Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 60 (2015) 1453 – 1461

Procedia Computer Science

19th International Conference on Knowledge Based and Intelligent Information and Engineering Systems

Hybrid User-Item Based Collaborative Filtering

Nitin Pradeep Kumar, Zhenzhen Fan*

Institute of Systems Science, National University of Singapore
25 Heng Mui Keng Terrace, Singapore- 119615

Abstract

Collaborative filtering (CF) is widely used in recommendation systems. Traditional collaborative filtering (CF) algorithms face two major challenges: data sparsity and scalability. In this study, we propose a hybrid method based on item based CF trying to achieve a more personalized product recommendation for a user while addressing some of these challenges. Case Based Reasoning (CBR) combined with average filling is used to handle the sparsity of data set, while Self-Organizing Map (SOM) optimized with Genetic Algorithm (GA) performs user clustering in large datasets to reduce the scope for item-based CF. The proposed method shows encouraging results when evaluated and compared with the traditional item based CF algorithm.

Keywords: Item based collaborative Filtering, SOM, GA, CBR

1. Introduction

Understanding the online customer's needs and expectations is considered important for the current day consumer-oriented electronic commerce market. Recommendation systems[1] are widely used in the internet to provide personalized recommendations to users. One of the most promising techniques is Collaborative Filtering[2] (CF), which is based on the assumption that users with similar taste have similar preference to products or items. For a user of concern, CF predicts what product or item the user may like by considering the opinions from other users. Many online

* Corresponding author. Tel.: +65-6516-6928; fax: +65-6778-2571.
E-mail address: zhenzhen@nus.edu.sg

shopping, news and social networking sites make use of such algorithm to recommend movies, books, articles, or other items to their users.

There are mainly two types of CF algorithms discussed in literature – user based and item based CF[3]. User based CF algorithms look for users that share similar preference patterns (in terms of ratings of items) with the user of concern and recommend items that are rated high by these similar users. Item based CF algorithms[4] in contrast consider the similarity between items instead of users. They recommend to the user items that are similar to other items that the user has rated high.

As stated in Sarwar et al. [4], one of the main challenges faced by user based CF is the problem of scalability as it can be slow when searching for nearest neighbors in very large datasets with millions of users and products. The other main challenge is the issue of data sparsity as even active users only have ratings for a very small percentage of items considering the large amount of items available in recommender systems. Sparsity makes it difficult to determine the similarity between users and thus affect the quality of recommendation. Since item similarity changes less frequently than those between users, the similarities between items can be pre-computed and kept in memory instead of computing it during runtime. Therefore, at the cost of additional overhead of maintaining the item similarity matrix, item based CF scales better than user based CF in terms of online performance when dealing with large database. However, data sparsity still remains a big issue affecting the item similarity computation and as a result affecting the recommendation quality.

In this paper we propose a hybrid user-item based CF method to achieve a more personalized product recommendation for a user while addressing the traditional issues of data sparsity and scalability in collaborative filtering algorithms. The remainder of the paper is organized as follows. Section 2 presents an overview of the related work in addressing the challenges in CF. Section 3 discusses the framework and steps of computation for the proposed algorithm which includes sparsity removal using Case Based Reasoning (CBR) and average filling, and user clustering using Self-Organizing Map (SOM) optimized with Genetic Algorithms (GA). Section 4 presents experimental evaluations of our method using the MovieLens dataset[13] and a comparison of results with the traditional item-based collaborative filtering algorithm. Section 5 finally draws conclusions.

2. Related Work

Given the issues of scalability and sparsity faced by traditional collaborative filtering algorithms, there have been various approaches proposed to address these issues. For scalability problem, the main idea is to reduce the size of data to be considered during recommendation. Sarwar et al. [5] has applied clustering to group users with similar rating patterns into clusters, so as to reduce the scope of neighborhood to be considered during recommendation. It's shown to significantly improve online performance while giving comparable prediction accuracy with the traditional algorithm. Gong and Ye[6] in their work have tried to join user clustering with item based CF. They apply the K-Means algorithm on the user-item matrix of ratings to obtain clusters as neighborhoods. For a target user X, the closest cluster is identified and item based CF prediction is then applied within the cluster. SOM clustering has also been experimented as a pre-processing step[7] and thereby the similarity and weighted averaging methods only need to be applied on the users in a given cluster.

For the issue of sparsity of the User-Item matrix, efforts focus on how to reduce the sparsity by filling in the vacant cells in the User-Item matrix with likely rating values before the matrix is used for further processing. Xia et al[8] use a simple technique called average filling, assigning the unrated items of a user with the average value of all his other ratings, and then use the modified data for item based collaborative filtering. Gong's work[9] employs CBR techniques to fill in vacant ratings. For an active user case, the most similar user cases are retrieved from the case base using Euclidean distance and the missing ratings are estimated based on the ratings from these similar users.

There are several optimization techniques proposed to improve the results of collaborative filtering. Kim and Ahn[10] have proposed a hybrid method combining K-Means clustering with GA optimization. K-Means clustering has the drawback that the clustering results can be sensitive to the initial seed used to partition the dataset. GA is therefore used to select optimal or suboptimal seeds for K-Means clustering. The fitness function for GA is the performance of the clustering algorithm, measured using intra-class inertia which is the average of the distances between the mean and the observations in each cluster. After GA-K Means clustering, the cluster for a target user is identified and the nearest neighbors are found from that cluster.

Other techniques have also been proposed to improve the recommendation quality of item based CF. Item based CF using UserRank[11] tries to use a modification of PageRank algorithm to rank or weight users in the User-Item matrix based on their importance. The weights of the users are then incorporated into the computation of item similarities and item based CF recommendations. Zhang et al[12] have explored three new item similarity measures, essentially weighting the original correlation-based similarity formula with a ratio of the number of users that rate both items *i* and *j* (*N*) and the number of users that rate item *i* or *j* (*M*). They have concluded that a simple multiplication of (*N/M*) with the original similarity measure gives the best result. Item based hybrid similarity[13] instead brings in the values of item attribute similarity to adjust the predicted rating of the targeted item. Deviation Adjustment[8] is another technique that has been suggested to minimize the error between the actual and predicted user ratings. The CF algorithm's prediction is modified based on user deviation adjustment and item deviation adjustment which are measured from the algorithm's error in predicting the training data.

3. Hybrid User-Item Based Collaborative Filtering

3.1. Overview

In striving for a more personalized product recommendation for a user, we propose a hybrid user-item based CF algorithm leveraging on user similarity as well as item similarity, trying to address both the data sparsity and scalability issues of traditional CF algorithms.

To address data sparsity, the key is how well the likely values of the vacant cells are estimated. We think the overall assumption of CF that similar users may have similar ratings still applies here. Therefore we use CBR and average filling for sparsity reduction of the user-item matrix. This step will help the subsequent user similarity and item similarity computation as both of them don't work well with sparse data.

We agree with other researchers[4, 6, 7] that scalability can be addressed by pre-grouping users sharing common attributes into clusters so that only a smaller subset of users needs to be considered during recommendation. Given a dense matrix already processed by CBR and average filling, we continue to cluster similar users using Self-Organising Map (SOM) network. Genetic algorithms (GA) is used here as an optimization step to compliment SOM to obtain sub-optimal clusters with more balanced number of users in each cluster.

Both sparsity reduction and SOM-GA clustering are performed as pre-processing steps before the actual recommendation starts. At the time of recommendation, for a target user, the closest cluster is first identified for the user, then the traditional item based CF is performed within the respective cluster.

The overall framework of our method is illustrated in Figure 1.

3.2. Sparsity Removal

To fill the vacant cells in the matrix, we employ CBR followed by average filling. Since we assume that similar users have similar ratings for the same items, CBR is an intuitive method to be used here with its assumption that similar cases require similar solution. The ratings given by individual users form the cases and the similarity between two users is determined by how similar their rating patterns are. The similarity measure used here is Euclidean distance, given by Eq. 1,

$$sim(X,Y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)(x_i - y_i)} \tag{1}$$

where x_i and y_i are the value of the *i*th feature of the input case and that of the existing case respectively. Only the items rated by both users are used for similarity calculation. For a target user, a sorted list of top *K* similar users is returned based on the similarity scores between this user and other users.

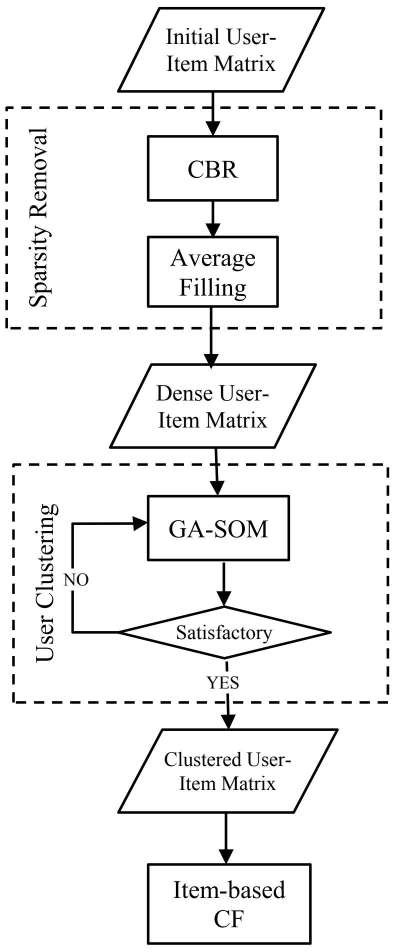

Fig. 1. Framework of Hybrid User-Item Collaborative Filtering algorithm

These retrieved cases are then used to fill in the vacant cells of the target user. For each unrated item, the rating is estimated based on the ratings of this item by these similar users. The estimated rating is the cumulative sum of the rating by each similar user weighted by the similarity score between this similar user and the target user, divided by the sum of similarity scores of the similar users involved. The estimation of the rating of a target user *u* for an unrated item *t* is given by Eq. 2,

$$F_{ut} = \frac{\sum_{i=1}^{k}(R_{it} \times sim(i,u))}{\sum_{i=1}^{k} sim(i,u)} \tag{2}$$

where R_{it} is the rating for item *t* by user *i*; *sim(i,u)* is the similarity between user *i* and user *u;* and *k* is the number of similar users under consideration.

Since it's possible that none of the similar users has rating for the target item, unrated items can still remain in the user-item matrix after filling in the matrix using CBR. These vacant cells are then handled using average filling. The estimated rating of a target user *u* for an unrated item *t* using average filling is given by Eq. 3,

$$A_{ut} = \frac{\sum_{i=1}^{k} R_{ui}}{k} \tag{3}$$

where R_{ui} is the rating for item i by user u, and k is the number of other items currently rated by user u.

3.3. User Clustering Using GA-SOM

After sparsity reduction, we have a dense user-item matrix. We then cluster users based on user rating similarity using SOM.

SOM is a neural network model for unsupervised learning, consisting of two layers, the upper output layer and the inner input layer. The number of input variables determines the number of input nodes. The individual ratings given by users form the basis for clustering in SOM and form the input variables. Every node in input layer is connected with every neuron in the output layer. The output neurons are activated when the Euclidean distance between input vectors and weights which link nodes with those in the output layer is minimum. Also, the coefficients of connection weight are amended and become more contiguous with the input vectors until the termination limitations are satisfied[14].

The type and number of clusters formed using SOM is dependent on input output mapping, shape of neighborhood, neighborhood functions, etc. When using SOM we are unable to predict the number of clusters that will be formed. The number of actual users per cluster after clustering can also be highly skewed.

In this work, we try to avoid such a cluster formation by complimenting SOM with GA optimization. GAs are stochastic search techniques inspired by natural genetics in biology, evolving solutions with iterations of selection, crossover, and mutation. We use GA to select a clustering result with clusters of similar sizes, by varying SOM parameters until a satisfactory fitness score is obtained.

The chromosome of GA is designed as an 11 bit binary string to represent SOM parameters such as X axis and Y axis length of the map, learning rate, and the neighborhood function. 2 bits are used to represent the neighborhood function whereas 3 bits each are used to represent other parameters respectively, as illustrated in Fig. 2. A combination of binary bits is used to represent integer values, while floating points are represented following the method described in Kim and Ahn[10] using Eq. 4,

$$x/(2^x - 1) \tag{4}$$

where, x is the decimal value of the bits combined. For example, the decimal value of binary number 101 will be 5, for which Eq. 4 yeilds a value of 0.161.

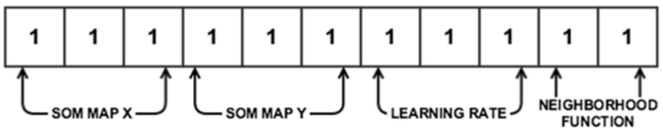

Fig. 2. Chromosome representation for GA.

The fitness value for GA reflects the deviation of the size of each cluster from the average size of all clusters. It's a value to be minimized by GA to avoid clustering results with very skewed clusters. The fitness function is given by Eq. 5,

$$F = \sum_{i=1}^{k} |u_i - avg(u)| \tag{5}$$

where, k is the number of clusters; u_i is the number of users in the ith cluster, and *avg(u)* is the average size of all clusters. Thus, the number of users in each cluster will be approaching the average number of users across all clusters as GA progresses. We also include a soft penalty for the maximum number of clusters that can be formed.

3.4. Recommendation Using In-Cluster Item-Based CF

After the above two preprocessing steps, we prepare for item based CF by pre-computing an item-item similarity matrix for each cluster based on Euclidean similarity. As indicated in Sarwar et al. [4], it's not necessary to retain the similarity values between an item and all other items, since to compute prediction, we only need a small number of similar items for the item of concern. Therefore in each cluster, for each item, we compute and store the similarity scores for 50 items most similar to it.

The actual recommendation is performed as follows. Given an individual user, we first identify the cluster that is closest to the user based on Euclidean distance between the user and cluster centers. Then, following the method described in Sarwar et al. [4], for an unrated item *t* for this user, its top most similar items are identified from the item-item similarity matrix for this cluster. User *u*'s ratings for these items are gathered to derive the predicted rating for *t.* The prediction of user *u*'s rating for *t*, P_{ut}, is defined using the weighted sum formula given in Eq. 6,

$$P_{ut} = \frac{\sum_{i=1}^{k}(R_{ui} \times sim(i,t))}{\sum_{i=1}^{k} sim(i,t)} \tag{6}$$

where R_{ui} is the rating for item *t* by user u; *sim(i,t)* is the similarity between item *i* and item *u;* and *k* is the number of most similar items under consideration.

Based on the predictions, the top items predicted with high ratings will then be recommended to the user.

4. Experimental Evaluation and Results

4.1. Dataset

The Movie Lens data set is used for evaluating the proposed algorithm. This dataset was collected by the GroupLens Research Project[15] at the University of Minnesota. There are about 100,000 ratings provided by 1000 users on 1680 movies with each user having rated at least 20 movies.

4.2. Evaluation Metrics

Recommendation systems are mainly evaluated by either statistical accuracy metrics[16] or decision support metrics[17]. In statistical accuracy metrics, the accuracy of the prediction algorithm is calculated by comparing the numerical deviation of the predicted rating with the actual user rating. For measuring the accuracy of the proposed method, the statistical accuracy measure, Mean Absolute Error (MAE) is used. Formally, MAE is given by Eq. 7,

$$MAE = \frac{\sum_{i=1}^{n}|p_i - q_i|}{n} \tag{7}$$

where, p_i and q_i are the predicted and actual rating for *i*th item, and *n* is the number of items under consideration.

4.3. Experiment Design

Cross validation is used to evaluate our results. In *k*-fold validation, the original dataset is partitioned into *k* equal sized subsets. Of the *k* subsets, one single subset is used as the validation data, and the rest *k*-1 subsets is used for training. This process is iterated for *k* times, each time selecting a different subset as the validation data. We use k=5, which means 20 percent of the data is used for validation and the rest 80 percent is used as the training data.

In each iteration, our experiment is designed in this way:

1) One subset (20%) of users is reserved as testing data. 10 ratings from each of these users are randomly removed. (Our algorithm tries to predict these removed ratings and the evaluation measures by how much our predictions deviate from the actual ratings.)
2) The rest 80% of the data goes through sparsity removal and GA-SOM clustering.
   For GA, the soft penalty limits the minimum and maximum number of clusters to be formed at 4 and 10 respectively. A population size of 100 is used. The stopping criteria is the maximum number of generations, which is set to be the same as the population size. The mutation and crossover rates are 0.1 and 0.6 respectively.
3) In each cluster, the item-item similarity matrix is pre-computed.
4) For each user in the validation set, the closest cluster is identified. Then the removed ratings are predicted using in-cluster item based CF.
5) MAE for the predicted ratings and the actual ratings is calculated.

It's worth noting that when using top most similar items to predict for an unrated item, it's likely that none of these items have been rated by the current user. In this case, Eq. 7 will return no ratings. Although it's possible that a default strategy could be adopted (such as using a most neutral value 3, or the average of current user's other ratings) so that the algorithm still returns some ratings, we decide to treat unpredicted ratings and predicted ratings separately in MAE computation so as to better reflect the algorithm's ability to produce ratings and the quality of the predictions:

- To capture our algorithm's ability to produce prediction for the removed ratings, we report Prediction Sensitivity, the percentage of ratings that it can predict, and compare it with that of traditional item based CF.
- Overall MAE is computed over the actually predicted ratings, and compared with that of traditional item based CF.
- We also calculate MAE for items that can be rated by both proposed and traditional algorithm. These are typically the cases that can be predicted quite well by item based CF algorithms. We want to find out if our proposed method has made improvement in such cases.

4.4. Experiment Results

Prediction Sensitivity: The prediction sensitivity of the proposed and traditional methods are summarized in Table 1. The table shows the percentage of predicted ratings for each of the 5 validation sets. The results show that the proposed method can produce predictions in cases where the traditional item-based CF fails to return ratings.

Table 1. Prediction sensitivity for each of the 5 validation set.

	Prediction Quality Percentage				
	1	*2*	*3*	*4*	*5*
Traditional Item-based CF	10.58	11.06	14.25	7.92	*8.32*
Proposed algorithm	51.01	53.24	56.70	52.55	52.19

Overall MAE: To evaluate the quality of prediction for items that can be rated, we calculate the overall MAE for each of 5-fold validation. The results are shown in Fig. 3. The overall MAE for the proposed algorithm is found to be lower compared with the traditional item-based CF method for each of 5 validation sets.

Overall MAE for items predicted by both algorithms: For the group of items which could be predicted by both the traditional and proposed algorithms, the MAE of the ratings are calculated respectively. The results are shown in Table 2. It's observed that for such cases, traditional algorithm performs quite well and our method has achieved further improvement.

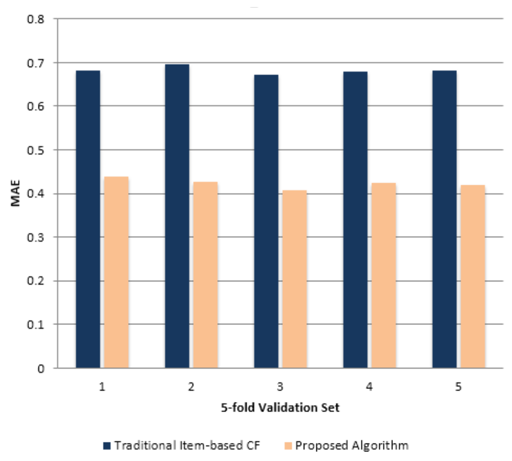

Fig. 3. Overall MAE for each of the K-fold validation set.

Table 2. Overall MAE of items predicted by both the proposed and traditional item-based CF algorithms for each validation set.

	Overall MAE				
	1	*2*	*3*	*4*	*5*
Traditional Item-based CF	0.25	0.20	0.20	0.23	*0.22*
Proposed Algorithm	0.17	0.16	0.16	0.17	0.15

5. Conclusions

Collaborative Filtering (CF) is widely used in the internet for making recommendations. Data sparsity and scalability are the two main drawbacks of traditional CF algorithms. In this paper we have presented a hybrid user-item collaborative filtering method to address these issues in two phases, aiming to produce more personalized recommendations with better rating quality. In the first phase, the data sparsity is reduced using CBR followed by average filling. In the second phase, to address scalability issues, the dense matrix is clustered into groups of similar users using SOM optimized with GA. More personalized recommendations are then performed at cluster level using the traditional item-based CF. The results are compared with those of the traditional item based CF algorithm. Our initial experiment gives encouraging results, with the proposed method showing better prediction sensitivity and better prediction quality than the traditional item-based CF algorithm. For future work, more experiments will be done on other data sets to further validate our method, and other similarity functions such as cosine similarity will be explored as well.

References

1. J. Ji, Z. Sha, C. Liu,N. Zhong. Online Recommendation Based on Customer Shopping Model in E-Commerce. *Proceedings of the IEEE/WIC International Conference on Web Intelligence*. 2003. P. 68-74.
2. J. S. Lee, C.H. Jun, J. Lee, S.Y. Kim. Classification-based Collaborative Filtering using Market Basket data. *Expert Systems with Applications*. 2005:29:700–704.
3. T. Segaran. Chapter 2. Making Recommendations. *Programming Collective Intelligence*. 2007. 7-28.
4. Sarwar, Badrul, et al. Item-based Collaborative Filtering Recommendation Algorithms. *Proceedings of the 10th International Conference on World Wide Web*. ACM, 2001.
5. Sarwar, Badrul M., et al. Recommender Systems for Large-Scale E-Commerce: Scalable Neighborhood Formation using Clustering. *Proceedings of the Fifth International Conference on Computer and Information Technology*. Vol. 1. 2002.

6. S.J. Gong, H.W. Ye. Joining User Clustering and Item Based Collaborative Filtering in Personalized Recommendation Services. *Proceedings of International Conference on Industrial and Information Systems*. 2009. 149-151.
7. S.J. Gong, H.W. Ye, X.M. Zhu. Item-Based Collaborative Filtering Recommendation using Self-Organizing Map. *Proceedings of Chinese Control and Decision Conference*. 2009. 4029-4031.
8. S. Xia, Y. Zhao, Y. Zhang, C.X. Xin, S. Roepnack, S.H. Huang. Optimizations for Item-based Collaborative Filtering Algorithm. *Proceedings of Natural Language Processing and Knowledge Engineering (NLP-KE)*. 2010. 1-5.
9. S.J. Gong. Joining Case-based Reasoning and Item-based Collaborative Filtering in Recommender Systems. *Proceedings of the Second International Symposium on Electronic Commerce and Security*. 2009. 40-42.
10. K. Kim, H. Ahn. A Recommender System using GA K-Means Clustering in an Online Shopping Market. *Expert Systems with Applications*. 2008. 1200–1209.
11. M. Gao, Z.F. Wu, F. Jiang. Userrank for Item-based Collaborative Filtering Recommendation. *Information Processing Letters*. 2011. 440-446.
12. J. Zhang, Z. Lin, B. Xiao, C. Zhang. An Optimized Item-Based Collaborative Filtering Recommendation Algorithm. *The Proceedings of of IC-NIDC*. 2009. 414-418.
13. S. Puntheeranurak, T. Chaiwitooanukool. An Item-based Collaborative Filtering Method using Item-based Hybrid Similarity. *Proceedings of Software Engineering and Service Science (ICSESS), IEEE 2nd International conference*. 2011. 469-472.
14. S.H. Zhou, L. Fu, B.L. Liang. Clustering Analysis of Ancient Celadon based on SOM Neural Network. *Science in China Series E: Technological Sciences*. 2008:51(7):999-1007.
15. GroupLens Research Project, Movie Lens Data Set http://www.grouplens.org/
16. Q.H. Huang, W.M. Ouyang. Fuzzy Collaborative Filtering with Multiple Agents. *Journal of Shanghai University (English Edition)*. 2007:11(3):290-295.
17. H. N. Kim, A. T. Ji, G.S. Jo. Enhanced Prediction Algorithm for Item-Based Collaborative Filtering Recommendation. *EC-Web 2006*. 2006. pp A l - 50.