(k,u)|} \tag{5}$$

Algorithm 3: Adaptive Solution for Combination Coefficient

Set the initial value as 0.5.

Decrease the initial value by a constant value 0.1.

If the performance also decreases,

    Increase the initial value by a constant value 0.1, until the performance decreases again

Else

    Continue to decrease the value by a constant value 0.1 until the performance increases.

Fig. 5 Adaptive solution for combination coefficient

where $P_{k,i}$ represents the prediction for user k of item i; n denotes the top N nearest neighbors of user k; and $\overline{R}_u$ represents the average ratings of user u on the items.

3.5 New user problem

In the classic CF approach, it is difficult to make predictions for a new user since this user does not make any ratings on items. However, in our approach, based on the grouping information, we can make predictions for the new user with the manual user profile. Notice that although in our approach new users should have their own manual profiles before entering the system, such user profiles can be easily constructed by an abstract specification; for instance, Tom specifies "*I like suspense movie (80%)*". However, in the case of pure CF, before using the system, new users have to view some movies and subsequently make ratings. This is time-consuming and a potential deterrent to prospective users.

In Eq. 5, $\overline{R}_k$ is the average rating of user k on items. For a new user, because he or she has not made any ratings on items, $\overline{R}_k$ should be the zero. Since $\overline{R}_k$ is the standard baseline of the user ratings and it is zero for a new user, it is unreasonable for us to apply Eq. 5 to the new user. Therefore, for a new user, we use $\overline{R}_{neighbors}$, the average rating of all ratings on the new user's nearest neighbors instead of $\overline{R}_k$, where $\overline{R}_{neighbors}$ is inferred from the group-rating matrix. In our approach, we use this method to make predictions for new users and apply Eq. 5 for existing users respectively.

4 Experimental evaluations

4.1 Data set & evaluation metric

We performed experiments on Each-movie data set. There are 1,623 movies in this data set and 61,265 users with a total of over 2.8 million ratings. The rating scale is discrete, taking values from 1 to 5. Note, the original ratings are between 0 and 1 and have been mapped. The sparsity of this data set is defined as $1 - \frac{nonzero\ \ entries}{total\ \ entries}$. Thus the sparsity of Each-movie data set is 0.9717. We divide the data set into a training set and a test data set. Twenty percent of users are randomly selected to be the test users, and the rest serves as the training set.

MAE (Mean Absolute Error) Breese, Heckerman, and Kardie (1998) is used as the metric in our experiments, which is calculated by summing these absolute errors of corresponding rating-prediction pairs and then computing the average. A lower *MAE* indicates greater accuracy.

4.2 User profiles

In the Each-movie data set, there are no user preferences. Therefore, we have to construct users from the training data set using the auto-weighted method outlined in Section 3.1. Each-movie data only provides genre information, and therefore we collect the movie synopsis, actor, actress, and director information from the Internet Movie Database (http://www.imdb.com) to provide more information for constructing the user profiles.

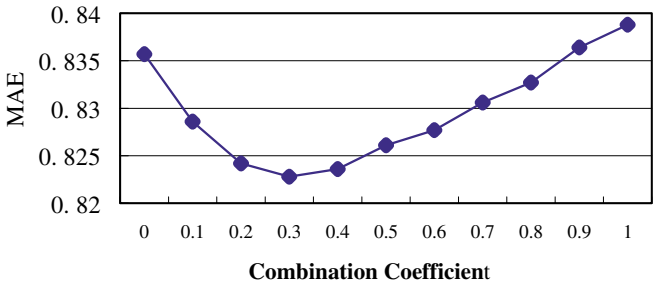

Fig. 6 The combination coefficient, number of clusters is 30, neighbor size 40

4.3 Behaviors of our method

4.3.1 The combination coefficient

As noted earlier, the optimal value of the combination coefficient changes according to the different cluster strengths. We use an adaptive solution to find the optimal value for our test data. The result are given in Figure 6. The optimal value for our test data is 0.3.

4.3.2 Number of clusters

We implement group-rating methods described in Section 3.2 and carry out our experiments with different numbers of clusters. Figure 7 shows the experimental results. It can be observed that the number of clusters affects the quality of prediction. As discussed previously, the fuzzy k-means algorithm appears to represent the fuzzy membership better than the adjusted k-means algorithm. However, in our experiments, it does not show clear advantages. Since the computation complexity of the fuzzy k-means algorithm is heavier than the adjusted k-means algorithm, we use the latter in the following sections.

4.3.3 Methods for computing user-user similarity

As we have observed, the value scale of the group-rating matrix and user-rating matrix is different. Hence we should modify the value to the same scale or

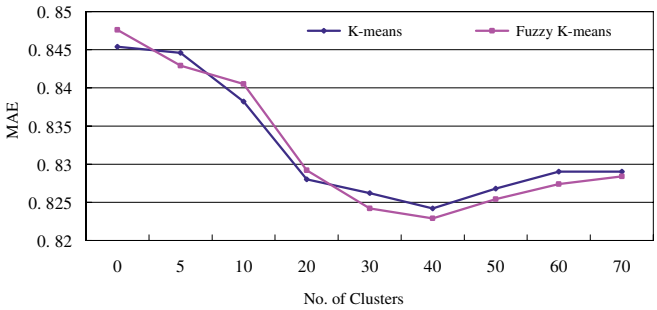

Fig. 7 Clustering

separately compute the user-user similarity. In our experiments, we enlarge the value scale of the group-rating matrix from [0 1] to [0 5], and use the Pearson correlation-based algorithm to calculate the similarity based on a new rating matrix. We call this method the **enlarged Pearson correlation-based approach**. At the same time, we also use the Pearson correlation-based algorithm to calculate the similarity based on the rating matrix, which consists of the group-rating matrix and the user-rating matrix. We call this method the **non-enlarged Pearson correlation-based approach**. Finally, the Pearson correlation-based algorithm is applied to calculate the user-user similarity from user-rating matrix, and the adjusted cosine algorithm is applied to calculate the user-user similarity from the group-rating matrix. The total user-user similarity is then calculated as the linear combination of the above two. We call this the **linear combination approach**.

Figure 8 shows that the linear combination method displays the best performance, followed by the enlarged Pearson method and the classic Pearson method. The non-enlarge Pearson correlation-based method displays the poorest performance.

4.3.4 Neighborhood size

The size of the neighborhood has a significant effect on the prediction quality (Herlocker, Konstan, Borchers, & Riedl, 1999). In our experiments, we vary the number of neighbors and compute the *MAE*. It can be observed from Figure 8 that the size of the neighborhood affects the quality of the prediction. When the number of neighbors is changed from 30 to 50 in our approach, the optimal *MAE* value is obtained.

4.3.5 Construction of user profiles

As described in Section 3.1, the user profiles are created based on movie attributes—genre, actor, actress, director and synopsis. Appropriate item attributes can contribute to more accurate user profiles, in turn improving the recommendation performance. In our experiment, we find the movie synopsis is the most effective attribute in terms of describing the user preference accurately, as illustrated in Figure 9.

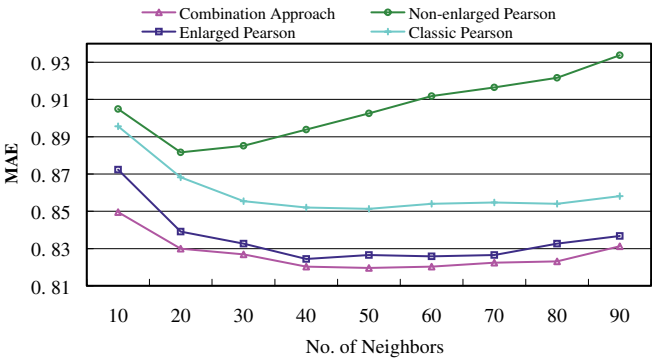

Fig. 8 Comparison of our approach and classic Pearson method

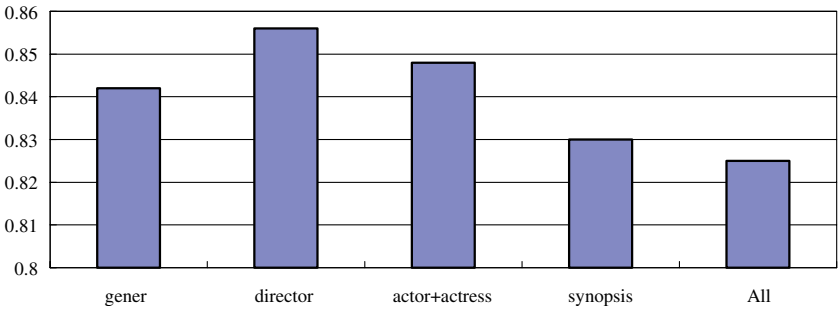

Fig. 9 Construction of user profiles

4.4 New user problem

To solve the new user problem, our approach requires that new users have manual profiles. For a movie recommender system, it is easier for new user to specify their favorite movie genre information than to make ratings on several movies. Once new users have created these simple manual profiles, our recommender system can make recommendations for them. In our experiment, first, all user profiles are constructed based only on the genre information by the method outlined in Section 3.1. We then randomly select one user ID 33302, and delete all of that user's ratings in the training set. Finally, we make predictions for the user by the average method described in Section 3.5. In the test data, the user has ratings for 21 items, which are described by the line "real value" in Figure 10. We can observe that the predictions for the new user partially reflect the user preference.

4.5 Our method versus other works

Although some hybrid recommender systems have already existed, it is difficult to objectively evaluate them. Some systems (Delgado et al., 1998) use Boolean values (relevant or irrelevant) to represent user preferences, while others such as our approach use numeric values. For instances, Ripper system (Basu & Cohen, 1998) works best when asked to make binary decisions, and Popescul et al. extended the aspect model to take item contents into account, which still based on binary

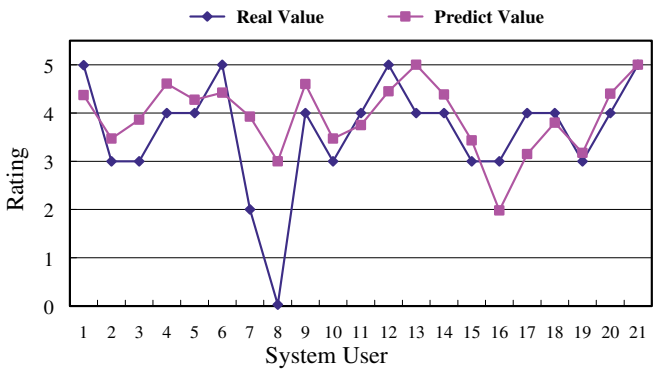

Fig. 10 New user problem

decisions made by its users. Both systems hence are unfair to compare it with our approach. Though a binary vote can stand for user preferences to some extent, human beings have more complex emotions. The five level rating systems such as our suggested system more closely reflect human psychology. The dynamic behaviors of systems also make a fair comparison difficult. In the P-Tango (Claypool et al., 1999) the weights of content-based filter and collaborative filter are changed with user feedbacks. In addition, Claypool does not clearly describe how to change the weights, and thus it is impossible for us to make a comparison with our approach.

However, we are able to make a simple concept comparison with the Fab system (Balabanovic & Shoham, 1997). In the Fab system, the similarity for prediction is only based on the user profiles. Clearly, this is a special case of our approach where the combination coefficient is 1. As shown in Figure 6, our approach shows better performance than the Fab system. Good et al. extended the filterbot concept (Sarwar et al., 2001)—rating robots participate as members of a CF system—in several ways. In particular, they demonstrate that a mixed CF solution that uses users and agents does indeed improve system performance. However, such variations of the filterbot do not solve the new user problem. Our approach, however, effectively addresses this issue by treating user clusters as items while Good treated agents as users.

5 Conclusions and future works

As we march into the age of digital information, the problem of data overload looms ominously ahead. Information filtering has subsequently been widely used to help us. CBF can directly select information based on a user's own profile contents without the opinions of other users, while CF can recommend information according to other opinions. A hybrid filtering method, which combines these two techniques, is a promising approach for information filtering.

In this paper, we investigate applying a clustering method to obtain information from user profiles for recommendation. Since item-based CF recommendation algorithms can further improve the performance of the recommendation, in our future work, we will apply the clustering method to group item contents instead of user profiles to achieve a better performance.

References

Anick, P. G., Brennan, J. D., Flynn, R. A., Hanssen, D. R., Alvey, B., & Robbins, J. M. (1990). A direct manipulation interface for boolean information retrieval via natural language query. In *Proc. of the ACM SIGIR-90* (pp. 135–150).

Baeza-Yates, R., & Riberio-Neto, B. (1999). *Modern information retrieval* (p. 30). Reading, MA: Addison-Wesley Publishing Co.

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

Basu, C., & Cohen, W. W. (1998). Using social and content-based information in recommendation. In *Proc. of the AAAI-98*.

Bradley, P. S., & Fayyad, U. M. (1998) Refining initial points for K-means clustering. In *Proc. of ICML '98* (pp. 91–99).

Breese, J. S., Heckerman, D., & Kardie, C. (1998). Empirical analysis of predictive algorithms for collaborative filtering. In *Proc. Of UAI* (pp. 43–52).
Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D., & Sartin, M. (1999). Combining content-based and collaborative filters in an online newspaper. In *ACM SIGIR '99 Workshop on Recommender Systems*.
Delgado, J., Ishii, N., & Ura, T. (1998). Content-based collaborative information filtering: actively learning to classify and recommend documents. In *Proc. of the CIA '98* (pp. 206–215).
Duda, R. O., Hart, P. E., & Stork, D. G. (2000) *Pattern classfication* (pp. 528–530). New York: Wiley-Interscience Publication.
Goldberg, D., Nichols, D., Oki, B. M., & Terry, D. (1992). Using collaborative filtering to weave an information tapestry. *Communications of the ACM Science*, *35*, 61–70.
Good, N., Schafer, J. B., Konstan, J. A., Borchers, A., Sarwar, B., Herlocker, J., & Riedl, J. (1999). Combininig collaborative filtering with personal agents for better recommendations. In *Proc. of the AAAI-99*.
Gupta, D., Digiovanni, M., Narita, H., & Goldberg, K. (1999). Jester 2.0: A new linear-time collaborative filtering algorithm applied to jokes. In *Proc. of ACM SIGIR '99 Workshop on Recommender Systems*.
Hauver, D. B., & French, J. C. (2001). Flycasting: Using collaborative filtering to generate a play list for online radio. In *Proc. of Web Delivery of Music*.
Herlocker, J., Konstan, J., Borchers, A., & Riedl, J. (1999). An algorithmic framework for performing collaborative filtering. In *Proc. of SIGIR-99*.
Kim, M., & Raghavan, V. V. (2000). Adaptive concept-based retrieval using a neural network. In *Proc. Of ACM SIGIR 2000 Workshop on Mathematical/Formal Methods in Information Retrieval*.
Lee, J. H., Kim, M. H., & Lee, Y. H. (1993). Ranking documents in thesaurus-based boolean retrieval systems. *Information Processing and Management*, *30*(1), 79–91.
MacQueen, J. B. (1967). Some methods for classification and analysis of multivariate observations. In *Proc. of the 5th Berkeley Symposium on Mathematical Statistics and Probability* (pp. 281–297).
Oard, D. W., & Marchionini, G. (1996). A conceptual framework for text filtering. *Technical Report EE-TR-96-25, CAR-TR-830*. University of Maryland.
Ogawa, Y., Morita, T., & Kobayashi, K. (1991). A fuzzy document retrieval system using the keyword connection matrix and a learning method. *Fuzzy Sets and Systems*, *39*, 163–179.
Popescul, A., Ungar, L. H., Pennock, D. M., & Lawrence, S. (2001). Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments. In *Proc. of UAI 2001*.
Resnick, P., Iacovou, N., Suchak, M., Bergstorm, P., & Riedl, J. (1994). GroupLens: An open architecture for collaborative filtering of netnews. In *Proc. of CCSCW* (pp. 175–186).
Robertson, S. E., & Sparck Jones, K. (1976). Relevance weighting of search terms. *Journal of the American Society for Information Science*, *27*, 129–146.
Salton, G., & Buckley, C. (1988). Term-weight approaches in automatic retrieval. *Information Processing and Management*, *24*(5), 513–523.
Salton, G., & McGill, M. J. (1983). *Introduction to modern information retrieval*. New York: McGraw-Hill.
Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In *Proc. of WWW* (pp. 285–295).
Terry, D. B. (1993). A tour through tapestry. In *Proc. of COOCS* (pp. 21–30).
Upendra, S., & Patti, M. (1995). Social information filtering: Algorithms for automating "word of mouth". In *Proc. of ACM CHI '95* (pp. 210–217).
Verhoeff, J., Goffman, W., & Belzer, J. (1961). Inefficiency of the use of the boolean functions for information retrieval systems. *Communications of the ACM*, *4*, 557–558, 594.
Wasfi, A. M. A. (1999). Collecting user access patterns for building user profiles and collaborative filtering. In *Proc. of IUI* (pp. 57–64).
Wilkinson, R., & Hingston, P. (1991). Using the cosine measure in a neural network for document retrieval. In *Proc. of ACM SIGIR* (pp. 202–210).
Wittenburg, K., Das, D., Hill, W., & Stead, L. (1995). Group asynchronous browsing on the world wide web. In *Proc. of the Fourth WWW* (pp. 51–62).