Advances in Intelligent Systems Research, volume 146
2nd International Conference on Artificial Intelligence: Technologies and Applications (ICAITA 2018)

An Improved Naïve Bayes Classifier for Large Scale Text

Huaixin Chen* and Daocai Fu
School of Resources and Environment, University of Electronic Science and Technology of China, P. R. China
*Corresponding author

***Abstract*—Naïve Bayes classifiers is widely used for text classification because of its simplicity and effectiveness. In this paper, an improved Naïve Bayes classifiers was proposed, using multinomial model to modify its rough parameter estimation and parallel competing with MapReduce to categories to text documents. The experimental results show that the proposed method is able to improve the accuracy of Naïve Bayes classifiers dramatically, and has good scalability and extensibility for large-scale text classification.**

Keywords—text classification; Naïve Bayes; words frequency; semantic analysis; parallel computing

I. INTRODUCTION

Since the technology of computer and network appeared, it had been developed very rapidly. Network has becoming one of the most mainly-used information sources. Because most of the information in the network is text data type, automatic text categorization which is the important basic of effective organization and management text data has become an important study field[1]. A variety of machine learning paradigms have been applied to text categorization, including Rocchio[2], Naïve Bayes[3], k-nearest neighbor[4](k-NN), support vector machines[5](SVM) and neural networks (NNet). Naïve Bayes classifier is based on the hypothesis that each attribute is mutual independent, thus it is widely used for its easiness and high efficiency. But because of the text redundant features and rough parameter estimation, the performance of Naïve Bayes in text classification tasks is not good.

Map-Reduce is proposed by Google in 2004. It is a programming tool and an associated implementation for processing and generates large data sets[6]. It simplifies various operations on large data sets. Users specify a map function that processes a key/value pair to generate a set of intermediate key/value pairs, and a reduce function that merges all intermediate values connected with the same intermediate key. Moreover Map-Reduce will provide the fault tolerance, scalability and reliability because its library is designed to help process very large amount of data using hundred and thousands of machine.

The rest of this paper is structured as follows: Section 2 describes what a document is, and how it is represented. Section 3 introduces the Naïve Bayes model and our approach for enhancing Naïve Bayes by using per-document length normalization and word frequency parameter estimation. In Section 4 gives the detail of our methods, illustrating all the steps in MapReduce training and testing tasks. In Section 5 includes experiments and evaluation to show the effectiveness of our proposed approach. Concluding remarks and future work of our work in section 6.

II. TEXT REPRESENTATION

A document is typically stored as a sequence of characters, with characters representing the text of a written natural language expression. Information retrieval has developed a variety of methods for transforming the character string representing a document into a form more amenable to statistical classification. The documents are generally represented using a "bag-of-words" approach[7]. When dealing with a set of documents, document vectors are combined to create a term-document(TD) matrix (1), where x(i,j) denotes the weight for term (ti) in document (di):

$$TD = W = \begin{bmatrix} w_{(1,1)} & w_{(1,2)} & \cdots & w_{(1,n)} \\ w_{(2,1)} & w_{(2,2)} & \cdots & w_{(2,n)} \\ \vdots & \vdots & \ddots & \vdots \\ w_{(m,1)} & w_{(m,2)} & \cdots & w_{(m,n)} \end{bmatrix} \quad (1)$$

A document is represented as a vector. Each dimension corresponds to a separate term. If a term occurs in the document, its value in the vector is non-zero. One of the best known schemes is term frequency - inverse document frequency (TF-IDF) weighting, it is based on an assumption that: the characteristics ability to distinguish the text to appear in the same text higher than in the different text. Based on the above assumptions, there is the classic TF-IDF weighting formula:

$$W(t,d) = \frac{tf(t,d) \times \log_2(N / n_i + a)}{\sqrt{\sum_{t \in d} [tf(t,d) \times \log_2(N / n_i + a)]^2}} \quad (2)$$

W(t,d) is the weight of the feature t in the text d. Where: tf(t,d) is the frequency of feature t in the text d, nt is the amount of text containing t in the text set, N is the total number of text, a is a constant, log2(N/nt+a) inverse text frequency function, that is, the greater nt is, the smaller the value is. In order to reduce the impact of characteristic frequency tf(t,d), you can obtain the following formula:

$$W(t,d)=\frac{(1+\log_2 tf(t,d))\times\log_2(N/n_t+\mathrm{a})}{\sqrt{\sum_{t\in d}[(1+\log_2 tf(t,d))\times\log_2(N/n_t+\mathrm{a})]^2}} \quad (3)$$

Thus, a high weight in tf-idf is reached by a high term frequency and a low document frequency of the term in the whole collection of documents; the weights hence tend to filter out common terms.

III. AVERAGE MULTI-NAÏVE BAYES TEXT CLASSIFICATION

Naïve Bayes is based on Bayes theorem and an attribute independence assumption. Its competitive performance in classification is surprising, because the conditional independence assumption on which it is based, is rarely true in real world applications. Naïve Bayes have been studied extensively by some researchers in text classification task. McCallum compared the different classification results of Naïve Bayes two models: the Bernoulli model and the multinomial model[8]. Jason studied the distribution of data sets, weights error and other system-level error in multinomial distribution, and proposed the corresponding improved method[9], Ashraf think feature weight and text length have a major impact on the text classification results in the multinomial model, and compared the different data sets classification results using support vector machine classifier[10].

Denote a vector of variables d=<di>,i=1,2,…n, represent document, where di is corresponding to a letter, a word, or other attributes about some text in reality, and a set of C={c1,c2,…,ck} is predefined classes. Text classification is to assign a class label cj, j=1,2,…,k from C to a document.

Bayes classifier is a hybrid parameter probability model in essence:

$$p(c_j|d)=\frac{p(d|c_j)\cdot p(c_j)}{p(d)} \quad (4)$$

where P(cj) is prior information of the appearing probability of class cj, p(d) is the information from observations, which is the knowledge from the text itself to be classified, and p(d|cj) is the distribution probability of document d in classes space. Bayes classifier is to integrate these information and compute separately the posteriori of document d falling into each class cj, and assign the document to the class with the highest probability, that is

$$c^*(\mathrm{d})=\arg\max_j p(\mathrm{c}_j|d) \quad (5)$$

Assume the components di are independent with each other since conditional probability p(d|cj) cannot be computed directly in practice. Thus

$$p(d|c_j)=\prod_i p(\mathrm{d}_i|c_j) \quad (6)$$

In the multinomial model, a document is an ordered sequence of word events, drawn from the same vocabulary V. We assume that the lengths of documents are independent of class. We make as assumption: that the probability of each word event in a document is independent of the word's context and position in the document. Thus, each document di is drawn from a multinomial distribution of words with as many independent trials as the length of di. Define fit to be the count of the number of times word wt occurs in document di. Then, the probability of a document given its class is the multinomial distribution:

$$p(d_i|c_j)=p(|d_i|)|d_i|!\prod_{k=1}^{|v|}\frac{p(\mathrm{w}_t|c_j)^{f_{it}}}{f_{it}!} \quad (7)$$

The multinomial distribution after Laplace smoothing manipulation:

$$p(\mathrm{w}_t|c_j)=\frac{1+\sum_{i=1}^{N} f_{it}\,p(c_j|d_i)}{|v|+\sum_{k=1}^{|v|}\sum_{i=1}^{N} f_{it}\,p(c_j|d_i)} \quad (8)$$

To deal with large-scale text data, multiple occurrences occupy the percentages, which bring much redundancy data. This results in a large underestimation of the probability of documents with multiple occurrences of the same word. Therefore, we bring forward an improved algorithm for the words counts as: fit=min{1+log2fit,fit} This does not eliminate word frequencies but pushes down the effect of larger counts. The traditional multinomial model considers each term occurrence as an equally important event, which results in giving different importance to each training document for the learning classifier according to the length of each document. We normalize the text length according to the average method. Then we can change the classification rule as follow:

$$p(d_i|c_j)=\frac{1}{|d|}p(|d_i|)|d_i|!\prod_{k=1}^{v}\frac{p(\mathrm{w}_t|c_j)^{\min\{1+\log_2 f_{it},f_{it}\}}}{f_{it}!} \quad (9)$$

IV. MAPREDUCE FOR TEXT CLASSIFICATION

MapReduce programming model is used in parallel and distributed processing environment to deal with vast amounts processed of data calculations. This model divides a task into sub-tasks. These sub-tasks are dispatched and among the idle processing nodes, it generates the final results through key-value rules to merge. Then we introduced the main job for training and testing processes.

For input text files, it will be input in <file name, file contents> way and process key-value pair. Flowchart of Job1 operation can be illustrated as Figure 1.

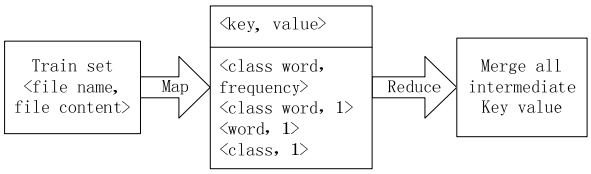

FIGURE I. FLOWCHART OF JOB1 OPERATION

Then we can get the word wt in category cj frequency tf and the text number nt which is containing the feature word t in training set. At the same time calculating the total number of training texts N in the reduce setup, so we can be calculate the weight of each feature word in the next step, The MapReduce job is explained in figure 2.

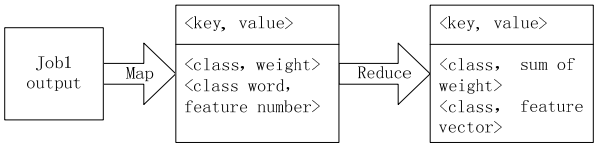

FIGURE II. FLOWCHART OF JOB2 OPERATION

We calculated the feature wt TF-IDF weight in category cj according to the Job2 output, then reduce function is to get this class sum of weight and the total feature number |v|. Then directly construct the vector space model by loading the feature and weight table in HDFS. Finally setting up a MapReduce Job according to equation (8) to obtain the conditional probability p (wt | cj).

In the test process, the processing of each test text is shown in Figure 3.

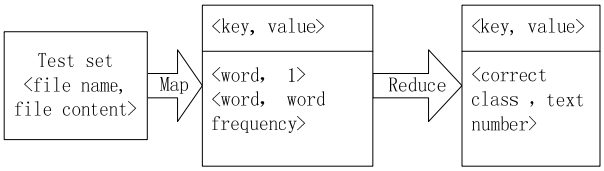

FIGURE III. FLOWCHART OF TESTING OPERATION

According to the length of each test text and word frequency of each word in the text, we can calculate the maximum posterior probability, so the test text category is determined.

V. EXPERIMENT RESULTS

The experiments are done in five nodes Hadoop cluster environment. We repeat the experiment three times and choose the average value as the result. Experimental data using machine learning common standard data sets: 20 newsgroups. This data set includes approximately 20,000 text, distributed in 20 categories.

A. Evaluation Metrics

To evaluate the systems classification accuracy, we used precision, recall and F1 Measure to evaluate the system.

Suppose that the number of the documents which are in cj category in fact and also the classifier judge them to cj category is a; the number of the documents which are not in cj category in fact, however the classifier judge them to cj category is b; the number of the documents which are in cj category in fact, however the classifier do not judge them to cj category is c. So we get contingency table, we can define precision and recall as follows:

Precision, P: Precision is the ratio of the number of documents which judge correctly by classifiers to the number of documents which classifiers judged to this category, so the precision of cj is defined as following:

$$P = \frac{a}{a + b} \tag{10}$$

Recall, R: Recall rate is the ratio of the number of documents which judge correctly by classifiers to the number of documents which are this category in fact, so the recall rate of cj is defined as following:

$$R = \frac{a}{a + c} \tag{11}$$

Then the F1 Measure is defined as:

$$\text{F1 Measure}=2PR/(P+R) \tag{12}$$

B. Experimental Results

Firstly, choosing five categories they are alt.atheism, com.graphics, misc.forsale, rec.autos, sci.med in 20newsgroups did text classification experiments, we were with A, C, M, R, S instead of the corresponding category in Table 1. So we get the classification confusion matrix:

TABLE I. CLASSIFICATION CONFUSION MATRIX

Classification	A	C	M	R	S	Recall/%
A	313	0	15	1	4	93.9
C	13	334	0	23	5	89.0
M	7	6	332	4	14	91.2
R	1	5	13	353	8	92.8
S	0	8	12	4	327	93.1
Precision/%	93.7	94.6	88.9	91.6	91.3	

Then we used all categories, contrast to the Naïve Bayesian classifier, deployed five nodes in different data size, the experimental results is shown in the following figure:

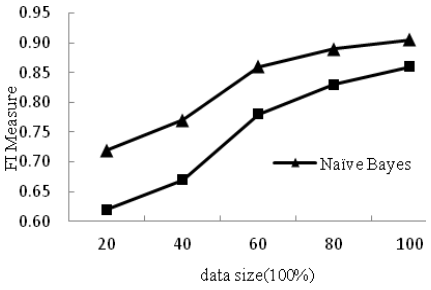

FIGURE IV. COMPARISON EXPERIMENTATION RESULTS IN DIFFERENT DATA SET

Figure 4 shows that the improved Naïve Bayesian classification method was superior to the Naïve Bayesian approach. And with the amount of data increases, the value of F1 is also improved. That confirmed the improved algorithm has good flexibility, it can adapt to greater the amount of data size.

In the efficiency test, we deal the same training mission with different node numbers, and compared between different node numbers of training time and speedup ratio. Test results are shown in figure 5.

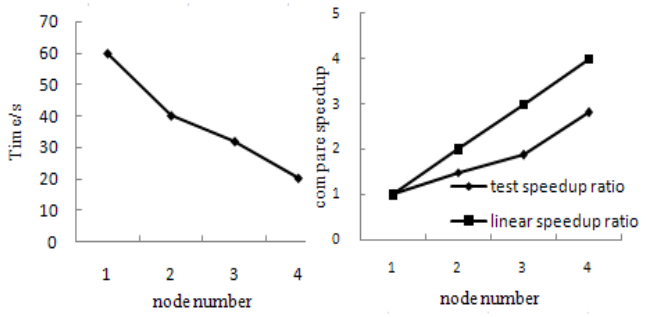

FIGURE V. RUNTIME AND SPEEDUP IN DIFFERENT NODE

VI. CONCLUSION

This paper implements a parallel MapReduce framework to achieve a simple and efficient text classification algorithm - average multinomial Naïve Bayesian classification methods. In this method, due to reduce the impact of a large number of redundant features and text length for text classification result, it is superior to the general Bayesian method on the classification accuracy. Because of using parallel computing, it has a good scalability and large-scale data processing capabilities. As a future work the experiments can be expanded to a cluster with more machines, which increase the computational power and reduce the time cost.

ACKNOWLEDGMENT

This study is supported by New Teacher's Fund of University of Electronic Science and Technology of China(Grant No. ZYGX2017KYQD194).

REFERENCES

[1] HE Qing, LI Ning, LUO Wen-juan, et al. A survey of Machine Learning Algorithms for Big Data. Pattern recognition and Artificial intelligence, 2014, 27(4):328-336.(in Chinese)

[2] Thorsten J. A probabilistic analysis of the Rocchio algorithm with TF-IDF for text categorization, Proc of the 14th International Conference on Machine Learning(ICM L-97). 1997:143-151.

[3] Lo S, Ding L. Probabilistic reasoning on back- ground net: An application to text categorization, Proc of 2012 International Conference on Machine Learning and Cybernetics (ICMLC) .IEEE Press,2012,2: 688-694.

[4] Kuang, F., Xu, W., & Zhang, S. A Novel Hybrid KPCA and SVM with GA Model for Intrusion Detection, Applied Soft Computing, 18: 178-184.

[5] Alpaydın, E. Introduction to Machine Learning, 2nd Edition Cambridge, MA: The MIT Press, 2010.

[6] White T. Hadoop: the definitive guide. O'Reilly Media Inc, 2009.

[7] Makoto Suzuki, Naohide Yamagishi, Takashi Ishidat, et al. ,On a New Model for Automatic Text Categorization Based on Vector Space Model, Systems Man and Cybernetics (SMC), IEEE International Conference, 2010.

[8] F.Thabtah, M. Eljinini, M. Zamzeer, and W. Hadi, Naïve Bayesian based on chi square to categorize Arabic data, proceedings of the 11th International Business Information Management Association Conference (IBIMA) Conference on Innovation and Knowledge Management in Twin Track Economies, 2009, 4–6.

[9] Kim Sang-bum, Han Kyoung-soo, Rim Hae-chang. Some Effective Techniques for Naïve Bayes Text Classification. IEEE Press 2006:1457-1466.

[10] Su Jiang, SHIRAB J S, MATWIN S. Large scale text classification using semi-supervised multinomial Naïve Bayes, Proc of the 28th International Conference on Machine Learning (ICML-11). 2011: 97-104.

[11] Chu C, Kim S, Lin Y A, et al. Map-Reduce for machine learning on multicore , Proceedings of NIPS 19,2007

[12] Goncalves C, Assuncao L, Cunha J C. Data analytics in the cloud with flexible MapReduce workflows, Proc of 4th International Conference on Cloud Computing Technology and Science (CloudCom). IEEE Press, 2012:427-434.

[13] S.Sathya, Prof. M.Victor Jose, Application of Hadoop MapReduce Technique to Virtual Database System Design, Noorul Islam Centre for Higher Education, ICETECT 2011.

[14] Rama Satish K V and N P Kavya , An approach to optimize QOS Scheduling of Map-Reduce in Big Data, International Journal of Engineering Research and Technology, Volume 2, Issue 11, May 2014.

[15] BenediktElser, Alberto Montresor, An Evaluation Study of Big Data Frameworks for Graph Processing, IEEE International Conference on Big Data, 2013, 60-67