Recommendation with Item Clustering Based Collaborative Filtering

Xin Wang[1,a], Zhi Yu[1,b] and Can Wang[1,c]

[1]Zhejiang Provincial Key Laboratory of Service Robot, College of Computer Science, Zhejiang University, Hangzhou 310027, China

[a]xinwang@zju.edu.cn, [b]yuzhirenzhe@zju.edu.cn, [c]wcan@zju.edu.cn

Keywords: recommendation; collaborative filtering; clustering

Abstract. Recommender systems are playing a more and more important roles in people's daily life and collaborative filtering (short for CF) is a widely used approach in recommender systems. In practice, many E-commerce companies such as Amazon use CF to make recommendations. However, as the number of users and items grow larger and larger, CF are suffering two kinds of problems: sparsity and scalability. So in this paper, we propose an item clustering based CF to solve these two problems. The experiments show that our method outperforms the traditional CF in term of both predicting accuracy and running time.

Introduction

In CF, for a target user, we may first find its similar users and then recommends these users' highly rated items to this user (user based CF) or we may also recommend him those items which are similar to some item rated a high score by this target user (item based CF). In practical recommender systems, even active users may have purchased well under 1% of the items. So for most recommender systems, the numbers of effective ratings are far less than the number of users and items, which reduces the accuracy of the recommendations. Moreover, nearest neighbor algorithms also require computations which grow dramatically with both the number of users and the number of items. With millions of users and items, running CF for each user and item will suffer from scalability problems because searching and analyzing millions of potential nearest neighbors in a very short time is very challenging. User clustering based approach is proposed in [4,5] to enhance the accuracy and scalability of CF. On the other hand, items do not change as quickly as users do, which will probably make the result of clustering more stable. So in this paper, instead of clustering users, we try to use an item clustering based CF algorithm which adopts a similar data smoothing strategy as that of [5] to improve the performance of traditional CF algorithm.

Proposed Approach

In this section, we'll discuss our item clustering based CF approach in detail. Actually there are more than one clustering algorithms that can be used to cluster items. In this paper, we choose the well-known K-means algorithm as our basic clustering algorithm. Moreover, the number K is a preset value which specifies the desired number of clusters. Similarly, the number k is also a preset values which specifies the desired number of nearest neighbours when predicting the rating of user u on item i. Below shows the general steps of our proposed approach.

1. Cluster the items into different clusters according to similarities between items through K-means clustering.
2. Within each cluster, reduce the data sparsity in the rating matrix with data smoothing strategy if the density of the data is less than 20%.
3. Select the 10 most similar item clusters (which are also called pre-selected neighbours) for the target item.
4. For the target item, select its k most similar items from the item cluster(s) chosen in last step.
5. Predict the rating of a target user on a target item.

Detailed Descriptions of Our Approach. Let's denote the rating of user *u* on item *i* as $r_{u,i}$ and rating matrix as $R_{m\times n}$. We first conduct clustering on items using K-means algorithm and cluster items into *K* clusters, denoted as C_P (P = 1, 2, ... , K). The distance between two items in K-means can be defined as the similarity (Pearson correlation coefficient) between them, which is

$$\text{dist(i, j)} = \text{sim(i, j)} = \frac{\sum_{u\in I_{ij}} \left(r_{u,i} - \bar{r}_i\right)\left(r_{u,j} - \bar{r}_j\right)}{\sqrt{\sum_{u\in I_{ij}} \left(r_{u,i} - \bar{r}_i\right)^2 \sum_{u\in I_{ij}} \left(r_{u,j} - \bar{r}_j\right)^2}}. \quad (1)$$

where I_{ij} is the set of users who rate both item *i* and item *j*, and $\bar{r}_i$ is the mean rating on item *i*. After the clustering, there will be a new rating matrix R_P (which is a sub matrix of $R_{m\times n}$) for each cluster C_P. Next we will calculate the density of each cluster and check if it is less than our preset threshold (which is 20% in our project). If the sparsity of a cluster C_P is lower than the threshold, we will use the data smoothing strategy proposed in [5] to approximate the missing value in R_P to reduce the sparsity.

$$r_{u,i} = \begin{cases} r_{u,i} & \text{if user u rates item i} \\ r'_{u,i} & \text{otherwise} \end{cases}. \quad (2)$$

where $r'_{u,i}$ denotes the smoothed value for user *u's* rating on item *i*. Given an item *i*, C_i refers to the cluster this item *i* belongs to. And $r'_{u,i}$ is defined as:

$$\text{r}'_{u,i} = \bar{r}_i + \Delta r_{u,C_i}. \quad (3)$$

where $\bar{r}_i$ denotes the average rating of item i and $\Delta r_{u,C_i}$ denotes the average deviations rating for user *u* on items in cluster C_i, which is:

$$\Delta r_{u,C_i} = \sum_{i'\in C_i(u)} \left(r_{u,i'} - \bar{r}_{i'}\right) / \left|C_i(u)\right|. \quad (4)$$

where $C_i(u) \in C_i$ is the set of items in cluster C_i which are rated by user *u* and $|C_i(u)|$ is the number of items in $C_i(u)$. Then the similarity between item cluster *C* and the target item is also calculated based on Eq. 5:

$$\text{sim(i, C)} = \frac{\sum_{\text{u}\in I_{i,C}} \Delta r_{u,C} \cdot \left(r_{u,i} - \bar{r}_i\right)}{\sqrt{\sum_{u\in I_{i,C}} \left(\Delta r_{u,C}\right)^2 \sum_{u\in I_{i,C}} \left(r_{u,i} - \bar{r}_i\right)^2}}. \quad (5)$$

After calculating the similarity between each cluster *C* and the target item *i*, we choose item *i*'s 10 most similar clusters and take the items in these 10 most similar clusters as the candidates. From this process, the item clustering can help speed up the calculations of similarity as well as remove those irrelevant items. After obtaining the 10 item clusters most similar to the target item, the next thing to do is to choose its *k* most similar items from these most similar clusters based on Eq. 6:

$$\text{sim(i, j)} = \frac{\sum_{\text{u}\in I_{i,j}} \left(r_{u,i} - \bar{r}_i\right) \cdot \left(r_{u,j} - \bar{r}_j\right)}{\sqrt{\sum_{u\in I_{i,j}} \left(r_{u,i} - \bar{r}_i\right)^2 \sum_{u\in I_{i,j}} \left(r_{u,j} - \bar{r}_j\right)^2}}. \quad (6)$$

In predicting the rating of the target user u on target item *i*, *k* most similar items from the most similar item clusters are selected based on the similarity between them and the target item, and a weighted aggregation of the target user *u's* ratings on those *k* most similar items is used to generate predictions. Finally, the predicted rating of a target user *u* on a target item *i* (which is $\hat{r}_{u,i}$) is computed in Eq. 7, which is proposed in [3]:

$$\hat{\text{r}}_{\text{u,i}} = \bar{r}_i + \frac{\sum_{j=1}^{k} sim(i, j)\left(r_{u,j} - \bar{r}_j\right)}{\sum_{j=1}^{k} sim(i, j)}. \quad (7)$$

Experimental Design

We will conduct several experiments to examine the effectiveness of our item clustering based CF approach in terms of predicting accuracy, scalability with/without the procedure of data smoothing.

Dataset. We use the famous 100-K Movielens dataset which consists of 100,000 ratings (integers from 1 to 5) from 943 users on 1682 movies (items). Besides, each user has rated at least 20 movies and the density of this dataset is around 6%, which means data smoothing procedure will be executed on this dataset. Also, we use 20%, 50%, 70%, 80%, 90% of the whole dataset as the training set. Fig. 1 indicates that as the size of training data increases, our approach will get a better result, which does make sense because intuitively speaking, given more information about the dataset, the model can obtain a better 'understanding' of the dataset through training phases.

Evaluation Metric. We adopt the Mean Absolute Error (MAE) [2] which is a statistical accuracy metric used to measure the quality of the predictions:

$$\text{MAE} = \frac{\sum_{(u,i)\in T} \left(\hat{r}_{u,i} - r_{u,i}\right)}{|T|}. \tag{8}$$

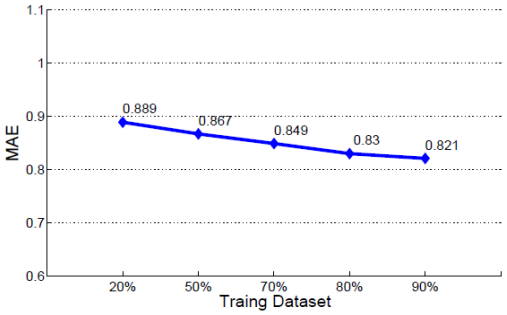

Fig.1: MAE on different sizes of training set

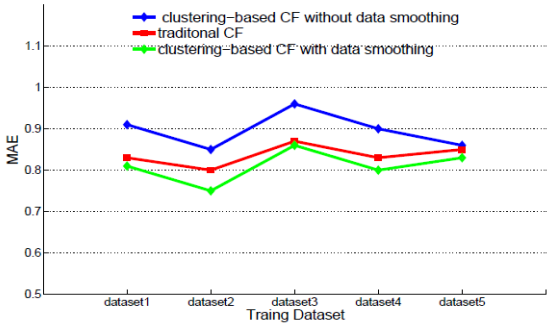

Fig.2: MAE of the three versions of CF on 5 datasets

where $\hat{r}_{u,i}$ is the predicted rating of user u on item i, $r_{u,i}$ is the true rating of user u on item i and T is the test set. In terms of MAE, a smaller value reflects a better performance of predicting models.

Experimental Results. We first use a 5-fold cross-validation to select $K = 80$ as the best value for the number of clusters (K in our case) from five candidate values $K = 20, 40, 60, 80, 100$. We also set the number of nearest neighbors to be $k = 20$ for each target item in advance. Thus in the following experiments of this project, K will be 80 and k will be 20.

We divide the whole dataset evenly into five equal-size sub datasets. We pick up only one of the five sub datasets, use it as the testing set and the remaining four as the training sets (e.g., 80% training and 20% testing). In this way, we obtain five different equal-size datasets so that we can compare the performance of three different CF approaches on each of them. Fig. 2 shows the MAE of traditional CF, item cluster based CF without data smoothing and item cluster based CF with data smoothing on the five datasets. The only difference between item cluster based CF with data smoothing and item cluster based CF without data smoothing is that the later skips the second step in section 2.1. It's obvious that our item cluster based CF with data smoothing has a better performance than item cluster based CF without data smoothing and also slightly outperforms the traditional CF. This is probably because the sparsity of the dataset has a negative influence on the other two approaches' effectiveness while our data-smoothing oriented item clustering based approach can be more robust to the sparsity of dataset due to the reason that we use data smoothing strategy after the procedure of item clustering.

The execution time of traditional CF and our item clustering based CF with data smoothing on 80% training set are shown in Fig. 3. For both methods, we record their execution time on each of the five equal-size 80% training sets above in Fig. 2 and display their average execution time separately in Fig. 3. With the increase of the pre-selected neighbors (notice that we may select one or more item clusters who are most similar to the target item according the similarities between them and the target item), the running time of our item clustering based CF will increase quickly while the running time of traditional CF remains high and unchanged, which indicates that our item clustering based CF has a much better scalability than traditional CF.

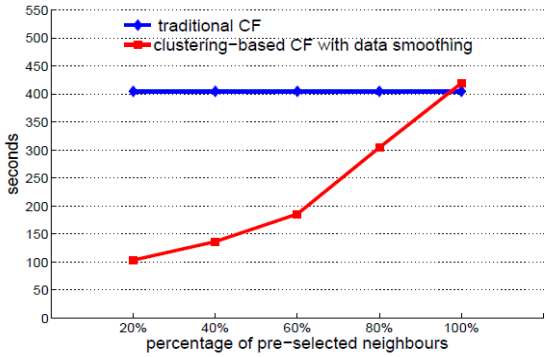

Fig. 3: running time of the two versions of CF with different percentage of pre-selected neighbours among the whole training set

Conclusion

In this paper, we propose a clustering based CF approach. We address the problem of scalability through item clustering. Besides, data smoothing strategy is adopted to address the missing-value problem (which potentially leads to the problem of sparsity). Experimental results indicate that our items clustering based CF approach does have an improvement over the traditional CF approach in terms of sparsity and scalability.

Acknowledgement

This work is supported by National Key Technology R&D Program (Grant No. 2012BAI34B01)

References

[1] John S Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence, pages 43–52. Morgan Kaufmann Publishers Inc.,1998.

[2] Jonathan L Herlocker, Joseph A Konstan, Loren G Terveen, and John T Riedl. Evaluating collaborative filtering recommender systems. ACM Transactions on Information Systems (TOIS), 22(1):5–53, 2004.

[3] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Analysis of recommendation algorithms for e-commerce. In Proceedings of the 2nd ACM conference on Electronic commerce, pages 158–167. ACM, 2000.

[4] Badrul M Sarwar, George Karypis, Joseph Konstan, and John Riedl. Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering. In Proceedings of the fifth international conference on computer and information technology, volume 1, 2002.

[5] Gui-Rong Xue, Chenxi Lin, Qiang Yang, WenSi Xi, Hua-Jun Zeng, Yong Yu, and Zheng Chen. Scalable collaborative filtering using cluster-based smoothing. In Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval, pages 114–121. ACM, 2005.