IOP Conf. Series: Materials Science and Engineering **359** (2018) 012008 doi:10.1088/1757-899X/359/1/012008

Layout Study and Application of Mobile App Recommendation Approach Based On Spark Streaming Framework

H.T. Wang[1], T.T. Chen[2], C Yan[2] and H Pan[2]
[1] Network Information Center, Wuhan University of Technology , Hubei Wuhan 430070, China
[2] School of Computer Science and Technology, Wuhan University of Technology , Hubei Wuhan 430070,China

Abstract. For App recommended areas of mobile phone software, made while using conduct App application recommended combined weighted Slope One algorithm collaborative filtering algorithm items based on further improvement of the traditional collaborative filtering algorithm in cold start, data matrix sparseness and other issues, will recommend Spark stasis parallel algorithm platform, the introduction of real-time streaming streaming real-time computing framework to improve real-time software applications recommended.

1. Introduction

Nowadays, data mining in big data has been applied to all walks of life and become a new subject of internet. Based on the mobile terminal equipment, this paper puts forward a discussion on the recommended method for mobile application software. In the field of information recommendation, due to the huge data, the problem of overload of information against big data becomes more and more obvious, and it is difficult to select recommendation information suitable for individual users from hundreds of millions of massive data. To solve this problem, we propose the introduction of big data computing platform Spark, to handle large amounts of data is recommended, as well as the introduction of Spark Streaming real-time computing framework to improve inefficiencies recommend Recommended present in the system. This paper collaborative filtering recommendation algorithm and the Slope One algorithm, and for those existing algorithm problems are also discussed: the introduction of implicit information feedback, to make the non-score behavior of the user quantified as score values, by using predict initial scoring matrix by Slope One recommendation algorithm Grading, solve the data sparseness problem of the recommended algorithm. Meanwhile make the improved recommendation algorithm parallelization running on the Spark platform, realized mobile phone App application under the big data real-time recommendations. This article grabs Google application store and Baidu mobile application market of mobile phone App information and application rating data as the experimental sample data sets, has done the tests comparing the improved recommendation algorithm and traditional collaborative filtering algorithm, then calculate the mean absolute error of prediction score, and recommended Accuracy and recall rate. Finally, the experimental results obtained by the introduction of weighting Slope One algorithm based on the pre-filled article collaborative filtering algorithm can achieve better recommendations.

2. Item-based collaborative filtering and Slope One algorithms

2.1 Principle of item-based collaborative filtering recommendation

Item-based collaborative filtering (Item-based CF) recommendation is to calculate the degree of similarity of items and then select the most similar item in accordance with information of items for its recommendation to users, which users are interested in; and it is schematically shown in Figure 1.

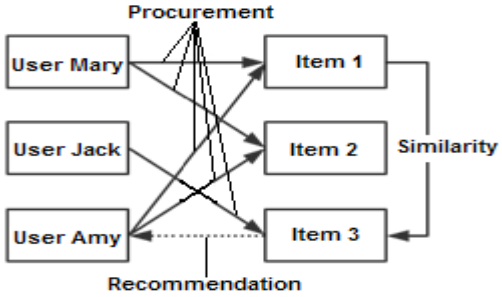


Figure 1. Schematic diagram of item-based recommendation

Users A and B simultaneously downloaded Softwares X and Z and Users C and B downloaded Softwares X and Y, respectively. It is assumed that downloading and non-downloading are scored as 5 and 0 and eigenvectors for Softwares X, Y and Z are defined as (5, 5, 5), (0, 5, 0) and (5, 5, 0), respectively; and it is assumed that the calculated Pearson Correlation Coefficients between Softwares X and Y and X and Z are $\sqrt{(1/3)}$ and $\sqrt{(2/3)}$, respectively, Item X is more similar to Item Z rather than Item Y. Thus, the system prefers to recommend Item Z to User C while he or she scores Item X or has other positive behaviors to Item X.

2.2 Limitations of recommendation of mobile Apps for traditional collaborative filtering algorithms

2.2.1 Data sparsity of the rating matrix: in accordance with the software applications with recommendation modules in the current market, the number of applications is far more than the number of scoring applications; and some users do not present any comment due to their own habits. As for more and more items, the matrix shall be very sparse while a user-item rating matrix is necessarily built; subsequently, the similarity calculation is more complicated and the results are less accurate.

2.2.2 Cold start: as for an App recommendation application, cold start means nay system recommendation may not be performed for a new user without any historical behavior; similarly, while a new mobile App is never downloaded, it may also be recommended t users.

2.2.3 Real-time capability: Due to the rapid updating of mobile Apps, new data is constantly increasing and users may only download an App for a certain period of time so that more timely and efficient real-time recommendation shall be necessary. Because calculation of a lot of data needs more time for a traditional collaborative filtering algorithm, regular updating of user data is usually necessary to update recommendations; however, this recommendation method is inefficient and may not meet the needs of users' getting the latest recommendations.

2.3 Principle and procedures of Slope One: The core concept of Slope One (or ascent algorithm) is also collaborative filtering, which is to predict scores of non-scoring Apps through calculation of deviations between mobile Apps and then in accordance with the similar Apps scored by users, whose deviations are smaller. It is assumed that there's sort of a straight-line relationship ($y = ax + b$) for the preference between two items. For modification of over-fitting while calculating the relevance, it is simplified as:

$$y = x + b \tag{1}$$

Determination of Parameter b is a preprocessing procedure of Slope One. It is assumed that each user's scoring vector set for Items A and B as $S = \{< A_1, B_1 >, < A_2, B_2 >, \ldots\ldots, < A_n, B_n >\}$, where: $A_1, A_2, \ldots\ldots,$ and A_n represent scores of Users 1, 2, ……, and n for Item A, respectively. Similarly, $B_1, B_2, \ldots\ldots,$ and B_n represent scores of Users 1, 2, ……, and n for Item B, respectively. Thus, the score difference (b) between Items A and B is estimated as:

$$b_{AB} = \frac{\sum_{i=1}^{n} B_i - A_i}{n} \quad (2)$$

Where: n represents the number of those users who evaluated Items A and B at the same time. b_{AB} represents the mean preference deviation of Items A and B.
Then, Users may predict scores of Items A and B.
If User u wants to predict scores of Item i, his or her scoring set ({User x, User y, User z,…}) shall be first selected and then his or her scored item set ({Item x, Item y, Item z,…}) shall be selected so as to form his or her scoring matrix in smaller dimensions; thus, his or her predicted scores (P(uA)) for Item A is expressed as:

$$P(uA) = \frac{\sum_{i \in P(u)} (b_{A,i} + r_{u,i})}{k} \quad (3)$$

Where: P(u) represents existing scores of User u; k represents the number of elements of Set P(u); and $r_{u,i}$ represents the User u's scores for Items i.
Because each App is independently scored, the weighted-calculation method (namely Weighted Slope One) is applied to balance effects of scores of various items, where the number (w_{AB}) of users who evaluated two items together with User u is regarded as the weight. Eq. (3) is modified as:

$$P(uA) = \frac{\sum_{i \in P(u)} (b_{A,i} - r_{u,i}) \cdot w_{AB}}{\sum_{i \in P(u)} w_{AB}} \quad (4)$$

In accordance with Slope One, scoring records of users who scored it are only involved into calculation while a certain item is predicted; thus, the results of calculating may be affected greatly for a smaller number of users; on the other hand, the collaborative filtering recommendation cold start may be relieved to some extent. Its workflow processes are shown in Figure 2.

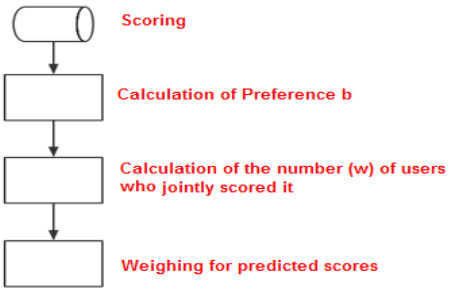


Figure 2. Slope One recommendation workflow processes

3. Improved scheme

Although the collaborative filtering recommendation algorithm is one most popular method, it still has a number of limitations. Our improvements are as follows:
Collecting users' implicit feedback data: the so-called implicit data are preference information which may be gained by means of observation of users' operation records rather than scores users intuitively present. While any user clicks, browses, stores, search or download any information in the recommendation system, these behaviors are quantified as the corresponding scores, which are filled into the user-item matrix to make up the defect (too sparse rating data).

Table 1. Quantization of users' implicit feedback scores

Behavior	Corresponding scores
Click	2
search	3
View details	3
Share	4
Download	5

In accordance with analysis of actual situations, while an App is scored below 2 scores by a user, this suggests that the App is not satisfied; on the other hand, while he or she presents any behavior in Table 1, this suggests he or she is interested in this App; thus, the scoring threshold is 2 scores. While a user performs various behaviors over the same time period, the corresponding maximum score shall be selected as the final score.

For improving those issues such as cold start due to being short of data, his or her downloaded mobile Apps as the initial data for any new user, which are scored as 5 scores for further recommendation. While information of any user's downloaded Apps may not be gained, recommendation may only be performed by means of those by means of Apps in Top Charts. In view of any new App, while scoring data of these Apps are not recorded, recommendation may be performed by searching keywords of Apps.

Next, the simple Slope One may be utilized to gain better results in case of relatively sparse behavior records and scoring data of users; thus, the user-item scoring matrix may be filled first with scores predicted by means of Slope One and then degree of similarity of items may be calculated; finally, the most similar item for recommendation may be determined in accordance with the scoring records of the recommended users.

For improvement of recommendation delay, the Spark Streaming real-time calculating framework was applied here, online and offline calculation is performed first to data; and Spark Streaming real-time receives scoring and implicit data from users. The recommendation model may be trained by using a lot of offline data in the recommendation engine so that these data may be processed for recommendation; moreover, the modified recommendation algorithm is implemented in parallel in the Spark platform; and recommendation results appear finally in the recommendation system. The process decomposition of offline and real-time data is schematically shown in Figure 3.

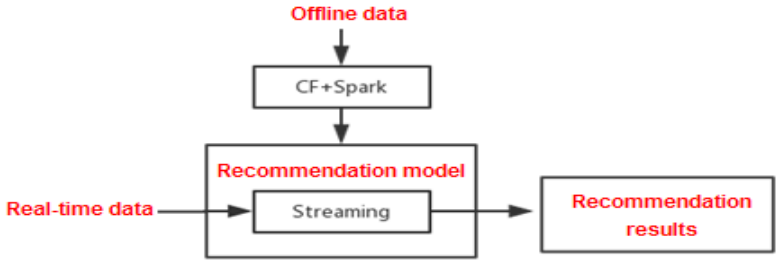


Figure 3. Schematic diagram of processing offline and online data

The entire modified recommendation flow chart is shown as follows:

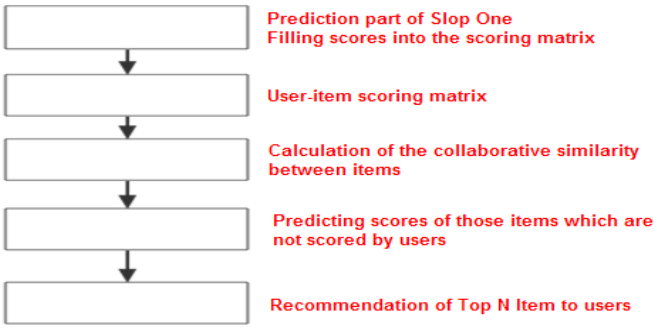


Figure 4. Recommendation flow chart modified by integrating Slope One

4. Experimental analysis of performances of the algorithm

4.1 Preparation

Recommendation of mobile Apps is faced with continuous and dramatic growth of user behavior data and calculation may be more intensive under the standalone mode. For optimization and future expansion of recommendation performances, recommendation calculation is arranged in the distributed Spark cluster to minimize calculation of a lot of data.

Data here were scraped from websites by means of the crawler developed with the python language tools, which come from the Baidu mobile application market and Google Play, respectively. More than 400,000 scoring data were used for the recommendation algorithm in Spark, which were uploaded to the HDFS file system in advance; and some data in the data set are shown in Table 2.

Table 2. App rating data set

User_id	App_id	Rating
70	417	5
305	15	1
201	370	1
57	409	4
125	173	5

The experimental PC is shown in Table 3.

Table 3. PC configurations

Operation system	CentOS 7
Cluster environment	Hadoop 2.4
Spark platform	Spark 2.0.0
IDE	IntelliJ IDEA2016
Single Node Memory	4G
Node number	2 (1 Master and 1 Worker)
Total host memory	32G

4.2 Experimental evaluation criteria

There are no uniform evaluation criteria for the recommendation algorithms and the recommendation effects of each system may be measured by means of various methods, some of which are measured by means of the user feedback mode or calculated recall and so on. Generally, the mean error between the scores predicted based on the test sample data set and the actual scores is regarded as the basis for determining the recommendation precision. The real-time capability is also required for some recommendation systems; thus, real-time recommendation performances are also a part of assessment of recommendation effects.

In general, the evaluation indexes for a recommendation system include Mean Absolute and Root Mean Square Errors (MAE and RSME), which are as follows:

$$\mathrm{MAE} = \frac{\sum_{i=1}^{m} |p_i - r_i|}{k} \tag{5}$$

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{m} (p_i - r_i)^2}{k}} \tag{6}$$

Where: p_i represents the actual scores of Item i from users; r_i represents the predicted scores of Item i from users; and k represents the number of items scored by users.

Smaller MAE or RMSE indicates that the predicted scores more approach the actual scores. In accordance with our experiments, the difference between the predicted and actual RMSEs is larger; thus, a higher recommendation precision is required.

In view of implementation of the entire recommendation, two concepts (namely precision and recall) are put forward, which are expressed as follows:

$$\mathrm{Precision} = \frac{\sum_u |R(u) \cap P(u)|}{\sum_u |P(u)|} \tag{7}$$

$$\text{Recall} = \frac{\sum_u |R(u) \cap P(u)|}{\sum_u |R(u)|} \quad (8)$$

Where: R(u) represents the set of items actually operated by User u; and P(u) represents the set of items recommended to users. In view of each recommendation scene, its precision and recall are different. As for recommendation of the mobile APPs, the precision represents the users' actual download rate of recommended Apps and Recall represents the percentage of recommended Apps in the software actually downloaded by users.

4.3 Experimental design and results analysis

*4.3.1 Absolutely flat All wrong MAE poor.*Indicates that a smaller MAE represents a lower difference between the actual and predicted scores. The data set was divided into two parts (namely the training and measurement sets) in accordance with the ratio (8:2) during our experiments. The final value shall be finally stabilized to a certain value after a few trainings.

The crawled App scoring data set was only utilized for comparison of Slope One and Weighted Slope One. The calculated MAEs are given in Table 4.

Table 4. Comparison of Slope One and Weighted Slope One

	Weighted Slope One	Slope One
MAE	0.845	0.877

The above MAEs indicate that the weighted Slope One has a lower predicted score error so that it shall be selected to fill some predicted scores into the original scoring matrix; and the recommendation algorithm shall be modified based on the weighted Slope One.

Experimental results for the modified recommendation algorithm are as follows:

Table 5. Comparison of MAEs for the recommendation algorithm under various data sets

Data Set	**Item-based CF**	**Item-based CF+slope one**
Movielens 100k	0.8325	0.812
Movielens 1M	0.8075	0.799
App rating	0.852	0.84

The comparison results are shown in Figures 5 and Figures 6, which indicate that:

- The larger the data set or the number of the training set is, the smaller MAE is and the better the recommendation effects are.
- 2) The minimum MAEs for the optimization and item-based CF algorithms are 0.792 and 0.8075, respectively. While data are fewer and the matrix is sparse, the precision of the modified algorithm is higher.
- 3) Due to our data acquired from 27,775 users and scoring information got from only 1691 Apps, the scoring matrix is too sparse so that the effects for prediction of App scores shall be the weakest.

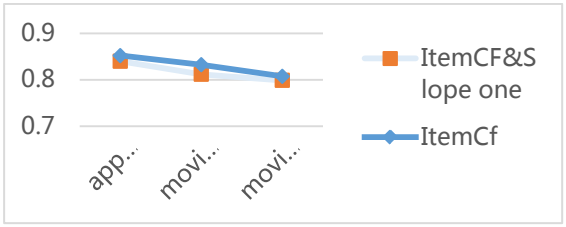


Figure 5. Comparison line chart for various algorithms under various data sets

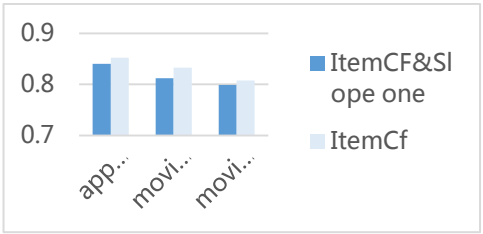


Figure 6. Comparison bar chart for various algorithms under various data sets

4.3.2 Recall and Precision. As for recommendation of mobile Apps, the precision is represented by means of the users' download rate in the recommendation results, which may be regarded as the

number of Apps scored as 5 scores by users in the recommendation table based on the implicit feedback quantization scores; on the other hand, the recall may be represented by the percentage of the Apps scored as 5 scores by all users in the mobile App software scoring table. Thus, higher precision and recall necessarily depend on setting a rational recommendation number (N), which is set as 5, 10, 15, 20 and 25 for 5 experimental groups, respectively; and the rate of the training and measurement sets is 8:2. The data set is movielens 1M scoring.

During the experimental processes, the integer (5 scores) seldom occurs because calculation results are not integers. For ensuring the experimental effects, the predicted scores (above 4.8 scores) may regarded as the standards for calculation of the precision and recall; and the corresponding precision and recall results are shown in Table 6 and the line chart is presented in Figure 7.

Table 6. Precision and recall results under various N values

Evaluation value \N	5	10	15	20	25
ItemCF precision	24.32%	19.38%	18.77%	17.65%	17.01%
ItemCF recall	12.08%	13.31%	16.41%	18.59%	22.7%
Precision of the modified algorithm	24.97%	21.87%	19.36%	18.03%	17.55%
Recall of the modified algorithm	13.51%	14.22%	17.86%	19.57%	23.01%

Figure 7 reveals the precision and recall trends for the general item-based CF algorithm and the modified CF algorithm integrated with Slope One under various N values, where the precision and recall are slightly higher for the latter. Concepts of precision and recall indicate that: the recall may fall while the precision rises and they tends to the balance within a certain interval; the precision is among 16-25% and tends to decline along with the growth of the recommendation number (N) while the precision is only taken into account; namely, a larger recommendation number leads to falling of the percentage of Apps liked by users; on the other hand, the recall rises among 12-23%;and such case indicates the users download these favorable Apps in all likelihood; thus, recommendation Apps account for more Apps downloaded by users. The optimal N values for curve intersections are among 15-20.

The modified recommendation algorithm is parallelized in the Spark engine and the scoring data are updated by means of the Streaming calculation framework to perform real-time calculation of big data. Analysis was carried out to those aspects such as recommendation results change, evaluation of recommendation effects and real-time processing time by means of simulation of streaming data for recommendation results, respectively.

By real-time inserting 1,000 new scoring data in batches in the netcat terminal, the batch duration for the Streaming context receiving streaming data and the Top N are set as 20s and 15, respectively. The operation period is set as 1 hour; and it is predicted that real-time data will be received in 180 batches and one calculation period is 20s.

The data calculation time curve is shown in Figure 8.

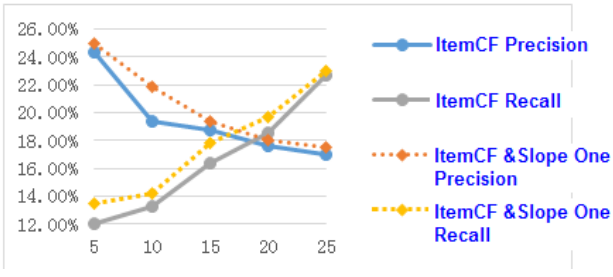


Figure 7. Recall and precision

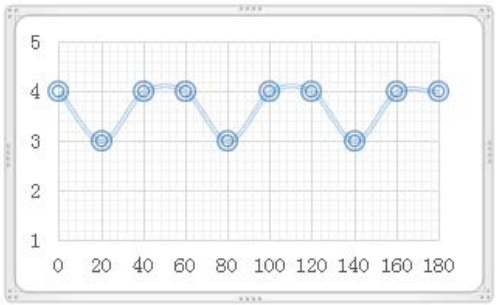
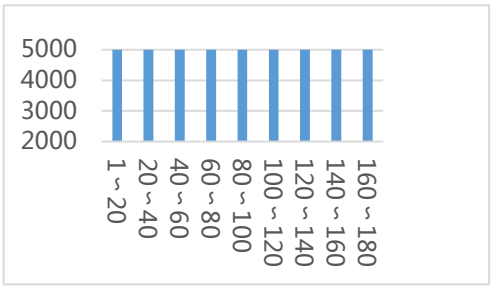


Figure 8. Training time consumption curve

The horizontal and vertical coordinate axes (Figure 8) are batches and training time consumption, respectively. The period for each batch is 20s and the mean training time consumption is 3.7s.
The receiving data streaming bars are shown in Figure 9.



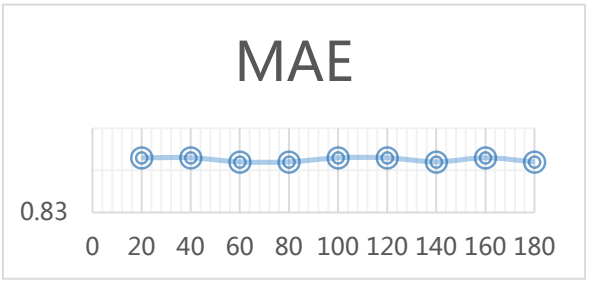


Figure 9. receiving data streaming bars **Figure 10**. MAE changes

Streaming contexts receive new scoring data real-time input per batch, to which calculation is carried out by means of streaming. Figure 9 indicates that the statistical unit is 20 batches and 5,000 new data were received in total; moreover, while about 1,000 data input in 5 batches are selected stochastically and those data input in other batches are 0 from the above data, the entire process may not be broken down or restarted for calculation.
MAE changes are shown in Figure 10 after inputting the real-time data. Calculated MAEs indicate that they do not change greatly, which are among 0.84-0.845; thus, calculation of new input data may not greatly change recommendation errors.

5. Conclusions

Based on study and application of recommendation of mobile Apps in the Spark Streaming framework, discussion was carried out here by focusing three aspects (namely those recommendation algorithms (such as Item-based CF and Slope One algorithms), Spark real-time streaming calculation in the distributed big data platform and mobile App Store clients based on the mobile devices.
As for the recommendation algorithm, evaluation experiments were carried out for verification of the effectiveness of the modified CF recommendation algorithm; and MAEs for Item-based CF and Slope One algorithms were calculated under various data sets, respectively; and the CF algorithm was then performed in the distributed big data platform Spark where real-time processing may be quickened and whose fault tolerance and expandability are more intensive than those of Hadoop (as a similar big data processing platform).

6. References

[1] Hu Yuxiang 2015 *Design and implementation of recommendation system based on Spark [D]* (Zhejiang University)
[2] Chen Yuzhao 2014 *Study and implementation of Personalized Recommendation based on Big Data [D]* (Xidian University)
[3] Guo Xiaobo, Zhao Shuliang and Niu Dongpan 2015 *A novel combination recommendation method for solving sparse and cold start problems [J.* (Journal of University of Science and Technology of China) 45(10):804-812
[4] Li Xue, Zuo Wanli, Hao Fengling, et al 2009 *An Improved Item-Based Collaborative Filtering Recommendation Algorithm [C]*
[5] Zhang Xiande 2016 *Research and Implementation of Real Time Stream Computing Recommendation System Based on Spark Platform [D]* (Jiangsu University)
[6] Li J, Feng and P, Lv J 2013 *An improved slope one algorithm for collaborative filtering [C]* (Ninth International Conference on Natural Computation IEEE) pp 1118-1123.
[7] Panigrahi S, Lenka R K and Stitipragyan A 2016 *A Hybrid Distributed Collaborative Filtering Recom-mender Engine Using Apache Spark [J]* (Procedia Computer Science) 83:1000-1006.