A Coupled Clustering Approach for Items Recommendation

Yonghong Yu[1], Can Wang[2], Yang Gao[1], Longbing Cao[2], and Xixi Chen[3]

[1] State Key Lab for Novel Software Technology, Nanjing University, P.R. China
[2] Advanced Analytics Institute, University of Technology Sydney, Australia
[3] Shandong Branch, Bank of Communications, P.R. China
{yuyh.nju,canwang613}@gmail.com, gaoy@nju.edu.cn, LongBing.Cao@uts.edu.au

Abstract. Recommender systems are very useful due to the huge volume of information available on the Web. It helps users alleviate the information overload problem by recommending users with the personalized information, products or services (called items). Collaborative filtering and content-based recommendation algorithms have been widely deployed in e-commerce web sites. However, they both suffer from the scalability problem. In addition, there are few suitable similarity measures for the content-based recommendation methods to compute the similarity between items. In this paper, we propose a hybrid recommendation algorithm by combing the content-based and collaborative filtering techniques as well as incorporating the coupled similarity. Our method firstly partitions items into several item groups by using a coupled version of k-modes clustering algorithm, where the similarity between items is measured by the *Coupled Object Similarity* considering coupling between items. The collaborative filtering technique is then used to produce the recommendations for active users. Experimental results show that our proposed hybrid recommendation algorithm effectively solves the scalability issue of recommender systems and provides a comparable recommendation quality when lacking most of the item features.

Keywords: Recommender systems, collaborative filtering, coupled object similarity, clustering algorithm.

1 Introduction

A recommender system is an important intelligent tool due to the huge volume of information on the Web. A recommender system overcomes the information overload problem by providing users with the personalized information, products or services (in this paper, we call them 'items'), which satisfy the user taste and preferences. In our daily life, typical applications of recommender systems including Amazon, Last.fm and MovieLens, recommend products, radios and movies, respectively. In addition, more and more e-commerce sites deploy recommender systems to attract public users, and therefore prompt their sale revenues.

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 365–376, 2013.

Collaborative Filtering (CF) [1] is one of the most widely used techniques for building recommender systems with a great success in e-commerce for its high recommendation quality. CF algorithms recommend items (e.g., products and movies) based on the opinions of other users that have the similar taste or preferences, rather than the content of items. However, CF algorithms suffer from the *scalability* problem [1].

Alternatively, content-based filtering (CBF) [2,3] make recommendations by analyzing the content of users or items. Balabanović et al. [2] and Melville et al. [3] have empirically shown that CBF techniques produce significant improvement against CF techniques in terms of the prediction quality. However, for CBF, it is hard to extract the reasonable features associated with items [1]. Moreover, CBF requires appropriate metrics to compute the similarity between items. But the existing metrics are not well-defined and not effective [4]. In addition, CBF also faces the serious scalability problem. The computational complexity exponentially rises when the number of users and items increase dramatically.

In order to overcome those challenges and solve the above issues, in this paper, we propose a hybrid recommendation algorithm by combining collaborative filtering with content-based filtering techniques. Our method firstly partitions items into several item groups by using a coupled version of the k-modes clustering algorithm, where the similarity between items is measured by the *Coupled Object Similarity (COS)* [4], which considers the coupling relation between items [5]. Then, CF is used to provide recommendations for active users. The key contributions are as follows: (1) we capture the correlation among items based on *COS*, which has been evaluated to outperform other similarity measures (e.g., *SMS* [6],*ADD* [7]) for categorical data. By this means, we overcome the similarity measure problem in content-based filtering; (2) we apply an effective clustering algorithm to group items, and compute the prediction within a small range of the item neighborhood. By applying the clustering algorithm, we solve the scalability problem in recommender systems; (3) we evaluate our proposed method on MovieLens data set in terms of the scalability and recommendation quality.

This paper is organized as follows. Section 2 briefly reviews the related work. The notations used in this paper are presented in Section 3. Section 4 describes the framework of our proposed hybrid recommendation algorithm combining the content-based with collaborative filtering techniques. The coupled similarity based k-modes clustering algorithm is proposed in Section 5. Experiments are evaluated in Section 6. Finally, we conclude this paper in Section 7.

2 Related Work

Generally, recommender systems can be classified into three classes [1]: collaborative filtering (CF) approaches, content-based filtering (CBF) approaches and hybrid approaches.

CF [8,9] approaches produce recommendations or predictions based on the assumption that similar users have similar tastes. The similarity between users is measured according to their history rating behaviors. CBF [2,3] approaches

recommend items for users by analyzing the content of items and the profiles of users. Typical CBF recommender system such as InforFinder [10]. However, pure content-based recommendation algorithms suffer from feature extraction problem [1].

Both CF and CBF approaches have limitations, since they make recommendations only relying on user-item matrix or features of users and items, respectively. Hybrid approaches by combining CF and CBF techniques help avoid certain limitations of CF with CBF approaches. For example, Balabanović et al.[2] proposed the recommender system Fab, which maintains user profiles based on content analysis, and then use CF techniques to find similar users for collaborative recommendation. Melville et al. [3] presented a content-boosted CF algorithm, which uses a content-based predictor to enhance existing user rating data and then makes prediction using a weighted person correlation-based CF algorithm. They are different from our proposed method, in which the similarity between items is measured through *COS* when a recommender system analyzes the content of items.

In addition, several clustering algorithms have been applied in recommender systems. Rashid et al. proposed *CLUSTKNN* [11], which uses a variant of basic k-means algorithm to partition users into clusters, and then leverages a CF algorithm to produce recommendations. Xue et al. proposed *CBSMOOTH* [12], which uses the clusters as the computed groups and smoothes the unrated data for individual users. Unlike our focus here, these algorithms group users or items over a user-item matrix, while ours groups items over the set of items. Furthermore, in order to group items, our hybrid recommendation algorithm adopts a coupled version of k-modes clustering algorithm, which outperforms other variants of k-modes clustering algorithms for categorical data sets.

3 Preliminaries

In a typical scenario, a recommender system consists of a set of n users $U = \{u_1, u_2, ..., u_n\}$, and a set of m items $O = \{o_1, o_2, ..., o_m\}$. Each user $u_i \in U$ expresses his/her preferences by rating a subset of items on a scale from one to five. This set of items rated by the user u_i is denoted as $O_{u_i}(O_{u_i} \subseteq O)$. Each item $o_j \in O$ is represented as a feature vector $o_j = \{a_{j_1}, a_{j_2}, ..., a_{j_l}\}$, and those features extracted from all the items are categorical. For example, if the item set O represents a collection of movies, then the features, i.e., *director, actor, genre* etc., are extracted to express a movie item. In addition, those features have categorical values, such as "*Koster*","*Grant*" and "*Comedy*" etc. for the feature genre.

Generally, user preferences on items are usually converted into a user-item matrix R, with n rows and m columns. Each element r_{ij} of R represents the rating given by user u_i on item o_j. The integer value of ratings falls into [0,5], in which 0 indicates that the user has not rated the item. The higher rating corresponds to the better satisfactory.

In essence, the objective of recommender systems is to predict the rating on the specified item o_j for an active user u_a, by leveraging all the various data mining and machine learning techniques.

4 Integrating Content-Based and CF Recommendation Algorithms

Our proposed approach is a hybrid recommendation algorithm by combining the content-based and collaborative filtering techniques. The framework of our proposed recommendation algorithm is presented in Fig.1. Our proposed recommendation algorithm consists of four major components: (1) Data Extraction: extract user-item matrix R and the set of items O from data source; (2) Item Neighborhood Formation: partition the set of items O into several clusters by applying a coupled version of k-modes clustering algorithm; (3) Model Building: compute the similarity between each pair of items in the same cluster and store these similarities in a model, namely a *HashMap*; (4) Prediction Computation: based on the trained model, use a nearest neighbor algorithm to produce recommendations for active users. We describe these four components in detail below.

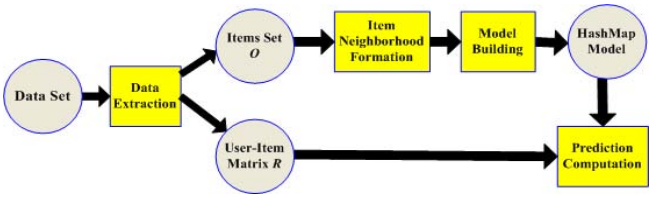

Fig. 1. The framework of our proposed hybrid recommendation algorithm

Data Extraction

In this component, our proposed hybrid recommendation algorithm extracts user-item matrix R and movie item set O from data source, which are used by movie item neighborhood formation and prediction computation component, respectively.

Item Neighborhood Formation and Model Building

1. Randomly select k distinct items from O.
2. Conduct the CK-modes algorithm on the set of items O, until the loss function converges or the number of iterations reaches the specified number of times. Once this CK-modes algorithm stops, the set of items O is divided into k disjoint clusters $\{O_1, O_2, ..., O_k\}(1 \leq i \leq k)$. Formally, $O = O_1 \cup O_2 \cup ... \cup O_k$, where $O_i \cap O_j = \emptyset(1 \leq i, j \leq k, i \neq j)$.

3. For each cluster, compute the similarity between each pair of items according to *COS*, and then store the pairs (e.g. $< itemid1, < itemid2, similarity >>$) in the *HashMap* model, where $itemid1$ is the key of the pair and $< itemid2, similarity >$ is the corresponding value. By using the map model, we quickly obtain the coupled similarity between $itemid1$ and $itemid2$.

Prediction Computation
Once the model with the coupled pairwise similarity has been built, the prediction method of the item-based collaborative filtering is adopted to generate the prediction on item o_i for an active user u. It takes the weighted sum of the ratings given by the active user u on the items similar to item o_i as the prediction. Weight measures the coupled similarity between the target item o_i and its similar item. Formally, the prediction P_{u,o_i} on item o_i for active user u is computed by the following formula.

$$P_{u,o_i} = \begin{cases} \frac{\sum_{\forall N_j \in N}(sim_{o_i,N_j} * R_{u,N_j})}{\sum_{\forall N_j \in N}(|sim_{o_i,N_j}|)} & \sum(|sim_{o_i,N_j}|) > 0 \\ \overline{r_u} & \sum(|sim_{o_i,N_j}|) = 0 \end{cases} \quad (1)$$

where N is the intersection of items rated by the active user u and items grouped by the CK-modes algorithm, R_{u,N_j} represents the rating on item N_j given by the user u. sim_{o_i,N_j} is the coupled similarity between item the o_i and the item N_j . $\overline{r_u}$ is the average of the active user's ratings.

5 Coupled Similarity Based K-Modes Algorithm

In this section, we present the coupled variant of k-modes clustering algorithm (CK-modes), which is used in our proposed recommendation algorithm to group items, by taking into account the coupling relationship among their features. In the recommender system, the features of items are categorical. For example, we use the categorical features (i.e. director, actor, genre and country) to represent a movie. K-modes [13], an extension of the basic k-means algorithm, is designed to deal with the categorical data sets. However, the similarity measure in k-modes is too rough to capture the closeness of two items. Hence, we decide to adapt the basic k-modes clustering algorithm to cluster items by incorporating the coupled similarity measure. The main difference between the basic k-modes algorithm and the adapted k-modes algorithm lies on the similarity measure and the method of updating modes. We further discuss these difference below.

5.1 Similarity Metric

For two items described by the categorical features, the k-modes clustering algorithm employs the Simple Matching Similarity [6] (*SMS*, which only uses 0 and 1 to distinguish similarities between distinct and identical categorical values) to compute the similarity feature values. However, *SMS* fails to capture the genuine

relationship between categorical feature values. In contrast, we adopt the Coupled Object Similarity (COS)$\in$ [0,1] to measure the similarity between items, which is more accurate than *SMS*. COS [4] considers both the intra-coupled similarity within a feature and the inter-coupled similarity between features, which has been evaluated to outperform other similarities (e.g. *SMS* [6],*ADD* [7]) in term of clustering quality.

Formally, the Coupled Object Similarity (COS) between categorical items X and Y is defined as follows.

$$COS(X,Y) = \sum_{j=1}^{n} \delta_j^A(X_j, Y_j), \tag{2}$$

where X_j and Y_j are the values of feature j for X and Y, respectively; and δ_j^A is Coupled Attribute Value Similarity($CAVS$).

The $CAVS$ consists of the *Intra-coupled Attribute Value Similarity (IaAVS)* measure $\delta_j^{Ia}(X_j, Y_j)$ and the *Inter-coupled Attribute Value Similarity (IeAVS)* measure $\delta_j^{Ie}(X_j, Y_j)$ for feature j. The definition of $CAVS$ between attribute values X_j and Y_j of feature j is as follows.

$$\delta_j^A(X_j, Y_j) = \delta_j^{Ia}(X_j, Y_j) \cdot \delta_j^{Ie}(X_j, Y_j) \tag{3}$$

$IaAVS$ measures the feature value similarity by considering the feature value occurrence frequencies within a feature, while $IeAVS$ measures the feature value similarity by taking the feature dependency aggregation into account [4].

5.2 Updating Modes

Let S be a cluster generated by the previous partition of k-modes algorithm. There are mm items described by categorical features $\{a_{j_1}, a_{j_2}, ..., a_{j_l}\}$ belonging to the cluster S. A mode of the cluster S is a item vector $Q = [q_1, q_2, ..., q_l]$ to maximize the sum of the similarity between each element of S and Q. The classic k-modes clustering algorithm updates the mode Q of cluster S by reassigning each component of Q with the corresponding feature value that occurs the most among all those values of the items in S.

In our proposed adapted k-modes algorithm, we update the mode of each cluster according to the following definition.

Definition 1. *The mode of item set S with mm items is a vector $Q = [q_1, q_2, ..., q_l]$ that maximizes:*

$$Sim(Q,S) = \sum_{i=1}^{mm} COS(S_i, Q) \tag{4}$$

Since the latter method needs to compute the similarity between each pair of items, it causes a high computation cost. The former method is more efficient than the latter one. However, in order to group items for recommender system,

we have to select the latter way. In recommender system, a movie item always has more than one genres. We extend the features of the movie item with t additional genre features, if the total number of genres of all the movies is t. In other words, a movie item is described by features $\{a_1, a_2, ..., a_l, g_1, g_2, ...g_t\}$. When a movie item has the genre $g_i (1 \leq i \leq t)$, we assign 1 to the corresponding value of the movie feature g_i, and otherwise 0. After the extension of features, the former mode updating method does not work well. The reason is that sparsity of genres: one movie has several genres, but the number of genres of all the movies is rather large, which leads to the phenomena that the occurrence frequency of 0 in feature g_i is much higher than that of 1. Hence, most of the corresponding mode values of $g_i (1 \leq i \leq l)$ are 0, resulting in the inaccurate description of the modes.

Therefore, we take advantage of the latter method formalized in Definition 1 in our proposed recommendation algorithm.

5.3 Coupled Similarity Based K-Modes

By using the coupled similarity measure and the method of modes updating described in the above sections 5.1 and 5.2, we design the Coupled similarity based k-modes clustering algorithm (CK-modes) described in Algorithm 1.

Algorithm 1. The Coupled Similarity Based K-modes Algorithm (CK-modes)

Input:

K : the number of clusters.
O : the set of m items.

Output:

a set of K clusters are generated when the loss function value converges.

1: Randomly select K initial modes from O
2: Allocate each item to the nearest mode, and the similarity between each item and the mode is measured by *COS*. At the same time, accumulate the loss function value when each item is assigned to a cluster.
3: Once each item is allocated to the corresponding cluster, update the modes of clusters according to the principle described in Section 5.2.
4: Reallocate every item against the current modes, and accumulate a new loss function.
5: Repeat (2) and (3) until the loss function value converges.

6 Experiments and Evaluation

In this section, we conduct several experiments to show the accuracy and the scalability of our proposed hybrid recommendation algorithm.

6.1 Data Set and Evaluation Metric

We use MovieLens data set in our evaluation, which has been widely used in collaborative filtering research in the last decade. MovieLens data set contains

100,000 ratings from 943 users and 1,682 movies, and users with less than 20 ratings have been removed. MovieLens data set was converted into a user-item matrix R with 943 rows (users) and 1682 columns (movies).

Since our proposed approach is a hybrid of collaborative filtering and content based recommendation algorithm, we extract features of movie from MovieLens data set, and present each movie item as a feature vector $o = (mid, director, actor, country, genre)$. In addition, MovieLens data set lacks of director, actor and country etc. features, we only retain genre in feature vector o. Particularly, a movie item has several genres, such as movie titled 'Toy Story' is both an animation and a comedy. In order to make use of these genres to group movie, we extend the features of movie item with t additional genre features, if the total number of genres of all the movies in data set is t. In other words, a movie item is described by features $\{g_1, g_2, ...g_t\}$. When a movie item has genre $g_i (1 \leq i \leq t)$, we assign 1 to the corresponding feature value g_i, and otherwise 0.

We choose MAE to evaluate experimental results, as MAE metric is simple to calculate and intuitive to interpret. Formally,

$$MAE = \frac{\sum_{i=1}^{N} |p_i - q_i|}{N} \tag{5}$$

where p_i and q_i are the real rating and the corresponding prediction, respectively, and N denotes the total number of predictions generated for all active users. The lower the MAE, the better the recommendation algorithm generates the predictions for users.

6.2 Experimental Settings

Benchmark Recommendation Algorithms. We evaluate several widely discussed algorithms in the recommender system research, including user-based collaborative filtering algorithm [8], item-based collaborative filtering algorithm [9] and *CLUSTKNN* [11]. Each benchmark recommendation algorithm has been tuned to produce the best prediction quality according to the principles described in the corresponding literature.

We conduct a five-fold cross validation over the MovieLens data set by randomly extracting different training and test sets each time, which accounts for 80% and 20%, respectively. Finally, we use the average of MAE and the run time costs over the five folds to present the experimental results.

6.3 Experimental Results

In this section, we firstly determine the sensitivity of some parameters of our proposed hybrid recommendation algorithm and then compare with other benchmark recommendation algorithms. The parameters in our proposed algorithm include the number of clusters K and the number of neighbors that are selected to compute predictions for the target movie items.

Sensitivity of the Number of Clusters K. We perform a group of experiments to evaluate the prediction quality on the number of clusters K, ranging from 5 to 150 with a step of 10. Fig.2 reports the results. We observe that the number of clusters K does have an impact on the prediction quality. As K increases from 5 to 20, the prediction quality downgrades. After that, the MAE fluctuates; and then the curve tends to be flat. The best prediction quality is $MAE = 0.73$, when the number of clusters K equals to 5. Thus, we select $K = 5$ as the optimal choice in our following experiments.

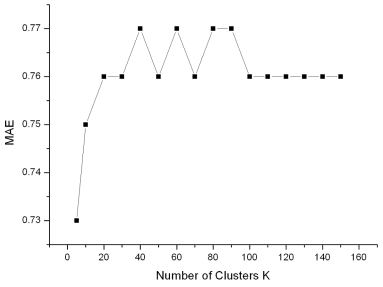

Fig. 2. Impact of K on our proposed recommendation algorithm

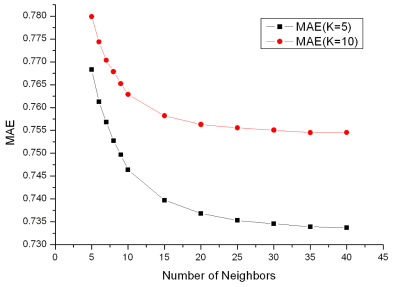

Fig. 3. Impact of the number of neighbors on our proposed recommendation algorithm

Sensitivity of the Neighborhood Size. We conduct another group of experiments to assess the prediction quality on the size of neighborhood used to produce the recommendation, ranging from 5 to 40 with a step of 5. Fig.3 describes that $MAEs$ decrease as the number of neighbors increases from 5 to 30, and then $MAEs$ remain stable. That is to say, our proposed method achieves a better prediction quality when more similar items are taken into account. Therefore, we select 30 as the optimum item neighborhood size.

From Fig.3, we observe that the more neighbors involved in making recommendation, the better the recommendation quality. There are two reasons: (1) the average size of neighborhoods decreases as the number of clusters K increases; (2) the intersection between the item neighborhood generated by the CK-modes algorithm and the set of items rated by the active user becomes smaller as the value of K increases. As a result, our proposed recommendation algorithm takes average of the active user's ratings as the prediction value when there are no neighbors for the target item.

Prediction Quality Comparisons. Once we determine the optimal parameters, we compare our proposed algorithm with those selected benchmark recommendation algorithms in terms of the prediction quality.

Table 1 presents the results of the best prediction quality by using different recommendation algorithms. It can be observed from Table 1 that the item-

based algorithm outperforms other recommendation algorithms, and followed by *CLUSTKNN* recommendation algorithm. The prediction qualities of the user-based and our proposed hybrid recommendation algorithm are comparable to the item-based recommendation algorithms (i.e., MAE =0.73). However, The difference between our proposed algorithm and the item-based algorithm is small and is not statistically significant. In fact, our proposed hybrid recommendation algorithm depends on the features of items, and does not work not well when most of the important features are missing. Only the genres of movie has been extracted from the MovieLens data set, and other features, such as director, actor and country, are missing. Thus, the lacking of information limits of our proposed hybrid recommendation algorithm to some extent.

Table 1. Comparison of prediction quality of recommendation algorithms

Recommendation algorithm	**MAE**
User-based CF	0.730
Item-based CF	0.72
CLUSTKNN	0.725
Our proposed algorithm	0.730

Performance. Here, we focus on the overall performance of the system. We denote the throughput as the number of recommendations generated per second. Fig.4 shows the throughputs of both our proposed recommendation algorithm and other benchmark recommendation algorithms. Note that the user-based recommendation algorithm scans the whole user-item matrix R, its throughput do not change with the number of clusters. However, the throughput of the item-based recommendation algorithm varies with the number of neighbors selected to produce predictions. We plot the throughput of the item-based recommendation algorithm when the number of neighbors is 30, where it generates the best prediction quality.

As we can clearly see from Fig.4, clustering based recommendation algorithms (i.e., *CLUSTKNN* and our proposed hybrid recommendation algorithm) outperform both the user-bused and the item-based recommendation algorithms. The throughputs of both the *CLUSTKNN* and our proposed hybrid algorithm are substantially higher than other approaches at all values of the number of clusters. We can observe that for the number of clusters $K = 20$, our proposed recommendation algorithm produces a throughput rate of 12492 while the user-based and the item-base recommendation algorithms produce only 1333 and 2564, respectively. In addition, increasing the number of clusters corresponds to scanning the decreasing movie item neighborhood. Fig.4 also shows that the throughput of our proposed method increases rapidly as the number of clusters goes up. By contrast, the throughput of *CLUSTKNN* drops down as the number of clusters grows. The reason is that the *CLUSTKNN* takes centroids of clusters as the neighbors of the active user. Then, it takes more neighbors into account as the number of clusters increases.

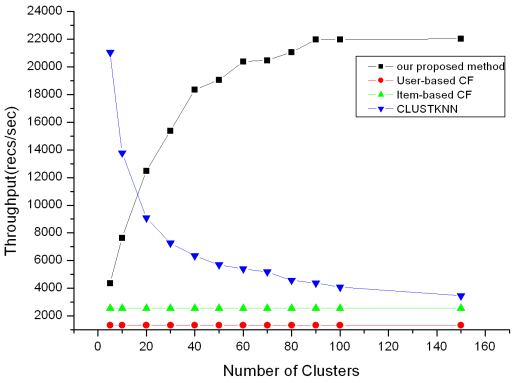

Fig. 4. Throughput of the selected recommendation algorithms

We make the following conclusions from the above experimental evaluation. First, the CK-modes algorithm is applied to divide the set of items O into several clusters, and then the rating predictions are computed within a small neighborhood. Hence, our proposed recommendation algorithm is highly scalable, which gives a quick response for active users. Second, the prediction quality of the hybrid recommendation algorithm is comparable with other classic recommendation algorithms. Only a minor difference on the prediction quality is observed. Finally, only the genres of movie are available in the MovieLens data set. Lacking of features information limits the accuracy of our hybrid recommendation algorithm. We regard that this hybrid recommendation algorithm will produce a better prediction quality when more features of movie item are available. As a matter of fact, finding similar movie items by using the movies' content is more reasonable and intuitive than by using the users' rating behaviors.

7 Conclusion and Future Work

Recommender systems play an important role in e-commerce for both users and businesses. It provides personalized recommendations for better business revenue. In this paper, we propose a hybrid recommendation algorithm by combining the content-based and the collaborative filtering techniques. This hybrid recommendation algorithm firstly partitions the items into several groups by using the coupled version of k-modes clustering algorithm. Then it uses the collaborative filtering technique to produce the recommendations for active users. Experimental results show that our proposed hybrid recommendation algorithm effectively solves the scalability issue of recommender systems with a comparable recommendation quality under the condition of lacking of most of the features.

We plan to extract more features of items to improve our proposed hybrid recommendation algorithm. We will also extend other recent clustering algorithms,

such as spectral clustering algorithm, to speed up the process of model building and improve the prediction quality.

Acknowledgments. We would like to acknowledge the support for this work from the National Science Foundation of China (61035003, 61175042,61021062), the National 973 Program of China (2009CB320702), the 973 Program of Jiangsu, China (BK2011005), the Program for New Century Excellent Talents in University (NCET-10-0476), the Australian Research Council Discovery Grant (DP1096218, DP130102691) and ARC Linkage Grant (LP100200774).

References

1. Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Transactions on Knowledge and Data Engineering 17(6), 734–749 (2005)
2. Balabanović, M., Shoham, Y.: Fab: content-based, collaborative recommendation. Communications of the ACM 40(3), 66–72 (1997)
3. Melville, P., Mooney, R., Nagarajan, R.: Content-boosted collaborative filtering for improved recommendations. In: Proceedings of the National Conference on Artificial Intelligence, pp. 187–192. AAAI Press, MIT Press (1999, 2002)
4. Wang, C., Cao, L., Wang, M., Li, J., Wei, W., Ou, Y.: Coupled nominal similarity in unsupervised learning. In: CIKM, pp. 973–978. ACM (2011)
5. Cao, L., Ou, Y., Yu, P.: Coupled behavior analysis with applications. IEEE Transactions on Knowledge and Data Engineering 24(8), 1378–1392 (2012)
6. Gan, G., Ma, C., Wu, J.: Data clustering: theory, algorithms & applications (asasiam series on statistics & applied probability, n 20). Recherche 67, 02 (2007)
7. Ahmad, A., Dey, L.: A k-mean clustering algorithm for mixed numeric and categorical data. Data & Knowledge Engineering 63(2), 503–527 (2007)
8. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J.: Grouplens: an open architecture for collaborative filtering of netnews. In: CSCW, pp. 175–186. ACM (1994)
9. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Item-based collaborative filtering recommendation algorithms. In: WWW, pp. 285–295. ACM (2001)
10. Krulwich, B., Burkey, C.: Learning user information interests through extraction of semantically significant phrases. In: Proceedings of the AAAI Spring Symposium on Machine Learning in Information Access, pp. 100–112 (1996)
11. Al Mamunur Rashid, S., Karypis, G., Riedl, J.: Clustknn: a highly scalable hybrid model-& memory-based cf algorithm. In: Proc. of WebKDD 2006, Citeseer (2006)
12. Xue, G., Lin, C., Yang, Q., Xi, W., Zeng, H., Yu, Y., Chen, Z.: Scalable collaborative filtering using cluster-based smoothing. In: SIGIR, pp. 114–121. ACM (2005)
13. Huang, Z.: Extensions to the k-means algorithm for clustering large data sets with categorical values. Data Mining and Knowledge Discovery 2(3), 283–304 (1998)