A Collaborative Filtering Recommendation Algorithm Based on Category and Influence of Current Items

Xiaoxi Chen[1] Zhong Yao[1]

[1] School of Economics and Management, BeiHang University, Beijing 100191, China

Abstract

The traditional user-based collaborative filtering (CF) algorithms often suffer from two important problems: Scalability and sparsity because of its memory-based k nearest neighbor query algorithm. Item-Based CF algorithms have been designed to deal with the scalability problems associated with user-based CF approaches without sacrificing recommendation or prediction accuracy. However, item-based CF algorithms still suffer from the data sparsity problems. This paper presents a CF recommendation algorithm, using category information and influence of current items, which is based on the concept of influence set and is a hot topic in information retrieval system.

Keywords: Recommender Systems, Collaborative Filtering (CF), Item Similarity

1. Introduction

The "information overload" problem affects our everyday experience while searching for knowledge on a topic. To overcome this problem, we often rely on suggestions from others who have more experience on the topics[17]. However, in Web case where there are numerous suggestions, it is not easy to detect the trustworthy ones. Shifting from individual to collective suggestions, the process of recommendation becomes controllable. This is attained with the introduction of CF, which provides recommendations based on the suggestions of users who have similar preferences. Since CF is able to capture the particular preferences of a user, it has become one of the most popular methods in recommender systems[18].

Two types of CF algorithms have been proposed in the literature: memory-based algorithms, which recommend according to the preferences of nearest neighbors, and model-based algorithms, which recommend by first developing a model of user ratings. Both practical experience and related research have reported that memory-based algorithms (a.k.a. nearest-neighbor algorithms) present excellent performance, in terms of accuracy, for multivalue rating data. On the other hand, model-based algorithms are efficiently handle scalability to large data sets [19].

2. Related Work

In 1992, the Tapestry system introduced collaborative filtering (CF) [1]. In 1994, the GroupLens system implemented a CF algorithm based on common users' preferences [2]. In 2001, another CF algorithm was proposed [3]. It is based on the items' similarities for a neighborhood generation of nearest items and is denoted as item-based CF algorithm.

Most recent work followed the two aforementioned directions: Herlocker, Konstan, and Riedl (2002) weight similarities by the number of common ratings between users/items [4]. Deshpande and

Karypis (2004) apply item-based CF algorithm combined with conditional-based probability similarity and cosine-similarity [5]. Xue, Lin, and Yang (2005) suggest a hybrid integration of aforementioned algorithms (memory-based) with model-based algorithms [6].

Furnas, Deerwester, and Dumais (1988) proposed latent semantic indexing (LSI) in information retrieval area to deal with the aforementioned challenges. More specifically, LSI uses SVD to capture latent associations between the terms and the documents [7]. SVD is a well-known factorization technique that factors a matrix into three matrices. Berry, Dumais, and O'Brien (1994) carried out a survey of the computational requirements for managing LSI-encoded databases [8]. He claimed that the reduced dimensions model is less noisy than the original data.

Sarwar, Karypis, and Konstan (2000, 2002) applied dimensionality reduction for the user based CF approach. [9][10]

3. Introduction Item-based collaborative filtering algorithm

Sarwar et al. proposed an item-based collaborative filtering algorithm [20]. In Sarwar method, one important step is to compute a similarity among items, that is, the ratings for user's target items are predicted by selecting some one that is most similar to the target item, and then the highest rating in all ratings will be selected as the recommending item for the customers [20]. Initially, the purpose of item-based collaborative filtering algorithm is to address data sparsity problem and model expansion of user-based algorithm. Therefore, it also can become an extension of user-based algorithms. The data sparsity problem means that in database of recommender systems, the ratings of user are little in general so that the evaluating matrix generally is a sparse matrix. Using these matrixes as the evaluating matrix, it is difficult to find a similar user set or item set, so recommending quality will be poor. Another kind of problems is model expansion. When new items and new users are added regularly to recommender systems, due to collaborative systems dependent solely on users' preferences to make recommendations, the new item can not be recommended until it has rated by a substantial number of users. This problem can be solved using hybrid recommendation approaches, that is, user-item rating collaborative filtering.

It is well known that the basic idea of item-based collaborative filtering recommendation algorithm is that the recommendation is obtained with those k-neighbor item historical ratings comparing to predict the target item rating. The items historical ratings is obtained from the evaluating the similarity among different items. Therefore, the critical step in the item-based collaborative filtering algorithm is to compute the similarity between items and then to select the most similar items and it is also a hot topic in recommender system research. The basic idea in similarity computation between two items i and j is to first isolate the users who have rated both of these items and then to apply a similarity computation technique to determine the similarity $sim(i; j)$.

In the recommendation system, the Dataset is formed by m users(a user-set of m users, $U=\{u_1,u_2,...,u_m\}$,n items(a item-set of n items, $I=\{i_1,i_2,...,i_n\}$,user' ratings on the items(user-item ratings matrix,,a m×n matrix, $A_{m\times n}$).

3.1. Similarity measures

There are three basic similarity measures for depicting the variation between item *i* and *j* [20].

A. *Cosine-similarity*

$$sim(i,j) = Cos(\vec{i},\vec{j}) = \frac{\sum_{u\in U} R_{u,i}\cdot R_{u,j}}{\sqrt{\sum_{u\in U} R_{u,i}^2 \cdot \sum_{u\in U} R_{u,j}^2}} \quad (1)$$

Where $R_{u,i}$ is the rating of user u for item i, and similar to $R_{u,j}$.

B. *Adjusted cosine-similarity*

$$sim(i,j) = \frac{\sum_{u\in T}(R_{u,i} - \overline{R_u})\cdot(R_{u,j} - \overline{R_u})}{\sqrt{\sum_{u\in T}(R_{u,i} - \overline{R_u}\cdot\sum_{u\in U}(R_{u,j} - \overline{R_u})}} \quad (2)$$

Where $R_{u,i}$ is the rating of user *u* for item *i*, $\overline{R_u}$ is the average of all the ratings of user u and set T contains the users both rate item i and j.

C. *Pearson correlation*

$$sim(i,j) = \frac{\sum_{u\in T}(R_{u,i} - \overline{Ri})\cdot(R_{u,j} - \overline{R_j})}{\sqrt{\sum_{u\in T}(R_{u,i} - \overline{Ri}\cdot\sum_{u\in U}(R_{u,j} - \overline{R_j})}} \quad (3)$$

Where $R_{u,i}$ is the rating of user u for item i, $\overline{Ri}$ is the average of all the ratings of item i and set T contains the users both rate item i and j.

In this paper we propose that incorporating the item category into the similarity as in section D stated and then we union all

D. *Applying category on similarity*

Considering the information of category, as MovieLen dataset has the different movie category, like action, adventure, animation, comedy, etc, we tune similarity with this:

$$sim'(i,j) = \frac{\sum_{u\in T}(R_{u,i} - \overline{R_i})\cdot(R_{u,j} - \overline{R_j})}{\sqrt{\sum_{u\in T}(R_{u,i} - \overline{R_i}\cdot\sum_{u\in U}(R_{u,j} - \overline{R_j})}} \times \frac{I}{F} \quad (4)$$

Where $R_{u,i}$ is the rating of user u for item i, $\overline{Ri}$ is the average of all the ratings of item I and set T contains the users both rate item i and j. I is the number of category that both item i and j are included. F is the sum of category that at least one of the items involves.

E. *The UNION similarity measures*

We first examined the two most important factors that are involved in the first stage: sparsity and the similarity measure. As mentioned, for the formation of sets T (see Eqs. (1), (2), (3) and (4)), past work takes into account only the items that are co-rated by both users.

Example: In Table 1, an example that describes the ratings of four users, U_1− U_4, over six items. Comparing the similarity of U_1 and U_2 users, when only co-rated items are considered, then the similarity measure will be computed based only on the ratings for I_1 and I_3.In case of sparse data, we have a very small amount of provided ratings to compute the similarity measure. By additionally constraining T with co-rated items only, we reduce further the effective amount of used information. To avoid this, we consider alternative definitions for T, given in Eq. (5):

$$T = I_u \cap I_{vj} \quad (5)$$

	I_1	I_2	I_3	I_4	I_5	I_6
U_1	4	-	1	4	-	-
U_2	1	4	4	-	-	4
U_3	2	1	4	-	-	4
U_4	1	2	1	1	-	-

Table 1: A description the ratings of four users.

According to Eq. (5), *S* includes items rated by at least one of the users. In the Example of Table 1, except the ratings for I_1 and I_3, the ratings for I_2 and I_4 will be considered too. Notice that in case of UNION Pearson correlation, user ratings correspond to the average user ratings over all rated items.

3.2. Generation of recommendation list

For each recommendation system, two goals have been focused on:

- Predicting the missing ratings of a user;
- Generating a list of best items recommendations for a user.

Using aforementioned computing similarity measure, we form the item-neighborhood, where objects inside the neighborhood have similar ratings and behavior. Then predict the ratings by the user *u* for item i ($i \in I_u^{'}$). The prediction formula is as follows[11]:

$$r_{u,i} = \frac{\sum_{j \in M_i} sim'(i,j) \times R_{u,j}}{\sum_{j \in M_i} sim'(i,j)} \qquad (6)$$

Finally, we pick the items, having several top ratings as the top-N list.

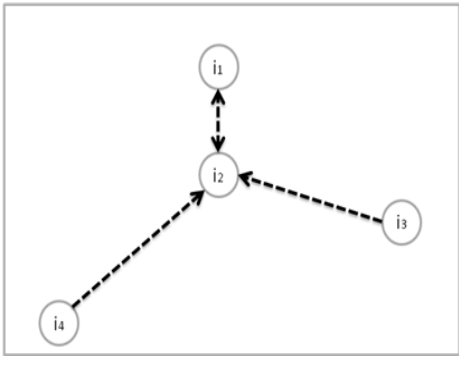

Fig. 1: An example of similarity relationships among items.

4. Applying influence on Item-based collaborative filtering algorithm

The traditional user-based collaborative filtering (CF) algorithms often suffer from two important problems [11]: Scalability and sparsity because of its memory-based k nearest neighbor query algorithm. Item-Based CF algorithms have been designed to deal with the scalability problems associated with user-based CF approaches without sacrificing recommendation or prediction accuracy. However, item-based CF algorithms still suffer from the data sparsity problems [20].

Combining the item-based CF with influence, it defines a new prediction computation method for this new recommendation mechanism [12].

4.1. Concept of influence set

In the traditional collective filtering algorithm, only one-side interaction has been considered. This prediction is partial, without including influence of the current item to others. However, a new item's appearance usually influences other early ones. Bringing the two influences together, we will improve the original algorithm performance [13].

Reverse k nearest neighbor (RkNN) is the reversal algorithm of kNN. In a c-dimension space, dataset S refers to similarity as the distance between two items. As aforementioned, D(p,q) stands for the similarity between p and q. The following is how RkNN is defined [14]:

$RkNN(q)=\{p \in S | q \in kNN(q)\}$

Note that, kNN and RkNN are asymmetric, as $p \in kNN(q) \neg \Rightarrow p \in RkNN(q)$ and the contrary is the same.

In Fig.1, the end of the arrow stands for the neighbor of the end form which the arrow comes. According to Fig.1, we can see, if u_a doesn't rate any neighbor of i_2, by traditional CF algorithm, we can't perform a prediction.

4.2. Generate a prediction

Based on the aforementioned analysis, we use the Eq.(7) to perform a prediction. Where $\overline{R_i}$ is the average of the ratings of item i. $sim'(i_j, i_i)$ uses Eq.(4) to process the similarity between i_j and i_i.

4.3. Steps of the algorithm

Part 1: Similarity computation
Inputs: User-item rating matrix R, item category matrix G
Outputs: kNN list and RkNN list
Step 1: According to the User-item rating matrix and item category matrix, using Eq.(4),compute the similarities and store the result in the similarity matrix.
Step 2: For every item i, find its nearest k neighbors in kNN list.
Step 3: According to the kNN list, in the contrat, perform the improved algorithm to find RkNNlist.
Part 2: Rating Prediction
Inputs: Target user
Outputs: ratings of missing items
Step 1: In the kNN list find k toppest items.
Step 2: In the RkNN list find k toppest items.
Step 3: Perform the prediction by Eq.(7)

$$r_{u,i} = \overline{R_i} + \frac{\sum_{i_j \in kNN(i_j)} (R_{u,j} - \overline{R_j}) \cdot sim'(i_j, i_i)}{\sum_{i_j \in kNN(i_j)} sim'(i_j, i_i) + \sum_{i_{j'} \in RkNN(i_j)} sim'(i_j, i_i)} + \frac{\sum_{i_{j'} \in RkNN(i_j)} (R_{u,j'} - \overline{R_{j'}}) \cdot sim'(i_{j'}, i_i)}{\sum_{i_j \in kNN(i_j)} sim'(i_j, i_i) + \sum_{i_{j'} \in RkNN(i_j)} sim'(i_{j'}, i_i)} \quad (7)$$

5. Experimentation Study

5.1. Dataset

Dataset in this paper comes from the Movielens dataset at Grouplens Research Project in Minnesota University.

5.2. Evaluating Metric

Often used evaluating method is statistical accuracy metrics that evaluates the accuracy of a system by comparing the numerical recommendation scores against the actual user ratings for the user-item pairs in the test dataset. Mean Absolute Error (MAE) between ratings and predictions is a widely used metric for evaluating the quality of recommendation [15]. MAE is a measure of the deviation of predicting ratings of recommendations from their true user-specified values. Let the actual user-specified value set is $\{q_1, q_2, ..., q_n\}$, and the predicting rating set by recommending algorithm is $\{p_1, p_2, ..., p_n\}$, then MAE is calculated as follows[16]:

$$MAE = \frac{\sum_{i=1}^{N} |p_i - q_i|}{N} \quad (8)$$

The lower the MAE, the higher of the recommending quality, because of the deviation of the predicting rating from the actual user scoring is small.

5.3. Steps of the algorithm

This experiment is to perform the comparison of traditional CF and the CF based on category and influence. Results displayed in Fig.2.

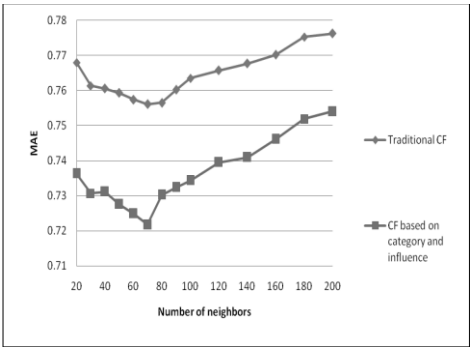

Fig.2: Comparison of two different algorithms.

6. Conclusions

This paper has improved the traditional collaborative filtering recommending algorithm by incorporating the item category clustering into similarity evaluation. It is obvious that incorporating category information into algorithm will increase the similarity of items. In particular, the more of number of the common categories is, the higher is the similarity of items. Moreover, fully use of the influence of new items can improve the quality of prediction. Note that, experimental comparison has shown that as the number of neighbors increase, minimum MAE point is about 70. It indicates that we should pick the appropriate number when performing this algorithm.

References

[1] M. Berry, S. Dumais, andG. O'Brien. Using linear algebra for intelligent information retrieval. SIAM Review, 1994, 37(4), 573 – 595.

[2] J. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In Proceedings of the conference on uncertainty in artificial intelligence, pp. 43 – 52, 1998.

[3] M. Deshpande, M., and G. Karypis. Item-based top-n recommendation algorithms. ACM Transactions on Information Systems, 2004, 22(1), 143 – 177.

[4] G. Furnas, S. Deerwester, and S. Dumais. Information retrieval using a singular value decomposition model of latent semantic structure. In Proceedings of the ACM SIGIR conference, pp. 465 – 480, 1988.

[5] D. Goldberg, D. Nichols, M. Brian, and D. Terry. Using collaborative filtering to weave an information tapestry. ACM Communications, 1992, 35(12), 61 – 70.

[6] K. Goldberg, T. Roeder, T. Gupta, and C. Perkins. Eigentaste: a constant time collaborative filtering algorithm. Information Retrieval, 2001, 4(2), 133 – 151.

[7] J. Herlocker, J. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In:Proceedings of the ACM SIGIR conference, pp. 230 – 237, 1999.

[8] J. Herlocker, J. Konstan, andJ. Riedl. An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. Information Retrieval, 2002, 5(4), 287 – 310.

[9] J. Herlocker, J. Konstan, L. Terveen, and J. Riedl. Evaluating collaborative filtering recommender systems. ACM Transactions on Information Systems, 2004, 22(1), 5 – 53.

[10] T. Hofmann, Latent semantic models for collaborative filtering. ACM

Transactions on Information Systems, 2004, 22(1), 89－115.

[11] Z. Huang, H. Chen, and D. Zeng. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. ACM Transactions on Information Systems, 2004, 22(1), 116－142.

[12] G. Karypis, G.Evaluation of item-based top-n recommendation algorithms. In Proceedings of the ACM-CIKM conference, pp. 247－254, 2001.

[13] P. McJones, and J. DeTreville. Each to each programmer’s reference manual. Tech. Rep. Systems Research Center, 1997-023, 1997.

[14] R. McLauglin, and J. Herlocher. A collaborative filtering algorithm and evaluation metric that accurately model the user experience. In Proceedings of the ACM SIGIR conference, pp. 329－336, 2004.

[15] B. Mobasher, H. Dai, T. Luo, and M. Nakagawa. Improving the effectiveness of collaborative filtering on anonymous web usage data. In Proceedings of the workshop intelligent techniques for web personalization, pp. 53－60, 2001.

[16] M. O’Mahony, N. Hurley, N. Kushmerick, and G. Silvestre. Collaborative recommendation: a robustness analysis. ACM Transactions on Internet Technology, 2002, 4(4), 344－377.

[17] R. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: an open architecture for collaborative filtering on netnews. In Proceedings of the conference computer supported collaborative work, pp. 175－186, 1994.

[18] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Analysis of recommendation algorithms for e-commerce. In Proceeedings of the ACM electronic commerce conference, pp.158－167, 2000.

[19] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Application of dimensionality reduction in recommender system－a case study. In ACM WebKDD workshop, 2000.

[20] B. Sarwar, G. Karypis, J. Konstan and J. Riedl. Item-based collaborative filtering recommendation algorithms. In Proceedings of the 10th international conference on World Wide Web, pp. 285－295, 2001.