Item-Based Clustering CF Recommend Algorithm in MapReduce Framework

Liquan Han, Yuqiang Jiang
College of Computer Science and Engineering
Changchun University of Technology
Changchun, P.R.C
asehan@163.com; jiang.yq281@gmail.com

Qi Chu
School of Engineering & Applied Science
George Washington University
Washington, DC
qichu6@gwmail.gwu.edu

***Abstract*—Due to the overload of information, it's difficult to find the information we need from large amount of information. Thus, the recommendation algorithm appears. When the size of data is too large, the traditional collaborative filtering recommendation algorithm becomes unable to satisfy the requirement, such as out of memory. Therefore, this paper proposed a collaborative filtering algorithm based on feature clustering. Using the co-occurrence matrix replaces the similarity matrix and enhancing the efficiency of the algorithm. The item-based collaborative filtering algorithm was improved in order to make the algorithm run in parallel modes, such as running on Hadoop. So the algorithm can be easily improved performance by adding more nodes. After that, I clustered the rating data by the feature of user's, generated intra-group and inter-group recommendation results and mixed the two recommendation results. The experimental platform is Hadoop. And experiments show that: the complexity of matrix calculation has been reduced while the accuracy of recommendation results has been improved.**

Keywords—clustering, collaborative filtering, recommend algorithm, mapreduce, hadoop

I. INTRODUCTION

With the rise of the Internet, especially the mobile Internet, the scale of e-commerce has been expanding, and the number of goods in the database has increased dramatically. This makes it difficult to find the information users need in the vast amount of data. Recommend system has become the best solutions to solve this problem. According to the user's habits and customs and needs, recommend system provide the personalized recommendation services. In a variety of recommend algorithms, the recommendation based on collaborative filtering is the most successful one undoubtedly. Recommend algorithm based on collaborative filtering is divided into user-based recommendation and item-based. In order to improve the performance of the algorithm, in [1], an item-based collaborative filtering algorithm has been improved to run in parallel on Hadoop. Reference [2] used the co-occurrence matrix instead of the similarity matrix, and the accuracy of the results didn’t reduced.

This paper improved the item-based collaborative filtering algorithm by the Map-Reduce distributed computing framework. At the same time using the method of clustering, making the co-occurrence matrix closer to the real similarity matrix. Finally, improving the recommended accuracy.

II. RELATED WORK

Collaborative filtering algorithm is proposed by D. Goldberg[3] in 1992 and successfully applied in the study of e-mail recommendation system in the recommended algorithm [4]. The core idea is that if the interest, hobbies, habits among users are similar, the things that those users like are the similar. Therefore, similar users can be used to predict the rating of the item that the target user does not rated.

A. Map Reduce and HDFS

Apache Hadoop is an open-source software framework built from commodity hardware that used for distributed storage and distributed processing of very large data sets on computer clusters. MapReduce[5] is the heart of Hadoop. It is a programming paradigm that allows massive scalability to go across hundreds or thousands of servers in a Hadoop cluster. Hadoop Distributed File System (HDFS)[6] is a distributed file system that provides high-throughput access to application data. Data in a Hadoop cluster is broken down into smaller pieces (called blocks) and distributed throughout the cluster. In this way, the map and reduced functions can be executed on smaller subsets of your larger data sets, and this provides the scalability that is required for big data processing.

B. Item-based Collaborative Filtering Algorithm

Item-based collaborative filtering recommendation algorithm refers to match a user-rated item to a similar item, and then places this similar item in the user's recommended list. The simple thing is to recommend the item to the user which is similar to his favorite items as before.

1. Generate the user's rating matrix: According to the same user's rating data, for the user u, a rating vector R_u may be generated. By combining all user’s rating vector, the original user-item rating matrix R_{m*n} is generated.

2. Calculate the similarity between items: This step is the key point of the algorithm. Common similarity calculation methods are [7]: cosine similarity, modified cosine similarity, Pearson correlation coefficient.

Pearson correlation coefficient: For the item *i* and item *j*, the rating vectors of all users' are R_i and R_j.

$$\rho_{i,j} = \frac{\mathrm{cov}(R_i, R_j)}{\sigma R_i \cdot \sigma R_j} = \frac{E\left(R_i\right)\cdot E\left(R_j\right)}{\sqrt{E\left(R_i^2\right) - E^2\left(R_i^2\right)}\cdot\sqrt{E\left(R_j^2\right) - E^2\left(R_j^2\right)}} \tag{1}$$

After center the data($r_{i,k} = r_{i,k} - \bar{r}_i$), (1) can be simplified into :

$$\rho_{i,j} = \frac{E(R_i \cdot R_j)}{E(R_i^2)\cdot E(R_j^2)} = \frac{\frac{1}{m}\sum_{k=1}^{m} r_{i,k}\cdot r_{j,k}}{\sqrt{\frac{1}{m}\sum_{k=1}^{m} r_{i,k}^2}\cdot\sqrt{\frac{1}{m}\sum_{k=1}^{m} r_{j,k}^2}} \tag{2}$$

By calculating all the similarity between items, the similarity matrix is generated. This matrix has *m* rows and *m* columns.

3. Generate the recommendation list for user: According to the similarity matrix of the items and the user's rating matrix

For user *u*, the predictive rating is calculated by (3):

$$p_{u,i} = \overline{R_i} + \frac{\sum_{j\in S(i)} sim(i,j)\cdot(r_{u,j} - \overline{R_j})}{\sum_{j\in S(i)} sim(i,j)} \tag{3}$$

S(i) represented the set of items which is similar to item *i*. $\overline{R_i}$ represented the average rating of all users to item *i*.

Although the traditional collaborative filtering algorithm is widely used, this algorithm still has some short comings, such as scalability issues, interest-change issues, and so on. Common solutions[8] to solve the scalability issues include dimension reduction, clustering, data set reduction, etc.

III. Algorithms Model And Implementation

A. Improved Item-based Collaborative Filtering Algorithm

Using the MapReduce framework to implement the collaborative filtering algorithm is a way to solve the scalability problem. Reference [2] use the co-occurrence matrix instead of the similarity matrix. Therefore, the complexity has been reduced effectively. And the accuracy of the result has not been reduced. To implement the algorithm in MapReduce framework, the modification is shown as below [9]. Suppose there are m users and n items.

1. Generate the rating matrix of user: This part has not been modified.

2. Generate the co-occurrence matrix: According to the rating vectors, counting the times of two item is rated by same user. $S_{i,j}$ = *count(i, j)*.The matrix S_{n*m} consisting of all the $S_{i,j}$ is the co-occurrence matrix. Usually, S_{n*m} is a Sparse Symmetric Matrix. For MapReduce process, the input data of map is the rating vector.

3. Matrix multiplication: The result element in result matrix is $P'_{i,u}$. For one user u, the matrix multiplication process shows as follow.

$$\begin{array}{ccccccc} \mathrm{I}_1 & \cdots & \mathrm{I}_\mathrm{m} & & \mathrm{R}_\mathrm{u} & & \mathrm{U}'_\mathrm{u} \\ \left[\begin{matrix} S_{11} \\ \vdots \\ S_{m1} \end{matrix}\right. & \begin{matrix} \cdots \\ \ddots \\ \cdots \end{matrix} & \left.\begin{matrix} S_{1m} \\ \vdots \\ S_{mm} \end{matrix}\right] & \times & \begin{matrix} r_{u,i} \\ \vdots \\ r_{u,m} \end{matrix} & = & \begin{matrix} P'_{1,u} \\ \vdots \\ P'_{i,u} \end{matrix} \end{array}$$

$r_{u,i}$ represents the rating of movie *i* from user u, $S_{i,j}$ represents the co-occurrence times of item *i* and item j. $P'_{i,u}$ is the numerator of recommended result.

Final recommended result is calculated as (4).

$$P''_{i,u} = \frac{\sum_{k\in W_u} S_{ik}\cdot r_{u,k}}{\sum_{k\in W_u} S_{ik}} = \frac{P'_{i,u}}{\sum_{k\in W_u} S_{ik}} \tag{4}$$

S_{ik} represents the elements in the row *i* and column *j* in co-occurrence matrix. W_u represents the set of movies which user *u* has rated. For user u, $P''_{u,i}$ is the predicted rating of item *i*. In (4), the numerator is the result of the row *i* in the co-occurrence matrix multiplies the rating vector of user *u*. Actually, calculating $P''_{u,i}$ only needs the rating data of user and the data of row *i* in the co-occurrence matrix. So use the item ID as the key, and use the data of this row as the value. In this way, the predicted value can be calculated in one MapReduce process. And target users' rating data is stored in HDFS. When all the map data is loaded, read the rating data of user from the file in HDFS row by row.

B. Feature of Users

By statistics, we can get a lot of different features from the user data of rating history. We can cluster users by these features. Compared to the rating history, the calculation of clustering is much smaller. So clustering is much easier. The features have been divided into three categories.

- Time related:

The time period of user rated on: The day is divided into four parts: morning, afternoon, evening, early morning. According to user ratings' time data, count the times of the four parts and then calculate the proportion of the four parts.

Time difference between rate and release time: This feature is represented how interested the user is in the new movie. count the difference between the earliest time and rate time. It is close to the difference between releasing time and rate time. The mean and variance represent this feature and calculate as shown in (5) and (6).

$$avg_e(u) = \frac{1}{m}\cdot\sum_{i\in W_u}(t_{u,i} - \min(T_i)) \tag{5}$$

$$D_e(y) = \frac{1}{m}\cdot\sum_{i\in W_u}(t_{u,i} - avg_e(y))^2 \tag{6}$$

$t_{u,i}$ represents the time of rating of movie *i* by user *u*. T_i represents the times of all the ratings.

- Rating related:

The mean and variance of all ratings: By counting all the rating data of a user, the mean and variance can be easily calculated. The mean and variance is represented by $avg_{all}(u)$ and $D_{all}(u)$.

The mean and variance in each genre: Each genre is counted separately. A movie may have multiple different genres. To deal with this situation, the rating is weighted to each genre. For one movie, the weight in each genre is the same. The means and variances are calculated as follow.

$$avg_g(k,u)=\frac{\sum_{i\in Wg_{k,u}}(\frac{1}{gc_i}\cdot r_{u,i})}{\sum_{i\in Wg_{k,u}}(\frac{1}{gc_i})} \qquad (7)$$

$$D_g(k,u)=\frac{\sum_{i\in Wg_{k,u}}(\frac{1}{gc_u}\cdot(r_{u,i}-avg_{g1}(k,u)))}{\sum_{i\in Wg_{k,u}}(\frac{1}{gc_i})} \qquad (8)$$

$W_{gk,u}$ represents the set of movies which user u has rated and the movie's genre include k. gc_i represents the count of genres of movie i. $r_{u,i}$ represents the rating of movie i from user u.

The difference between total mean and genre's mean: Each genre is calculated separately.

- Count related:

Total count of ratings:The total count of rating from one user.

The proportion of count of each genre:The calculate process is same as the rating. For each genre, calculate the sum of the weighted count from each movie. The counts of each genre are calculated as shown in the (9)

$$count_g(u,k)=\sum_{i\in W_{gk,u}}(\frac{1}{gc_i}) \qquad (9)$$

C. K-means Clustering

Clustering[10], also known as unsupervised learning assigns items to a group. So the items of same group are more similar than different groups. K-means is a classical clustering algorithm based on partition. The N items are partitioned into k unrelated subsets. So that the items in the same subset are as close as possible. And the subset S_i has k items. The modification is re-selecting the cluster centers. The new centers must be a real object.

D. Implementation

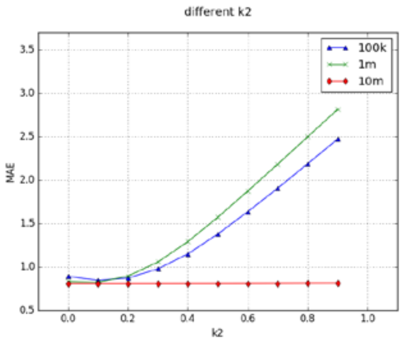

Fig. 1. MAE with different k_2

The clustering process is added to the recommended process. First of all, cluster user by the features. Then calculate the predicted rating of intra-group and inter-group and mix the two ratings by a ratio. The mixed rating is the final predicted rating.

- Combine the rating data of same user to a single line

The format of original data is < uID, itemID, rating, rating Time>. Combine the ratings from same user, generate the rating vector :<userID, list (itemID, rating) >

- Generate the feature data

According to the rating vector, calculate the features of users< userID, list(feaature1, ⋯ feture85) >. The count of features is 85.

- Clustering user by the feature data

After clustering, mark the center user and others. The rating data is split by the mark data. If there are n clusters, n sets of data will be generated. For the cluster i, generate two files. The first file is the rating data of the users in cluster i. The second file is the rating data of the other n-1 centers.

- Calculate the intra-group recommended result

Use the rating data of the user(file 1) to generate the co-occurrence matrix. And use the test data to generate the ratings matrix. Then run the algorithm and get the result U'_u.

- Calculate the inter-group recommended result

Use the rating data of other cluster centers (file2) to generate the co-occurrence matrix. The user matrix is same as intra-group. Then run the algorithm and get the result U''_u.

- Mix the two recommended result

Mix the two predicted results as the finally recommended result. It can be easily calculated by (18):

$$U_u = k_1\cdot U'_u + k_2\cdot U''_u \qquad (10)$$

In addition, $k_1+k_2=1$.

E. Parameter Optimization

Due to that $k_1+k_2=1$, When k_2 obtain the optimal value, the best k_1 has been found. So only the value of k_2 changed. The lower MAE means the higher accuracy. So the MAES has been calculated with different k_2 and then the best k_2 is found.

From Figure 1 we can see that when the value of k_2 between 0.1 and 0.2, the value of MAE is lower than others.

IV. EXPERIENCE

A. Data

In this paper, the experimental data is film rating data called MovieLens. This data set includes ratings, rating time, movie name, movie genre, etc. There is a script in the data set. It can split the ratings data for five-fold cross-validation of rating predictions. There are three data sets used: ml-100k, ml-1m, ml-10m. ml-100k: This data set contains 100000 ratings from 943 users on 1682 movies. ml-1m: This data set contains 1000209 ratings from 6040users on 3900movies. ml-10m: This data set contains 10000054 ratings and 95580 tags applied to 10681 movies by 71567 users.

B. Speedup

The speedup has been used to evaluate the performance of the algorithm. If the algorithm is scalable, the speedup has a linear relation with the numbers of nodes with the data size fixed.

From the figure 2, only part of the speedup increased relative linearly. Increasing the number of nodes does improve the efficiency of the algorithm. Although in theory, the acceleration of the speedup should be relative linearly. But in fact it is impossible. When the overhead of the communication reaches the upper limit of the bandwidth, the speedup ratio drops.

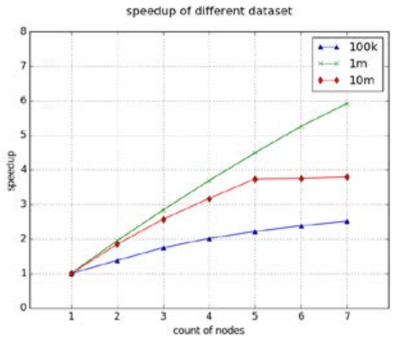

Fig. 2. Speedup

C. MAE

The mean absolute error (MAE) [11] was selected as the evaluation indicator of algorithm accuracy.

For the three data sets above, each data set has been spited to five sets. Run test on the five set, the result is shown as figure 3. From figure 3, the MAE of improved algorithm decreases by 0.02 relative to the original algorithm. After clustering, the MAE has reduced about 0.05. This illustrates that replace the similarity matrix with the co-occurrence matrix, clustering and split data can improve the algorithm accuracy significantly.

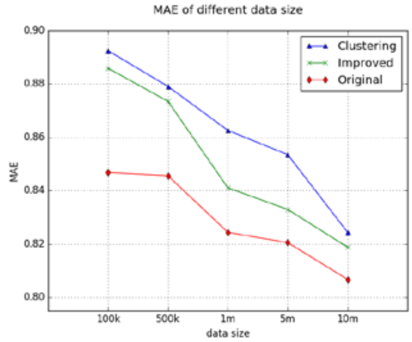

Fig. 3. MAE of different data size

V. Conclusions and Future Works

This paper improved the collaborative filtering algorithm, and the similarity matrix has been replaced by a co-occurrence matrix. After this replacement operation, the process of calculation becomes simple. By the clustering, the similarity matrix of the current matrix is close to the real one without adding large amount of data. Then mix the result from the in cluster data and different cluster centers. This can avoid the problems that when the habit of user has changed. According to the experimental results, it can be seen that using the accuracy of the collaborative filtering algorithm is improved. After the clustering process, the accuracy can be further improved. This shows that the main method to improve accuracy is to make the similarity matrix closer to the true value. The co-occurrence matrix used in this paper can be approximate to the true similarity matrix. By adding nodes, the efficiency of the algorithm can be improved effectively. However, there are still cold start problems. The further work is solving this problem.

References

[1] Cun Y, Genlin J. Design and Implementation of Item-Based Parallel Collaborative Filtering Algorithm[J]. Journal of Nanjing Normal University (Natural Science Edition), 2014, 1: 014.

[2] Xiao Q, Qinghua Z, Hua J. Design and Implementation of Distributed Collaborative Filtering Algorithm on Hadoop[J]. New Technology of Library & Information Service, 2013(1):83-89.

[3] Goldberg D. Using collaborative filtering to weave an information tapestry[J]. Communications of the Acm, 1992, 35(12):61-70.

[4] Luo X, Ouyang Y X, Xiong Z, et al. The effect of similarity support in k-nearest-neighborhood based collaborative filtering[J]. Jisuanji Xuebao(Chinese Journal of Computers), 2010, 33(8): 1437-1445.

[5] Dean J, Ghemawat S. MapReduce: Simplified Data Processing on Large Clusters.[C]// Conference on Symposium on Opearting Systems Design & Implementation. USENIX Association, 2004:107-113.

[6] Shvachko K, Kuang H, Radia S, et al. The Hadoop Distributed File System[C]// IEEE, Symposium on MASS Storage Systems and Technologies. IEEE Computer Society, 2010:1-10.

[7] Huang Z, Zeng D, Chen H. A Comparison of Collaborative-Filtering Recommendation Algorithms for E-commerce[J]. Intelligent Systems IEEE, 2007, 22(5):68-78.

[8] Li C. Research on the bottleneck problems of collaborative filtering in e-commerce recommender system[D]. PhD thesis, Hefei University of Technology, Hefei, China, 2009.

[9] Tang G, Zhang W. Development and Analysis of Co- word Analysis Method at Home and Abroad[J]. Library & Information Service, 20144, 58(22):138-145.

[10] Sun J, Ming H U, Zhao J. An optimal algorithm for K-means initial clustering center selection[J]. Journal of Changchun University of Technology, 2016, 37(1):25-29.

[11] Zhao Z D, Shang M S. User-Based Collaborative-Filtering Recommendation Algorithms on Hadoop[C]// International Conference on Knowledge Discovery and Data Mining, Wkdd 2010, Phuket, Thailand, 9-10 January. 2010:478-481.