An Improved Item-based Collaborative Filtering Algorithm Based on Clustering Method

Junhao WEN*, Wei ZHOU

College of Computer Science, Chongqing University, Chongqing 400044, China

Abstract

Item-based collaborative filtering recommendation algorithm is one of the most widely used recommendation algorithm, which is widely used in many recommendation systems. But there are some drawbacks when used in large e-business systems. The existing traditional algorithms can’t perform well when the item space changes; on the other side, the performance of the recommendation system will go down as the items increase into a large amount. An improved collaborative filtering recommendation algorithm based on dynamic item clustering method was proposed in the paper. A similitude threshold model was introduced to divide the item space into clusters dynamically. Experiment states that using dynamic item clustering method can satisfy the requirement of increasing amount users and consumers in large e-business systems. The improved collaborative filtering recommendation algorithm performs relatively good recommendation with less resource consumption.

Keywords: Recommendation System; Item-based; Collaborative Filtering; Recommendation Algorithm; Clustering; MAE

1 Main Text

The scale of E-business has grown extraordinarily since the spread of the Internet. Facing a mass of product information, users (consumers) are often difficult to find the most necessary or most suitable products. A mass of transaction data will be conducted by e-business system, how to dig and discover useful knowledge to make transactions more efficient is a meaningful research [3]. Collaborative filtering is the process of predicting ratings based on a database of ratings from various users [2]. It is widely applicable to e-Business, e-Learning, and so on [3]. Currently, many e-business sites are already using the recommended system, such as Amazon, CDNow, Drugstore and Moviefinder etc.

There are some problems in traditional collaborative filtering that researchers are trying to solve, such as efficiency and scalability of the recommendation system.

Firstly, sparse matrixes lead to inaccuracy of the recommendation system, which is a common problem in recommendation system. The number of items is very huge actually, the rating matrix
will very sparse in the situation. Secondly, the efficiency of the recommendation system will go down with the increment of items and users. With the increment of the rating items, the matrixes dimension be larger, the expense of conducting each recommendation will increases. Last but not least, the scalability problem. The known rating information changes with time. A number of practical scenarios require dynamic adaptive collaborative filtering that can allow new users, items and to be added in the system at a rapid rate. But the existing algorithm can’t satisfy the requirement.

To solve the above problems, we proposed an improved collaborative filtering recommendation algorithm based on dynamic item clustering method. A similitude threshold limits the similitude between clusters. By calculating the similitude between the current item and the cluster center, choose the greatest similitude cluster, and then find the target items’ nearest neighbors, Produce recommendation results using collaborative filtering.

## 2 Traditional Item-based Collaborative Filtering

Recommendation Algorithm

Item-based Collaborative filtering is a method of making automatic predictions (filtering) about the interests of a user by collecting preferences or taste information from many users (collaborating) [4, 8]. The underlying assumption of the approach is that those who agreed in the past tend to agree again in the future. By analyzing historical data, collaborative filtering technology generates result set which is the most similar with the current users’ interests. There are three phases in the algorithm: data representation; finding the nearest neighbor; generating recommended result sets [4, 6].

### 2.1 Data representation

A model is built to represent the rating items. The input data of the algorithm is presented by a matrix, \( m \) rows for \( m \) users. While \( n \) for \( n \) items being evaluated; items in the matrix represent evaluation of items by users. Different methods can be used to represent the evaluation. Discrete value (such as 1, 2, 3, 4, 5) can be used to represent preference degrees of the user to the items. We agree that the higher of the discrete value, the more prefer of the user on the item.

<table>
<thead>
<tr>
<th>User</th>
<th>Item _1</th>
<th>( \cdots )</th>
<th>Item ( k )</th>
<th>( \cdots )</th>
<th>Item ( n )</th>
</tr>
</thead>
<tbody>
<tr>
<td>User ( 1 )</td>
<td>( R_{1,1} )</td>
<td>( \cdots )</td>
<td>( R_{1,k} )</td>
<td>( \cdots )</td>
<td>( R_{1,n} )</td>
</tr>
<tr>
<td>( \cdots )</td>
<td>( \cdots )</td>
<td>( \cdots )</td>
<td>( \cdots )</td>
<td>( \cdots )</td>
<td>( \cdots )</td>
</tr>
<tr>
<td>User ( j )</td>
<td>( R_{j,1} )</td>
<td>( \cdots )</td>
<td>( R_{j,k} )</td>
<td>( \cdots )</td>
<td>( R_{j,n} )</td>
</tr>
<tr>
<td>( \cdots )</td>
<td>( \cdots )</td>
<td>( \cdots )</td>
<td>( \cdots )</td>
<td>( \cdots )</td>
<td>( \cdots )</td>
</tr>
<tr>
<td>User ( m )</td>
<td>( R_{m,1} )</td>
<td>( \cdots )</td>
<td>( R_{m,k} )</td>
<td>( \cdots )</td>
<td>( R_{m,n} )</td>
</tr>
</tbody>
</table>
2.2 Finding the nearest neighbors

At this stage, items which are most similar to the target user are found. First, similitude between user and other users are acquired. A few of methods we can choose from to calculate similitude between two items. Such as Cosine-based Similitude, Adjusted Cosine Similitude and Pearson Correlation Coefficient. We use Pearson Correlation Coefficient in the paper.

Suppose that \( i \) and \( j \) are two users in the user space, we use Pearson Correlation Coefficient to calculate the similitude of \( i \) and \( j \) \( \text{sim}(i, j) \).

\[
\text{sim}(i, j) = \frac{\sum_{c \in I_{i,j}} (R_{i,c} \cdot \overline{R_i}) (R_{j,c} \cdot \overline{R_j})}{\sqrt{\sum_{c \in I_{i,j}} (R_{i,c} - \overline{R_i})^2} \sqrt{\sum_{c \in I_{i,j}} (R_{j,c} - \overline{R_j})^2}}
\]  

In equation (1), \( \text{sim}(i, j) \) is the similitude of user \( i \) and user \( j \). \( R_{i,c} \) is the evaluation value of item \( c \) by user \( i \); \( R_{j,c} \) is the evaluation value of item \( c \) by user \( j \); while \( \overline{R_i} \) and \( \overline{R_j} \) are the average evaluation value on the item by user \( i \) and user \( j \); \( I_{i,j} \) for the item set that user \( i \) and user \( j \) have evaluated on in common.

2.3 Generating recommended result sets

In this phase, recommended results set is generated by the algorithm. Suppose \( I_u \) is the set that user \( u \) had evaluated on, evaluation \( p_{u,i} \) on item \( i \) by target user \( u \) can be got in this way:

\[
p_{u,i} = \overline{R_u} + \frac{\sum_{v \in N_u} \text{sim}(u, v) \cdot (R_{v,j} - \overline{R_v})}{\sum_{v \in N_u} \text{sim}(u, v)}
\]

Items that are not evaluated before can be predicted by items that had been predicted.

3 The New Algorithm

![Fig. 1: The process of the algorithm](image-url)
When given enough and clear information, the traditional algorithm usually shows good performance. But with the increment of users and items of the rating matrix, item-based collaborative filtering algorithm gradually exposes some shortcomings. With items and users added in the rating matrix every day, we need an algorithm that adjusts to the dynamic change of the rating matrix.

There are $x$ clusters in the cluster set, each cluster has many items which have similar ratings. The cluster center of each cluster is the mean rating value of all the items in the cluster. When a new item is added in, the similitude between the item and other cluster centers will be calculated. If the max value of similitude is bigger than the threshold, the item will be added to the cluster which has the biggest similitude with the item. The cluster center will be recalculated. Else a new cluster center will be build with the rating score of the item be the cluster center of the new cluster.

Algorithm 1:

Definition:

Collection of all items $I = \{i_1, i_2, \cdots, i_m, \cdots, i_n\}$, while $i_m$ is an item of the collection $I$.

Collection of all items $U = \{u_1, u_2, \cdots, u_a, \cdots, u_b\}$, while $u_m$ is an item of the collection $U$.

Collection of cluster $C = \{c_1, c_2, \cdots, c_p, \cdots, c_x\}$, while $c_p$ is a cluster of the cluster collection $C$. Initialize $C = \phi, x = 0$.

Collection of cluster centers $CC = \{cc_1, cc_2, \cdots, cc_p, \cdots, cc_x\}$, while $cc_p$ is a cluster center of the collection $CC$. Initialize, $CC = \phi, x = 0$.

Input: User Rating Database (URDB) and similitude threshold $threshold$.

Output: $x$ clusters

1. Retrieve all $n$ items from the database URDB, assigned as a collection $I = \{i_1, i_2, \cdots, i_m, \cdots, i_n\}$, while $i_m$ is an item in $I$.

2. The first item $i_1$ is retrieved from URDB; initialize an empty cluster, assign the cluster center of the cluster with the score of the item in URDB. $I = I - i_1, x = 1$.

3. for each item $i_m$ in collection $I$

4. for each cluster center $cc_p$

5. Calculate the similitude of item $i_m$ and cluster center $cc_p$, $sim(i_m, cc_p)$

6. endfor;

7. $max(i_m, CC) = sim(i_m, cc_p) = max(sim(i_m, cc_1), sim(i_m, cc_p), \cdots, sim(i_m, cc_x))$

8. if $max(i_m, CC) < threshold$

9. $x = x + l$, add a new cluster $cc_x$, assign the cluster center of $cc_x$ with the score of the item in URDB. $I = I - i_m$.

10. else $c_p = c_p + i_m$, $I = I - i_m$

11. for each user $u_a$ in $U$
12. Recalculate the mean score of the items in the cluster by user $u_a$. Generate the new cluster center.

13. endfor;

14. end;

A cluster center is the typically score of the cluster. The similitude score of items in the same cluster as high as possible, and similitude score of items as low as possible between different clusters. In the next step, choose the highest similitude clusters with the target item as the search space, finding the nearest neighbor in the search space.

Algorithm 2 Finding the nearest neighbors

Definition:

TargetClusters: Collection of the clusters that fulfill the condition of similitude threshold. Initialize NULL.

NearestNeighbors: Output collection of the algorithm. Initialize NULL.

Input: the target item; numbers of nearest neighbors $k$; URDB, Collection of clusters $C$; collection of cluster centers $CC$; the threshold of similitude $simthre$.

Output: $k$ nearest neighbors of the target item.

1. TargetClusters = NULL,

2. for each cluster $c_p$ in collection $C$

3. Calculate the similitude of item $i_m$ and cluster of $c_p$

4. if $sim(i_m, cc_p) > simthre$

5. TargetClusters = TargetClusters + $c_p$; $C = C - c_p$.

6. Endif;

7. Endfor;

8. Sort items in TargetClusters by the similitude with item $i_m$, find the top-$k$ neighbors.

9. End;

A rating matrix is constructed according to the generated cluster centers.

Cluster center score data matrix shown in Figure 2, $m$ lines represents the users, $x$ columns represents cluster centers. Elements on Row $i$ and Column $j$ $R_{ij}$ on behalf of the average score of cluster $j$.

Cluster center rating matrix ClusteCenterMatrix is used to calculate the similitude between target items and cluster centers.

Generating recommended result sets:
Equation (2) is used to generate recommended result sets. Predict scores for items that are not rated before, and then select the top $n$ results as recommendations back to the user.
Table 2: User rating matrix of the new algorithm

<table>
<thead>
<tr>
<th></th>
<th>c_{1}</th>
<th>\cdots</th>
<th>c_{k}</th>
<th>\cdots</th>
<th>c_{x}</th>
</tr>
</thead>
<tbody>
<tr>
<td>User 1</td>
<td>R_{1,1}</td>
<td>\cdots</td>
<td>R_{1,k}</td>
<td>\cdots</td>
<td>R_{1,x}</td>
</tr>
<tr>
<td>\cdots</td>
<td>\cdots</td>
<td>\cdots</td>
<td>\cdots</td>
<td>\cdots</td>
<td>\cdots</td>
</tr>
<tr>
<td>User j</td>
<td>R_{j,1}</td>
<td>\cdots</td>
<td>R_{j,k}</td>
<td>\cdots</td>
<td>R_{j,x}</td>
</tr>
<tr>
<td>\cdots</td>
<td>\cdots</td>
<td>\cdots</td>
<td>\cdots</td>
<td>\cdots</td>
<td>\cdots</td>
</tr>
<tr>
<td>User m</td>
<td>R_{m,1}</td>
<td>\cdots</td>
<td>R_{m,k}</td>
<td>\cdots</td>
<td>R_{m,x}</td>
</tr>
</tbody>
</table>

4 Experiments Design

4.1 Dataset

We used MovieLens dataset to test the algorithms. It can be downloaded from the GroupLens Research Project website (http://www.grouplens.org/node/12). MovieLens is an experimental platform for studying recommender systems. MovieLens data sets were collected by the GroupLens Research Project at the University of Minnesota. The data was collected through the MovieLens web site (movielens.umn.edu). This data set consists of 100,000 ratings (0-5) from 943 users on 1682 movies. We divide 80% of the dataset as the train set, 20% of the dataset as the test set of the experiments.

Two factors were considered in the experiment design, mean absolute error (MAE) and mean consumed time (MCT) on conducting the recommendation. In order to verify the effectiveness of the improved algorithm, we focused on these two factors to compare experimental results. MAE is a quantity used to measure how close forecasts or predictions are to the eventual outcomes. We use MAE (mean absolute error) to evaluate the accuracy of the method. Definition of MAE (mean absolute error).

\[
MAE = \frac{\sum_{i=1}^{N} |p_i - q_i|}{N} \tag{3}
\]

\{p_1, p_2, \cdots, p_N\} is the set of predictions by the algorithm, and \{q_1, q_2, \cdots, q_n\} is the real rating by users.

MCT (Mean Consumed Time) is the measurement of the time consumed to conduct the prediction. MCT (mean consume time) to evaluate the efficiency in this paper. The shorter MCT, the more efficient of the algorithm is.

Definition of MCT (mean consumed time)

\[
MCT = \frac{\sum_{i=1}^{N} t_i}{N} \tag{4}
\]

While \(t_i\) is the time consumed when generating the recommendation of item \(I_i\)

4.2 Experimental results

We make 3 experiments to test the algorithm. The first one we find the best threshold value. The second one the new algorithm was compared with other algorithms. The first experiment states
the influence that different threshold values have on MAE. Fig 2 shows that when the threshold is 0.6, the MAE is the lowest. So we choose 0.6 as the threshold.

In the next two experiments, we compare our new method with the existing algorithms with the threshold value 0.6. We will compare the MAE when the value of nearest neighbors change.

We can see from Fig 3 that, MAE value of the new method is lower than the traditional collaborative filtering, but a litter higher than the Item-based collaborative filtering. The performance of the recommendation was compared in the next experiment by the mean consumed time (MC-T); we use three methods to generate the same recommendations 100 times each algorithm. We calculate the time consumed to conduct a single recommendation using the same data set. The experiment stated that MCT of the new algorithm smaller than the traditional algorithm in the same condition. With the increase of neighbors, the gap is widening.
5 Conclusion

With the amount of items and users increasing every day, there are many items added in the item space all the time. The existing algorithm cannot perform recommendation very well. An improved collaborative filtering recommendation algorithm based on dynamic item clustering method was proposed in the paper. The new items can be added in the clusters dynamically without changing too much and less resource demanding. Experiments shows that the improved method has lower MAE and less resource consumption compared with other algorithms. But there are some problems being solved. For example, there may be too many items in a single cluster after the cluster division. The efficiency of the recommendation will be low because of the items amount of the cluster. In the future research, we will focus on research in this area.

Acknowledgement

This work is supported by National Natural Science Foundation of China (No. 61075053) and supported by Natural Science Foundation Project of CQ CSTC (No. 2010BB2244) and supported by the Fundamental Research Funds for the Central Universities Project (No. CDJXS11181161).

References