A retweet prediction method of micro-blog big data users based on Map/Reduce

Yuelong Zhao[a] and Meng Fang

School of Computer Science and Engineering, South China university of technology, GuangZhou, China

Abstract. In all kinds of social network software, retweet is a common behavior and a important mechanism for information dissemination. Especially retweet prediction of micro-blog users is very important to deep research. However traditional method can't be effectively applied to big data. To solve this problem, in this paper, first study the most relevant features of retweet, such as proximity social network, retweet activity, etc. Second, use Map/Reduce programming framework to achieve the extraction of feature set and an improved random forests algorithm. Third, gives a distributed approach based on Hadoop platform and use this algorithm for parallel retweet prediction of user's concern edge. Finally, do some experiment by using real data sets of Sina's Micro-blog. Experiments show that this distributed approach based on hadoop platform is better than traditional design, it can effectively predict retweet of Micro-blog user in less time.

Keywords: micro-blog; big data; hadoop; map/reduce; retweet prediction.

1 Introduction

With the rise of social networks, social software has become an indispensable part of people's lives. With incomplete statistics, as of December 2014, the number of active users in Sina micro-blog has reached 176 million, while the average number of online users in QQ reached more than nine thousand. In addition, the number of users, such as Wechat, Douban and other emerging social software is also soared.

Among the many social software, especially micro-blog, retweet is a very important mechanism for information dissemination. The so-called retweet, that is, users can put the thing (comments, music, graph, etc) published by other users he/she concern into their own platform, shared with their fans. This paper specifically for micro-blog retweet behavior, predict whether micro-blog users retweet.

By predict retweet among big data micro-blog users, we will be able to understand the depth of information in the network transfer process. It also has great significance in understanding users' personal interests, network public opinion monitoring, user recommendations, etc.

Now the mainstream micro-blog retweet prediction method in general can be divided into two steps: first, related features extraction. Then use classification algorithm for retweet prediction on all concern edges of all users based on these features.

However, these studies generally doesn't take into account the existing study object is a large amount of data. In view of this, we use Map/Reduce parallel programming model for user feature

[a] Corresponding author : zhaolab@126.com

extraction and implement a new algorithm of random forests, then apply it to the actual micro-blog retweet prediction.

2 Related work

In recent years, with the development of various types of social media platforms, research about dissemination of information in micro-blog has got attention of scholar.

H.Kwak[1] analyzed the Twitter network, but the study results show that compared to the social network - micro-blog, Twitter is a mix of social media and social network. But it tends to social media, rather than a social network. Z.Yang[2] et al proposed a factor graph prediction model, based on the characteristics of retweet in Twitter. Romero [3] et al find that different themes of micro-blog will be different retweet times - the more impressions micro-blog contain political content, the more easily spread. Z.Luo[4] et al studied the behavior of Sina micro-blog retweet, their research has shown that retweet are more easily come in strong connection such as friends, relatives, and pundits connection relations. These studies using support vector machine (SVM) or adaboost, which work well when using small amount of user data. However, due to these algorithms can not be applied very well in Map/Reduce distributed programming framework, it can't work well in case of massive data.

For this reason, this paper will put forward a new random forest algorithm based on Map/Reduce[5] distributed programming framework, and then use the algorithm to predict retweet of all users in Sina micro-blog. Experiments show that the new algorithm greatly improves the computing speed, while prediction hit rate was little changed.

3 Problem description

For social network G = (V, E), where V denotes the set of all users in the network and E denotes the set of concern edges. If user $a \in V$, $b \in V$, and a concern b, the concern edge can be defined as:

$$a \rightarrow b \tag{1}$$

The main issue of this paper is to study: for each concern edge, whether a will retweet message published by b or not.

Fig.1 shows that a user named Jack in the social network diagram:

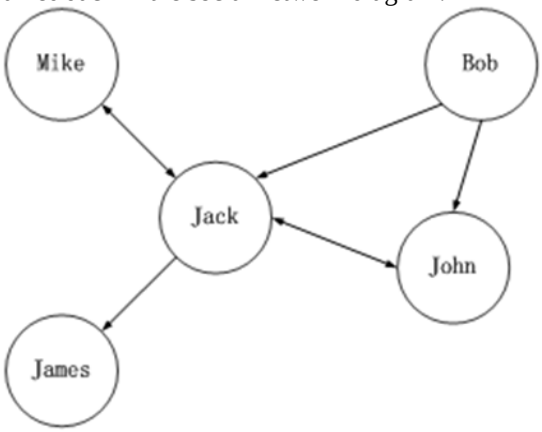

Figure 1. Social network diagram of user jack

As can be seen, the user jack concern Mike, Bob, John and James. This paper will study whether Jack will retweet message published by Mike, John, Bob and James or not.

4 Micro-blog retweet prediction on hadoop

4.1 The overall framework

4.1.1 Analysis features relevant to retweet

Among the many features of the user's information, not all of these is relevant to retweet behavior. Weed out those irrelevant features can not only enhance the computing speed and prediction performance, but also enhance prediction accuracy. Therefore, before retweet to predict, we must analyze a set of features that are most relevant to the user's retweet.

4.1.2 Distributed retweet prediction framework

Among the many features of the user's information, not all of these is relevant to retweet behavior. Weed out those irrelevant features can not only enhance the computing speed and prediction performance, but also enhance prediction accuracy. Therefore, before retweet to predict, we must analyze a set of features that are most relevant to the user's retweet.

Hadoop[6] is a distributed computing platform for big data, which is very popular in recent years. It uses mapreduce as its distributed programming model, which allows user don't the low-level details of distributed system develop a distributed application. This model can assign tasks to each node of a large-scale computer clusters. Firstly, it cut data and task into different parts and then send it to different node in the hadoop platform. Secondly, each node perform tasks in parallel. After the intermediate results are collected, then redistributed to different nodes to calculated again. Finally, summarize the final results.

Specifically, Map/Reduce is a parallel programming model and the process can be divided into two stages: Map and Reduce. In map stage, the object to be processed will be convert to Key-value pairs format. Subsequently, Hadoop will send these Key-value pairs to different reduce node depending on key. Finally the reduce stage will make appropriate treatment for different keys.

This paper has designed a framework of micro-blog retweet prediction based on Hadoop framework(Shown in Fig.2):

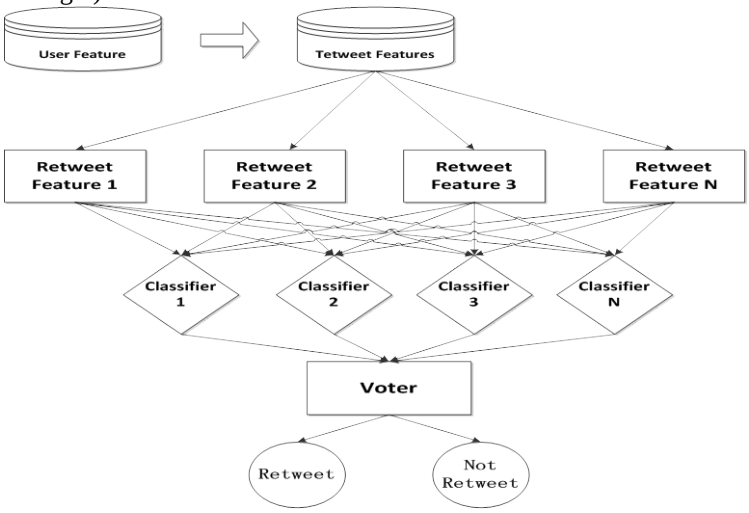

Figure 2. The overall framework

In Fig.2, the first stage is to extract features relevant to retweet from user feature matrix and then convert it into concern edge matrix. It use the Map/Reduce framework for feature extraction and save

it as a feature matrix of concern edge. Compared to the traditional feature extraction method, this distributed design greatly improves the efficiency.

The second stage is to use Map/Reduce framework to achieve a random forest algorithm and then use this algorithm to predict.

Obviously, This program make full use of Map/Reduce framework and give full play to the advantages of distributed.

4.2 Retweet features

In order to improve the accuracy and performance in retweet prediction, we should extract features that most relevant to retweet. Through a rigorous analysis of screening, we picked out 28 relevant features, which contains neighboring social network structure, average number of retweet, social authority attribute ratio and so on.

4.2.1 Neighboring social network structure

One important reason that undoubtedly affect their retweet is neighboring social network structure.

According to the user neighboring social network structure, concern edges can be divided into the following categories:

a. user a and user b concern each other(Shown in Fig.3):

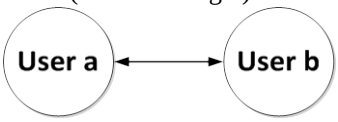

Figure 3. Type of concern edge 1

b. user a concern user b but user b not concern user a(Shown in Fig. 4) :

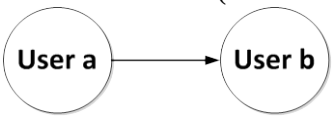

Figure 4. Type of concern edge 2

c. user a and user b concern each other and someone has concern edge with user a and user b(Shown in Fig.5) :

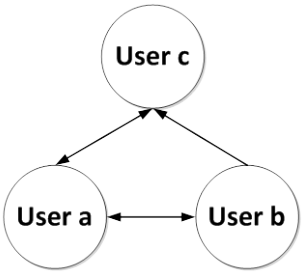

Figure 5. Type of concern edge 3

d. user a concern user b but user b not concern user a and someone has concern edge with user a and user b(Shown in Fig. 6):

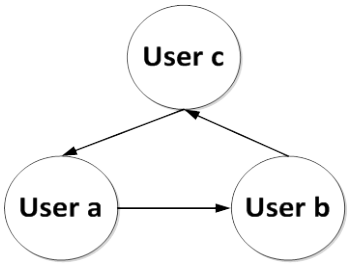

Figure 6. Type of concern edge 4

4.2.2 Retweet activity

Retweet activity is an important factor that affect retweet. This feature can be represent by discrete variable as follows(Shown in Table1):

Table 1. Level of retweet activity

retweet activity	Description
active	retweet more than five times a day on average
general	retweet one to five times a day on average
inactive	retweetless than one times a day on average

4.2.3 Social authority attribute ratio

Research has shown that users tend to retweet news published by authorities. User's Authoritative value proportional to his/her fans number. Social authority attribute ratio is the ratio of one's authoritative value and his/her concerns. It can be calculated by the following (shown in Eq.(2)):

$$p_{a \to b} = \frac{\text{funsNum}(b)}{\text{funsNum}(a)} \tag{2}$$

funsNum is the number of fans, is social authority attribute ratio between a and b.

4.2.4 Tweets heat and Current time

The more times tweet had been retweet, the more likely it is to be retweet. So the retweet times of each tweet need to be recorded, which will be used to improve retweet predicition accuracy.

At different time periods, users' retweet behaviour is different. People tend to retweet in some time periods more than others. So we divide time into 4 time periods:0:00 – 06:00, 06:00 – 12:00, 12:00 – 18:00, 18:00 – 24:00.

4.2.5 Other features

In addition, user location, user age, frequency of recently logins and so on are relevant to retweet behavior. In this paper, a total of 28 features selected.

4.3 Feature extraction based on the Map/Reduce framework

Because the extraction of features of each concern edge is independent, the extraction of features matrix of concern edges can be performed in parallel in the MapReduce framework. Overall, the extraction of concern edges feature matrix can be divided into the following two steps:

Step 1. Input the user feature matrix as MapReduce jobs and split the matrix by user ID.

Step 2. Map function extract features in parallel.

In this paper, each user's features are in different files and each filename is user ID. Therefore, the input keys in map function are filenames, which also means user ID and the input values are features of user. Map function will convert user features matrix to concern edges features matrix. After treatment, the output keys in Map function are user ID and the output values are features of concern edges. After this stage, the user features matrix will be convert into concern edge features matrix.

In feature extraction, only map function is need, so there is no use of the reduce function.

4.4 Random forest algorithm based on Map/Reduce frame

Random forest is a kind of integrated classification algorithm, proposed by Leo Breiman in 2001[7]. The model of this algorithm is composed of some decision trees that has not been pruned [8]. The training set of each decision tree are sample from the original training set. In the process of building a decision tree, each tree node split based on the best split feature, which is according to information gain. Each tree weights equal vote to predict the results of retweet of each concern edge. Due to the nature of this algorithm decide it can be realized in a distributed system, many areas of big data classification/ regression problems using this algorithm to achieve

In this paper, the implementation of random forest based on Map Reduce frame work can be divided into the following steps:

Step 1. A random forest is composed by n cart decision tree classifier. So, extract n subset training set from whole training set by way of sampling with replacement(shown in Eq.(3))[9]:

$$I_t = \{(s_1, c_1), (s_2, c_2), \ldots, (s_p, c_p)\} \tag{3}$$

I is the concern edges matrix, (s,c) is a concern edge, is a subset of I, which is the input split of map function in next phase.

Step 2. map function generate each decision tree classifier in parallel.

The input keys in map function are the serial number of each subset training set, which also means the file name of each subset training set. The input values are the content of each training subset.

For each branch node , which is assigned to a subset training set , it needs to extract m feature from it's training set and split tree node in turn until classification results in subset split set are the same. During the process of splitting, the split components which makes the training set information gain maximum is the best split components. Information gain function is defined as follows(shown in Eq. (4)):

$$G(I, f_i, I_k) = -E(I) + E(I_{rk}, f_i) * p(I_{rk}) + E(I_{lk}, f_i) * p(I_{lk}) \tag{4}$$

Is the eigenvalue of split, E is the entropy, which can be defined as follows(shown in Eq.(5)):

$$E(I) = -\sum_{i=0}^{K} \frac{|I_c|}{|I|} \mathrm{lb} \frac{|I_c|}{|I|} \tag{5}$$

Is the probability of retweet, is the number of samples.

The left and right subtrees recursively split, until it split into a leaf node. When the classification result of the remaining samples are the same, it is converted to a leaf node. When all the samples have been converted to leaf node after training, you get a decision tree DT.

In this step, the output key in map function is the name of random forest and the output value is DT.

Step 3. Reduce function integrate all decision trees to a random forest.

In this stage, the input of reduce function are the same with the output of map function in previous stage. The output key of reduce function is the name of random forest and the output values is DT.

4.5 Retweet prediction based on Map/Reduce frame

After completion of the random forest generation, you can use it to predict retweet[10]. The basic process is divided into two steps: Firstly, input the eigenvectors of concern edge to each decision tree classifier in random forest and record the leaf nodes that final arrived, and then aggregated forecasts of each tree to get the final forecast results. This step is very suitable for the Map/Reduce programming model and the concrete prediction steps as shown below:

Step 1. Input the concern edges to be predicted and the Random forest RF= {DT_1, DT_2,…, DT_n} to Map/Reduce framework.

Step 2. Each map node use decision tree to predict retweet. Its essence is a binary classification problem - retweet or not retweet. The input keys of map function are concern edges and the input values are DT_i. In this step, the output keys of map function are also concern edge, while the output values is tuple <O, T>. T is the label of decision tree, O is the result of retweet prediction of decision tree(the value can be 0 or 1).

Step 3. Reduce function summarizes the results of all the decision trees and make a final determination(prediction) whether each concern edge will retweet.

In this step, the input key-value pairs of reduce function are the same as the output key-value pairs of map function in previous stage.

The results can be obtained by the following formula(shown in Eq.(6)):

$$P(c|s) = \frac{1}{n}\sum_{i=1}^{n} P_i(c|s) \tag{6}$$

Is the probability of retweet on concern edge that predict by decision tree i, is the mean of probability of retweet on concern edges that predicted by all decision trees. If the value is greater than 0.5 determination retweet, otherwise judged not retweet.

The output key of reduce function is concern edge, and the value is the corresponding predicted results.

5 Experiment

In this experiment, first predict retweet in stand-alone mode and then in distributed model. The stand-alone mode experiment based on the design proposed by LUO Zhi-lin[11], and distributed mode based on this paper. Finally, compare results of these two experiments, in order to analysis the performance enhance brought by the framework designed in this paper.

5.1 Experimental data

Experimental data acquired through the API provided by Sina micro-blog. These data generated between January 1, 2015 to March 1, 2015, 181983 user and 738942 concern edges are contained.

5.2 Test Environment

Single experiment based on the R language that widespread use today. R is dedicated to statistical analysis and drawing, and it integrates a lot of statistical algorithms and data mining algorithms,

including data preprocessing, classification, clustering, etc. Hardware configuration is i5-2400 CPU, 4GB of memory. Software configuration is: Windows7 64bit, R language version 3.1.3.

Distributed experiment is based on the design that achieved in this paper. The hardware environment are 23 Lenovo servers, contains a namenode nodes and 22 nodes datanode, The hardware configuration of each node used in distributed experiment are the same with the single machine used in single experiment.

Software configuration includes: Ubuntu 13.04, jdk 1.7.0, Hadoop 1.2.1 appach.

5.3 Parameter settings

The parameters of random forest algorithm, for example, the number of decision tree in random forest - ntree, the number of randomly selected features – m, the number of samples in sub-sample set – k, will largely affect the predicted effect. By reference to the relevant paper , set ntree to 2000, set m to 6, set k to 30% of the total sample.

5.4 Experimental results and analysis

In the first experiment we use 6-fold cross-validation to test the accuracy of the algorithm in this paper.

At first, the total amount of data is randomly divided into six parts, and then five experiments were carried out. In each experiment, one part is selected as the test set, and the other five parts are used as the training sets. Fig 7 shows the accuracy experimental results in stand-alone model and distributed model.

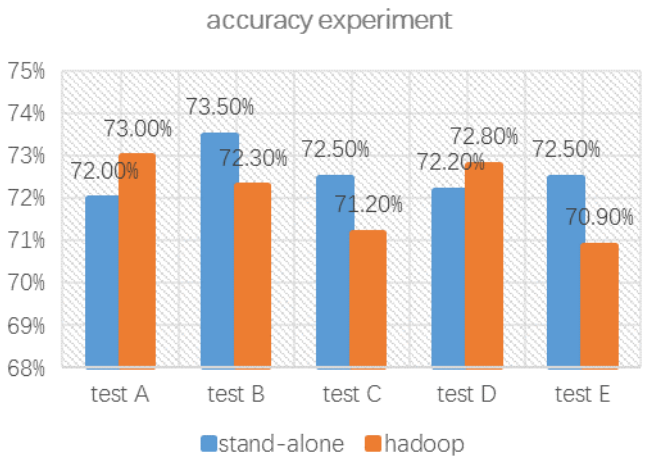

Figure 7. accuracy experiment

As it can be seen, the accuracy of the single experiment and distributed are almost the same.

Next experiment is to test feature extraction time, train time and prediciton time of the algorithm in this paper in stand-alone model and distributed model. This experiment randomly selected 70% of the concern edges for the random forest training, and the remaining 30% for testing. Fig 8 shows the experimental results.

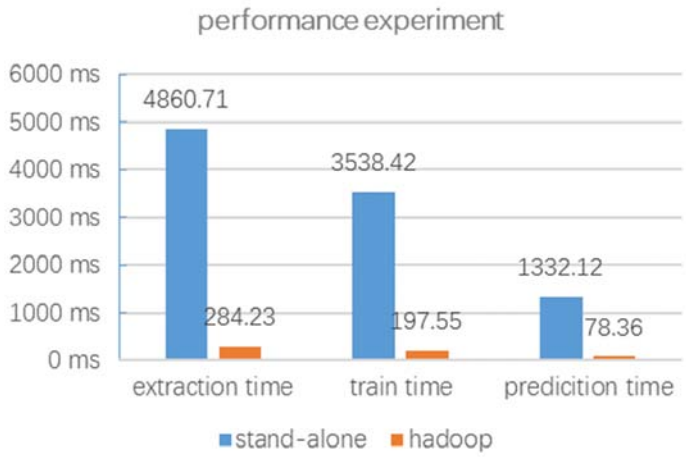

Figure 8. performance experiment

As can be seen, compared to single experiment, feature extraction, random forest training, prediction in distributed experiment has almost speed 17 times. It should be noted that, due to the inherent randomness of random forest, the result will be slightly different, a difference of about 2% to 4%, that can be regarded as within a reasonable range.

6 Summary

This paper implements a micro-blog retweet prediction method based on a Map/Reduce programming model and all stages are achieved parallel processing in this method. Experiments show that the algorithm can significantly improve the processing speed under the conditions of accuracy is not reduced compared to single experiment.

Acknowledgment

This research was supported by the National Natural Science Foundation of China under Grant No. 61572200.

References

1. Kwak H, Lee C, Park H, et al. What is Twitter, a social network or a news media?[C]/ International Conference on World Wide Web. 591-600 (2010)
2. Yang Z, Guo J, Cai K, et al. Understanding retweeting behaviors in social networks[C]/ ACM International Conference on Information and Knowledge Management. ACM, 1633-1636 (2010)
3. Romero D M, Meeder B, Kleinberg J. Differences in the mechanics of information diffusion across topics: idioms, political hashtags, and complex contagion on twitter[C]/ International Conference on World Wide Web, WWW 2011, Hyderabad, India, March 28 - April. 695-704 (2011)
4. Luo Z, Wu X, Cai W, et al. Examining Multi-factor Interactions in Microblogging Based on Log-linear Modeling[C]/Ieee/acm International Conference on Advances in Social Networks Analysis and Mining.189-193 (2012)
5. Dean J, Ghemawat S. MapReduce: Simplified Data Processing on Large Clusters.[J]. Communications of the Acm, **51**(1):107-113 (2008)
6. WHITE T. Hadoop: the definitive guide [M]. 3rd ed. Sebastopol: O'Reilly Media, 2012.
7. Breiman L. Random Forests[J]. Machine Learning, **45**(1):5-32 (2001)

8. Bernard S, Heutte L, Adam S. On the selection of decision trees in random forests [C]/Neural Networks, 2009. IJCNN 2009. International Joint Conference on. IEEE,302-307 (2009)
9. Gatnar E. A diversity measure for tree-based classifier ensembles [M]/Data Analysis and Decision Support. Springer Berlin Heidelberg, **30**-**38** (2005)
10. Huang C, Ding X, Fang C. Head pose estimation based on random forests for multiclass classification[C]/Pattern Recognition (ICPR), 2010 20th International Conference on. IEEE, **934-937**(2010)
11. Luo Z L, Chen T, Cai W D, et al. Microblogging Retweet Prediction Algorithm Based on Random Forest[J]. Computer Science, (2014).