2004 8th International Conference on
Control, Automation, Robotics and Vision
Kunming, China, 6-9th December 2004

Finding Item Neighbors in Item-based Collaborative Filtering by Adding Item Content

Chaiwat Tiraweerakhajohn, Ouen Pinngern

Department of Computer Engineering, Faculty of Engineering
Research Center for Communications and Information Technology (ReCCIT)
King Mongkut's Institute of Technology Ladkrabang
Bangkok, Thailand.
Email: chaiwat_ti@yahoo.com, kpouen@kmitl.ac.th

Abstract

In this paper we present an approach to that tries to alleviate the main item-based collaborative filtering (CF) drawback – the sparsity and the first-rater problem. By combining the contents of items into the item-based CF to find similar items and use the combined similarity to generate predictions. The first step concentrates is using association rules mining methods to discover new similarity relationships among attributes. The second step is to exploit this similarity during the calculation of item similar. Finally, combines new similarity and rating similarity measures to find neighbor item in item-based CF algorithm and generating ratings predictions based on a combined similarity measure. The experiments show that this novel approach can achieve better prediction accuracy than traditional item-based CF algorithm.

1 Introduction

In e-commerce environment, Collaborative Filtering (CF) is the most popular recommender system technology to date such as Amazon.com and CDNOW provide interesting and powerful recommendation services. The goal of a CF algorithm [1,2] is to recommend new items or predict the utility of a certain item for a certain user based on the user's preference and other user's opinions. In a CF scenario there is a list of m users $U = \{u_1,u_2,\ldots,u_m\}$ and a list of n items $I = \{i_1,i_2,\ldots,i_n\}$. Each user has a rating vector R, where r_k represents the user's rating for item i_k (The user's opinion and preference are explicitly given by the rating score.) In this paper we will concentrate on predicting the rating score for an unrated item i_k (called target item). Currently, there are mainly two types of CF algorithms, user-based and item-based. User-based CF algorithm uses some statistical techniques to find a set of users called user neighbors for the target user. Then different methods can be adopted to combine the user neighbors' ratings to product a prediction rating for the target user. Recently, a new class of item-based CF algorithm has been proposed to deal with the scalability problem in user-based CF algorithm [3]. The idea is rather than find similar users (user neighbors) this algorithm tries to find similar items that are rated by different users in some similar ways. Then for a target item predictions can be generated by taking a weighted average of the target user's ratings on these similar items. However the item-based CF algorithm still suffer from the problems associated with data sparsity, and they still lack the ability to provide recommendations or predictions for new or recently added items. The contribution of this paper is an approach to solve these problems that still exits in item-based CF algorithm in two distinct stages:

- We use association rule mining methods to discover new similarity relationships among attributes of items
- Finally we combine the attributes similarity and the rating similarity measures to find item neighbors and predictions based on a combined similarity measure.

2 Item-based CF Algorithm

Item-based CF algorithm looks into the set of items the target user has rated and computes how similar they are to the target item i and then selects k most similar items $\{i_1,i_2,\ldots,i_k\}$. At the same time their similarities are also computed. Once the most similar items are found and then, the prediction is computed by taking a weighted average of the target user's ratings on these similar items. Item-based CF algorithm consists of two processes, finding similar items and generating rating prediction based on similar item's ratings.

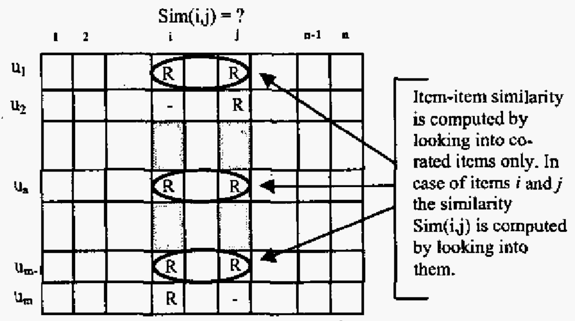

Sim(i,j) = ?
1
2
i
j
n-1
n
u1
u2
uu
um-1
um
R
-
Item-item similarity is computed by looking into co-rated items only. In case of items i and j the similarity Sim(i,j) is computed by looking into them.

Figure 1. Similarity computation

2.1 Finding Similar Items

The first step in computing the similarity of items i and item j is to identify all the users who have rated both item i and j (as show in Figure.1) illustrates this step, where the matrix rows represent users and the columns represent items. There are a number of different methods to compute the similarity between items. The most popular methods are given below:

- The cosine-based similarity is presented in Equation (1). In this case, two items i and j are considered as two column vectors in the user ratings matrix R. The similarity between them is measured by computing the cosine of the angle between these two vectors.

$$Sim(i,j)=\frac{\sum_{u\in U}R_{u,i}*R_{u,j}}{\sqrt{\sum_{u\in U}R_{u,i}^{2}}\sqrt{\sum_{u\in U}R_{u,j}^{2}}} \qquad (1)$$

where U means the set of users who both rated i and j, $R_{u,i}$ means the rating of user u on item i.

- The correlation-based similarity is presented in Equation (2).

$$Sim(i,j)=\frac{\sum_{u\in U}(R_{u,i}-\bar{R}_i)(R_{u,j}-\bar{R}_j)}{\sqrt{\sum_{u\in U}(R_{u,i}-\bar{R}_i)^2}\sqrt{\sum_{u\in U}(R_{u,j}-\bar{R}_j)^2}} \qquad (2)$$

where $\bar{R}_i$ means the average rating of the i-th item.

- The adjusted cosine similarity is presented in Equation (3)

$$Sim(i,j)=\frac{\sum_{u\in U}(R_{u,i}-\bar{R}_u)(R_{u,j}-\bar{R}_u)}{\sqrt{\sum_{u\in U}(R_{u,i}-\bar{R}_u)^2}\sqrt{\sum_{u\in U}(R_{u,j}-\bar{R}_u)^2}} \qquad (3)$$

where $\bar{R}_u$ means the average rating of user u.

2.2 Prediction Computation

The next step is to select a set of most similar items to the target item and generate a predicted rating for the target item using weighted sum as show in Equation (4).

$$P_{u,i}=\frac{\sum_{k\in K}(Sim(i,k)*R_{u,k})}{\sum_{k\in K}(|Sim(i,k)|)} \qquad (4)$$

where $P_{u,i}$ means the prediction rating of target user u on item i. Only the K nearest neighbors of item i.

3 Overview of Adding Item Content

Our approach, we integrate the similarity attributes and item ratings to calculate the item-item similarity in the finding item neighbors step.

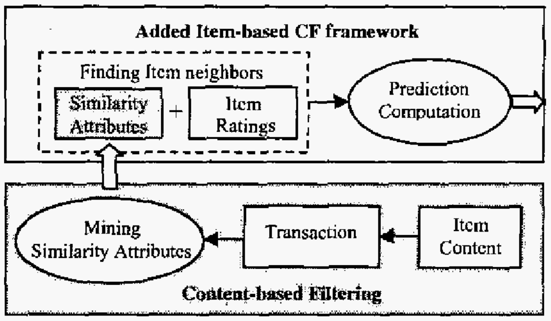


Figure 2. Added item-based CF framework

In Figure 2 shows the detail of our approach describes as follows:

- Represents the contents of items as a set of boolean values, each row represents an item and each column represents a unique attribute value. Then convert the contents of items into transaction of attributes.
- Apply association rules mining to discover similarity relationship among attributes of items, and exploit this new similarity to find similar items.
- Compute the similarity: firstly, calculate the similarity of item content using new similarity that will be describe in the section 3.2, then calculate the similarity of item ratings using user-item rating matrix that described in section 2.1. At last, the total similarity is the linear combination of the above two.
- Make a prediction for an item by performing a weighted average of deviations from the neighbour's mean.

3.1 Mining Similarity Attributes

In this section, we describe applying association rules mining, in particular the Apriori algorithm [5] to extract association rules between each pair of attributes in transaction of attributes. The association rules [6] are of the form $X \Rightarrow Y$, where X and Y are sets of items (item contents). The transaction T contains a certain itemset (set of attributes) X, then the transaction probably contains another itemset Y. The probability that a given rule holds, the *confidence*, is the percentage of transactions containing Y given that X occurs:

$$P(Y\subseteq T \mid X\subseteq T) \qquad (5)$$

The support of an itemset X is defined as the fraction of transactions supporting X with respect to the entire database. The support of a rule $X \Rightarrow Y$ is the probability that both itemsets occur together in a transaction:

$$Support(X \Rightarrow Y) = P((X \cup Y) \subseteq T) \quad (6)$$

The measure of rule confidence is related to support, and can be computed as follow:

$$Confidence(X \Rightarrow Y) = \frac{Support(X \cup Y)}{Support(X)} \quad (7)$$

The association rules that satisfy user specified minimum support threshold and minimum confidence threshold are called strong association rules.

Treating item contents as transactions and the item attributes therein as itemsets, the Apriori algorithm can be used to derive a set of attribute-attribute rules and associated confidence levels. We have limited this initial phase of the work to rules with single-attribute antecedents and consequents. Table 1 shows some sample rules that were generated by running Apriori on the EachMovie dataset. The confidence values are taken as probabilities and used to fill in the attribute-attribute similarity matrix, as shown in Table 2, which provides the additional similarity attributes necessary to find similar items.

Table 1: Sample association rules

Rule	Support	Confidence
Action ⇒ Comedy	25%	34%
Action ⇒ Drama	7%	11%
Action ⇒ Romance	14%	17%
Comedy ⇒ Drama	50%	66%
Comedy ⇒ Romance	5%	7%

Table 2: A simple attribute-attribute similarity matrix

	Action	Comedy	Drama	Romance
Action	1	0.34	0.11	0.17
Comedy	-	1	0.66	0.7
Drama	-	-	1	-
Romance	-	-	-	1

3.2 Finding Similar Items Using Similarity Attributes

In this section we descript how attributes similarity can be usefully exploited to finding similar items. The item similarity metric is the mean of the similarities between the attributes in the target item case t and the compare item case c as follows.

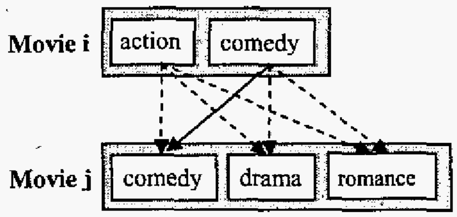
$$ISim(t,c,n) = \frac{\sum_{t_i \in t} ASim(t_i,c,n)}{t} \quad (8)$$

As for, The computation of the similarities between the attributes in the item cases t and c consists of two cases as show in Equation (9) and (10).

$$ASim(t_i,c,n) = 1, if \exists t_i \in t : t_i = c_j \quad (9)$$

$$= \frac{\sum_{j=1..n} Sim(t_i,c_j)}{n} \quad (10)$$

In the situation where there is a direct correspondence between an attribute in the target, t_i and the compare, c_j, the maximal similarity is assumed, otherwise, the similarity value of attribute in the target item is computed as the mean similarity between this attribute and the n most similar attributes in the compare item c ($c_1, c_2, \ldots, c_n$). Figure 3 shows how the find similar items using similarity attributes works on given movies, with a solid arc showing attribute overlap and dashed arcs representing the new similarity through association rules.



ASim("action", j, 3)	**=**	**<u>0.31</u>**
action ⇒ comedy		0.34
action ⇒ drama		0.11
action ⇒ romance		0.17
ASim("comedy", j, 3)	**=**	**<u>1</u>**
comedy ⇒ comedy		1
comedy ⇒ drama		0.66
comedy ⇒ romance		0.7
ISim(i, j, 2)	**=**	**<u>0.65</u>**

Figure 3. Finding similar items using similarity attributes

3.3 Prediction Based on a Combined similarity

The item similarities measure $ISim(i,j)$, for a pair of item i and j, is computed using the similarity attributes as described in Section 3.2. Similarly, we compute item similarities based on the user-item matrix as described in Section 2.1, we denote the rating similarity between two item i and j as $RSim(i,j)$. Finally, for each pair of items i and j, we combine these two similarity measures to get $CSim(i,j)$ as their linear combination as show in Equation (11).

$$CSim(i,j) = \alpha \times ISim(i,j) + (1-\alpha) \times RSim(i,j) \quad (11)$$

where α is a *combination parameter* specifying the weight of similarity in the combined measure. If $\alpha = 0$, the *CSim(i,j)* = *RSim(i,j)*, in other words we have the standard item-based CF. On the other hand, if $\alpha = 1$, then only the attribute similarity is used which, essentially, result in a form of content-based filtering.
In order to compute predicted ratings, we use the weighted sum approach described in Section 2.2. Specifically, as show in Equation (12).

$$P_{u,i} = \frac{\sum_{k \in K} \left(CSim(i,k) * R_{u,k} \right)}{\sum_{k \in K} \left(\left| CSim(i,k) \right| \right)} \qquad (12)$$

The only difference is that, here we will use the combined item similarity *CSim(i,j)* instead of *Sim(i,j)* to generate prediction rating.

4 Experiments and Results

4.1 Dataset

We perform experiments on a subset of movie rating data collected from the EachMovie dataset [7]. The EachMovie dataset deals with movie items using data from the EachMovie collaborative filtering site deployed by Digital Equipment Research Center from 1995 through 1997. The EachMovie dataset consists of 2,811,983 preferences for 1,628 movies rated by 72,916 users. User preferences are represented by means of numeric values from 0 to 1.0, such as 0, 0.2, 0.4, 0.6, 0.8, 1.0. In EachMovie dataset, genre can have 10 different values such as action, animation, art-foreign, classic, comedy, drama, family, horror, romance, and thriller. In our experiment, we retrieved 200 users and 200 movies. Furthermore, we randomly chose 90% of the ratings data as training dataset and the 10% as test dataset

4.2 Evaluation Metrics

In our experiments, we used MAE (Mean Absolute Error) as our evaluation metrics to measure prediction accuracy by comparing the numerical prediction scores against the actual user ratings in the test data. The MAE is calculated by summing these absolute errors of the corresponding rating-prediction pairs and then computing the average as show in Equation (13).

$$MAE = \frac{\sum_{u=1}^{N} \left| P_{u,i} - R_{u,i} \right|}{N} \qquad (13)$$

where $P_{u,i}$ means the user u prediction on item i and $R_{u,i}$ means the user u rating on item i in the test data, N is the number of rating-prediction pairs between the test data and the prediction result. The goal of an algorithm is to achieve as low a MAE as possible.

4.3 Experimental Results

The first experiment is the quality comparison of three different similarity algorithms such as cosine, correlation and adjusted cosine as described in Section 2.1. We implemented three different similarity algorithms with value of combination parameter α and tested them on our dataset. We performed an experiment where we varied the value of combination parameter α from 0 to 1 in an increment of 0.1 and determine the size of neighbors = 10. We ran these experiments on our training dataset and used test dataset to compute MAE. In Figure 4 shows the results of first experiments. We can observe that the value of combination parameter α does affect the quality of prediction. When combination parameter α = 0, which means traditional item-based CF. When α between 0.1-0.9 means combining attributes similarity with rate similarity. When α = 1, which means only using attribute similarity to find similar items and predict rating. Especially, the adjusted cosine similarity given the best result when α = 0.4, after this point the error values start increasing. Hence, we select the adjusted cosine similarity for the next experiment.

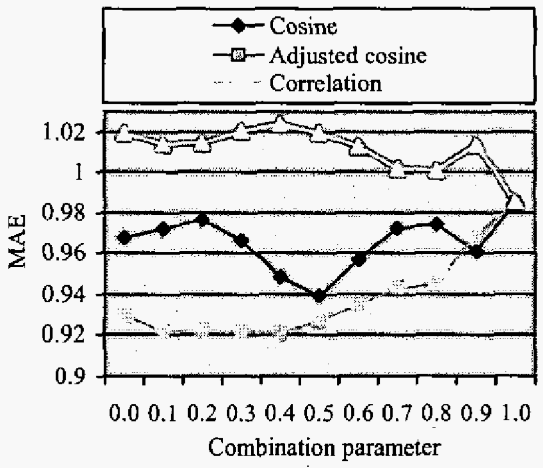


Figure 4. Comparison of prediction quality

The second experiment is the size of the neighbourhood. We performed an experiment where we varied the number of neighbors to be used for adjusted cosine similarity and computed MAE. In Figure 5 shows the results of our experiments. We can observe that the size of neighborhood does affect the quality of prediction.

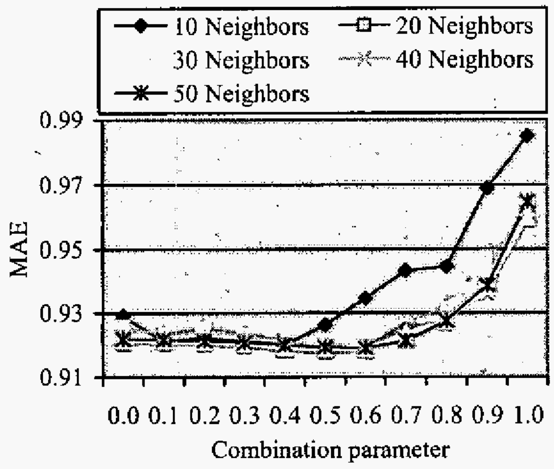


Figure 5. Experiments with neighborhood size

The final experiment, we implemented two different similarity attributes methods – one is Euclidean distance and our new similarity methods. In Figure 6 shows the results of experiments. We can observe that our approach for new similarity has a trend to show better performance than the Euclidean distance method, especially, when α = 1 the difference is so much, otherwise the difference is negligible

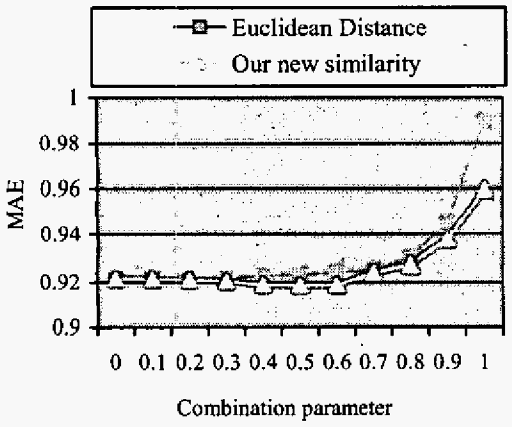


Figure 6. Comparison of our approach and Euclidean distance at the size of neighbors = 30

5 Conclusion and Future Work

In this paper, we combine the strengths of content-based filtering techniques with item-based CF to provide more accurate predictions which solves the sparsity and the first-rater problems. The results of our experiments show that the combination of attributes similarity and rating similarity can achieve better prediction accuracy than traditional item-based CF algorithms. For future work, we will continue to evaluate the effectiveness of our approach across a variety of collaborative filtering datasets and to make comparative evaluations to other techniques.

References

[1] J. Herlocker, J. Konstan, A. Borchers and J. Riedl, "An Algorithmic Framework for Performing Collaborative Filtering," in *Proceedings of the 1999 Conference on Research and Development in Information Retrieval*, Berkeley, CA, 1999.

[2] E. Vozalis and K. G. Margaritis, "Analysis of Recommender Systems'Algorithms," in *Proceedings of the 6th Hellenic European Conference on Computer Mathematics and its Applications (HERCMA-2003)*, 2003.

[3] B. Sarwar, G. Karypis, J. Konstan and J. Riedl, "Item-based Collaborative Filtering Recommenda tion Algorithms," *10th Int'l World Wide Web Conference, ACM Press*, pp. 285-295, 2001.

[4] R. Srikant and R. Agrawal, "Mining quantitative association rules in large relational tables," in *Proceeding of the 1996 ACM SIGMOD Conference on Management of Data, Montreal*, Canada, June 1996.

[5] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules," *IBM Almaden Research Center Technical Report RJ9839*, 1994.

[6] R. Agrawal, T. Imielinski, and A. Swami, "Mining association rules between sets of items in large databases," in *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pp. 207-216, Washington, D.C., May 1993.

[7] P. McJones, "EachMovie collaborative filtering data set," 1997.