Research on improved K - nearest neighbor algorithm based on spark platform

Yushui Geng, Xianzhao Yan

Qilu University of Technology, Jinan, China, gys@spu.edu.cn
Qilu University of Technology, Jinan, China, yxzzhao@163.com

Keywords: big data; hadoop; spark; K - Nearest Neighbor Algorithm; weight.

Abstract

Today, big data technology is growing rapidly. The birth of Hadoop makes people concerned about the study of MapReduce, And Spark through the introduction of RDD data model and memory-based computing model, So that it can be well adapted to the data mining of big data this scene, And superior to Hadoop in iterative computing, Quickly became the majority of enterprises, scholars of the research focus. K nearest neighbor algorithm(KNN is used instead of the following) is a very important classification algorithm. A lot of people are studying it, But there is no mature solution to the algorithm in the spark platform to achieve parallelization. In this paper, The author realizes the parallelization of the improved KNN on the spark platform. We use clustering algorithms, Find the weight of each training sample in the training sample set, The weights of the K samples are used to distinguish the K nearest neighbors from the test sample. It is proved by experiments that the improved KNN has better accuracy.

1 Spark Overview

Spark is the University of California, Berkeley AMP Laboratory Develops a common memory parallel computing framework. Spark with its advanced design concept, quickly become a popular community projects, Spark introduced Spark SQL, Spark Streaming, MLlib and GraphX components, that is, BDAS[1], As shown in Figure 1 below. Spark Core provides a memory computing framework, Spark SQL[2] is used for timely query, Spark Streaming[3] is used for real-time processing applications, MLlib[4] or MLbase is used for machine learning and GraphX[5] for graph processing. these components gradually form a large data processing one-stop solution platform.

From all aspects of Spark hope to replace Hadoop in the status of large data, a large data processing mainstream standards, But Spark has a great way to go from this goal. Spark is implemented using the Scala language, an object-oriented, functional programming language that makes it easy to manipulate distributed data sets as well as manipulating local collection objects. In the Spark official website, it has a fast running, easy to use, versatile and run anywhere.

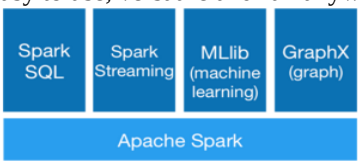

Figure 1. Spark's entire ecosystem a Berkeley data analysis stack

2 Spark Task Execution Process

Spark task execution process is divided into two stages, The first stage generates the RDD dataset, Incremental build DAG diagram (Direct Acyclic Graph, directed acyclic graph).Lineage record transform operator sequence, The second stage Task Scheduler through the Cluster manager such as (Mesos, Yarn), etc. will be sent to the DAG task set to the node in the cluster. Figure 2 shows the running scenario of the Spark program.

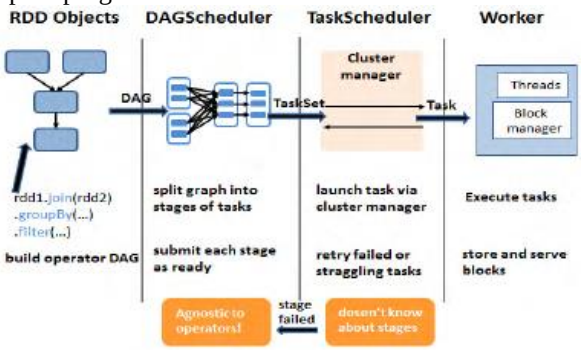

Figure 2. Spark program running process

In general, each Spark application through the driver (driver program) call the user's main function, Perform RDD on the cluster in parallel. RDD can be created from HDFS or generated by an existing RDD calculation, Users can keep RDD in memory for easy reuse, and finally when RDD is lost, it can be automatically restored via Lineage.

3 The Traditional K Near Neighbor Algorithm

Unlike other model-based and rule-based classification algorithms, The KNN is an instance-based supervised machine learning method. The classification of a new sample

is determined by the majority of its K nearest neighbors. The algorithm does not use any empirical pattern to match, So it is intuitive, without a priori statistical knowledge, no teacher learning and other characteristics, thus becoming a non-parametric classification of an important method. KNN is the guiding ideology of " One takes on the attributes of one's associates". By your neighbors to infer your category. For example, for a query point, find out the nearest K training points, And then use the "minority to obey the majority" of the voting method, the query points belong to the category. In fact, the KNN uses the neighbor classification as a predictor of a new query case.

KNN is very simple to achieve. Given a sample set X with m samples, each sample has n attributes:Xi=(xi1,xi2,⋯,xin); i=1,2, ⋯ ,m.Suppose a sample to be classified : Y=(y1,y2,⋯,yn).Then, The procedure for determining the KNN of the Y class is as follows:

[1] Calculate the distance between the sample to be sorted and the European distance of all training samples: Dist(Y,Xi)= $\sqrt{\sum_{j=1}^{n}(y_j - x_{ij})^2}$.

[2] Find neighbors: delineation of the nearest k training objects, as the test object of the neighbors.

[3] Classification: According to the main categories of k nearest neighbors, to classify the test object.

4 Improved K Near Neighbor Algorithm Based On Spark Platform

Although the KNN is robust and easy to use. But it still has many shortcomings:

[1] When calculating the distance from the test sample to the training sample, all attributes have the same status. However, in practical applications, the role of each attribute in the classification is not the same. Many researchers have improved their typical work, including Tian Xuan, Sang Ying bin proposed weighted Euclidean distance calculation method[6,7],Li Wei et al. Proposed a weighted cosine similarity calculation method[8].But none of these studies have parallelized the algorithm on the spark platform.

[2] The amount of calculation is large. Because for each of the data to be classified to calculate it to the distance of all known samples in order to obtain its K nearest neighbors. We can delete some of the samples that do not make much contribution to the classification, which eliminates some unnecessary computational overhead. The main method is the condensed algorithm[9],Edited algorithm[10].

[3] Improvement of K value selection method. KNN, the choice of K value is a focus of attention, Because we can not know in advance what the value of K is more appropriate, Previously, the K value was adjusted for the given experiment by constant experimentation. Ghosh et al. On how to determine the K value has been improved, using a dynamic method to determine the K value[11].

At present, there are few related papers on the Spark platform, and most of them are on the Hadoop platform[12-14].

In this paper, we focus on the first point of the KNN, combine the KNN and the Kmeans algorithm, Calculate the weight of each attribute in the test sample, Finally, we use weighted voting strategy to classify. And the improved KNN is scaled with the scala language on the spark platform. Specific steps are as follows:

[1] Read the data from the native data space into the Spark Rdd space and encapsulate each data into the RDD [LabeledPoint] data type.

[2] Normalize the data. In this paper, the normalization of the sample data using the minimum - the largest standardized, The method is as follows: For each attribute, find the minimum and maximum values for the attribute corresponding to all the samples in the dataset. Thus, the definition is as follows: $\frac{x_i - \min(x_i)}{\max(x_i) - \min(x_i)}$. Where max(xi)and min(xi) are the maximum and minimum values of the i-th attribute of all the samples in the data set, respectively. After normalization, the values of each attribute of the n data samples are mapped to the [0,1] interval.

[3] Use Kmeans algorithm to calculate the cluster center of each category.(The Kmeans algorithm has been parallelized in mllib and can be called directly through the KMeans.train).

[4] Calculate the weight of each data. weight=1/(dist*dist),dist is the distance of each data to each corresponding cluster center. When the dist is far from the center of the category, the lower the degree of determination of the category, The corresponding number of weights is also lower; On the contrary, if dist is closer to the center of the category, the greater the number of weights.

[5] Classify by weight.

Here are the types of data used in spark:

RDD, full name is Resilient Distributed Datasets, is a fault-tolerant, parallel data structure, Allows users to explicitly store data to disk and memory, and can control the data partition. At the same time, RDD also provides a rich set of operations to operate the data. In these operations, such as map, flatMap, filter and other conversion operations, a good fit Scala collection operation.

RDD as a data structure, in essence, is a read-only partition record set. An RDD can contain multiple partitions, each of which is a dataset fragment. RDD can be interdependent. If each partition of RDD can only be used by a partition of a Child RDD, it is called narrow dependency; if multiple Child RDD partitions can be dependent, it is called wide dependency. Different operations may produce different dependencies depending on their characteristics.

LabeledPoint simply, can be understood as a vector corresponding to a special value, the value of the specific content can be specified by the user, such as you developed an algorithm A, the algorithm for each vector will be processed after a special mark value P, you can use p as a vector tag.

5 Experiment And Analysis

In this paper, the experiment in a real environment, the specific version of the experiment is as follows: Spark Version: spark-1.6.1, Scala Version: scala-2.10.6, Hadoop Version: hadoop-2.7.1, JDK Version: jdk1.8.0_66.

In this paper, we build a Spark cluster composed of three nodes, Specific steps to build Spark cluster here is no longer described in detail, We built the Spark cluster on top of Hadoop HDFS, Which HDFS is responsible for storage, Spark replaced MapReduce calculation framework, responsible for the calculation of data in the experiment.

In the master and slave nodes, we will see the following process, as shown in Figure 3.

```
[yxz@spark01 sbin]$ jps
3284 NameNode
3576 SecondaryNameNode
3774 Master
3839 Jps
```

Figure 3. Master and slave node process

At the same time, Figure 4 shows the parallel KNN implementation process through the Spark cluster UI to monitor the operation of the cluster.

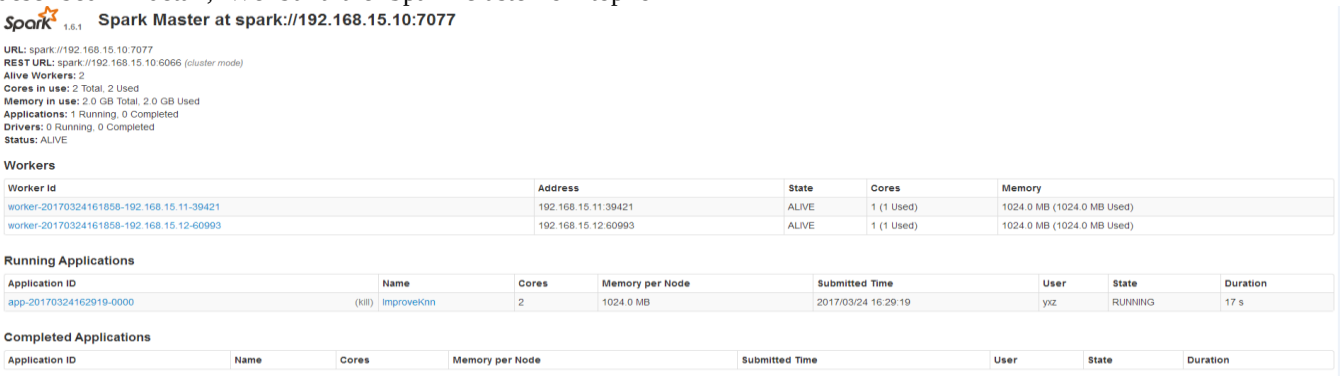

Figure 4. The Spark cluster UI monitor the operation of the cluster

The job in spark is shown in the figure 5.

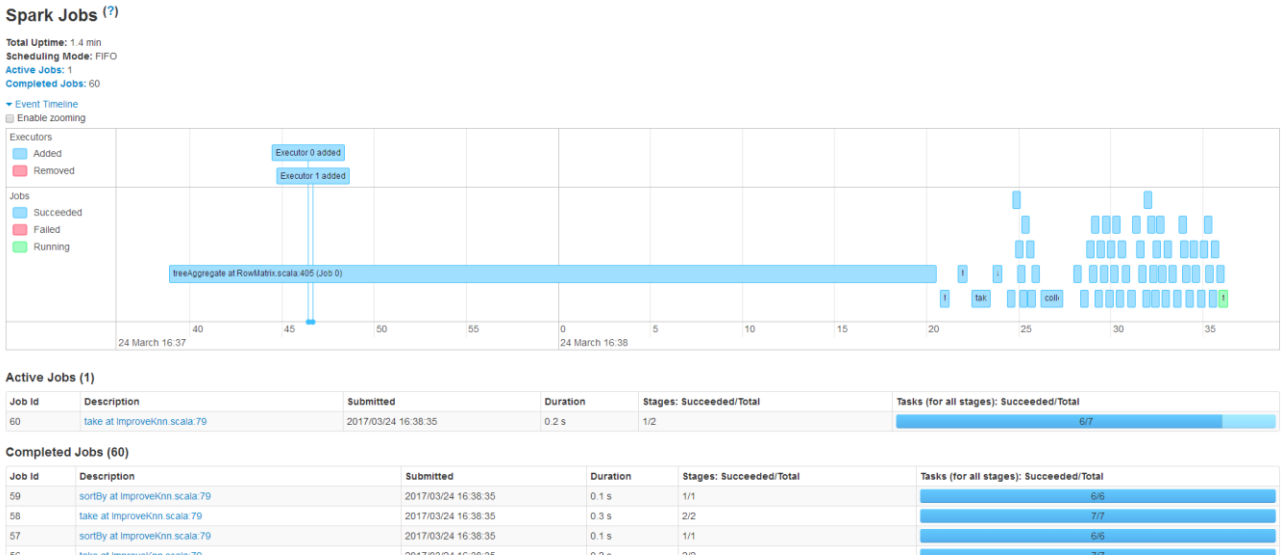

Figure 5. Spark jobs

The experimental dataset of this paper uses the datingTestSet ,transfusion and banknote authentication datasets in the UCI machine learning data set. The cross validation will be used to verify the accuracy of the algorithm. We use a portion of the data training model to evaluate the performance of the model. We used the data set 60% of the data training model, with 40% of the data to test. Here will be improved before the algorithm in the spark platform classification and algorithm to improve the classification of the results are compared as shown Figure 6:

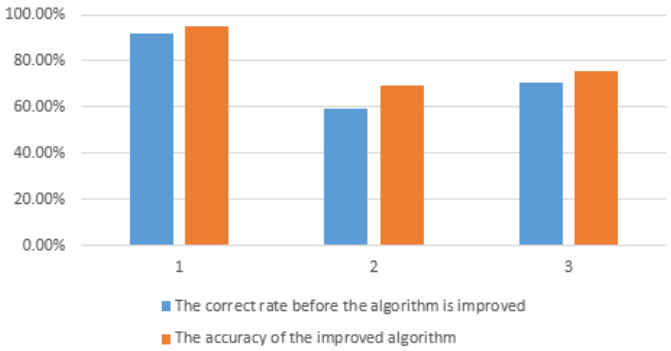

Figure 6. Comparison of accuracy before and after algorithm improvement

Experiments show that the improved KNN has better accuracy.

6 Conclusion

Nowadays, many people pay attention to the big data processing, A variety of distributed systems, cloud computing platform after another, While the system's function and performance are also uneven, Spark large data processing platform to absorb the essence of the mainstream processing system Hadoop, to its dregs, With hundreds of times faster than Hadoop's performance in the rapid development, replacing Hadoop will be just around the corner. The value of big data is mining, data mining has become a big data processing system is an essential part, Therefore, the author of this paper in-depth study of Spark's system architecture, programming model and its data mining module, Found that the machine learning algorithm library mllib algorithm library there are many algorithms did not achieve, Which KNN is one of them, This algorithm is simple and effective, easy to implement, in the field of artificial intelligence such as expert systems, data mining, pattern recognition and other aspects of widely used. Therefore, this paper improves the KNN and implements the parallelization on the Spark platform.

Of course, although this article has made great efforts and a lot of research to do KNN improvement, In the Spark parallelization and application data set, made some small results, but there are still some shortcomings. First, in the Spark parallelization, the KNN is fine-grained task parallel support is insufficient, the performance is poor; Moreover, since the RDD does not support the updating of variables in the Scala native space in operations such as maps, Some key operations are not parallelized, And RDD can not produce new RDD, can not do the task parallel and parallel data at the same time. Second, combining the kmeans algorithm with the KNN improves the accuracy of the algorithm, but the time of the computational task increases, the next step must also take into account the efficiency of the algorithm.

Spark platform is still in continuous research and development, is not very mature, it is necessary to continue to study the platform in the data mining parallel content. Spark platform itself are some shortcomings, also need to improve the place, Such as RDD can not be nested, that is, within the RDD can no longer trigger action, Spark also has much room for improvement.

Acknowledgements

This work was supported in part by Shandong Provincial Natural Science Foundation, China (No.ZR2014FQ021), Key Research and Development Plan Project of Shandong Province (No.2015GGX106003), and Shandong Province Higher Educational Science and Technology Program (No.J15LN03).

References

[1] Franklin M. The berkeley data analytics stack: Present and future[C]//Big Data, 2013 IEEE International Conference on. IEEE, 2013: 2-3.

[2] Armbrust M, Xin R S, Lian C, et al. Spark sql: Relational data processing in spark[C]//Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data. ACM, 2015: 1383-1394.

[3] Zaharia M, Das T, Li H, et al. Discretized Streams: An Efficient and Fault-Tolerant Model for Stream Processing on Large Clusters[J]. HotCloud, 2012, 12: 10-10.

[4] Meng X, Bradley J, Yavuz B, et al. Mllib: Machine learning in apache spark[J]. Journal of Machine Learning Research, 2016, 17(34): 1-7.

[5] Gonzalez J E, Xin R S, Dave A, et al. GraphX: Graph Processing in a Distributed Dataflow Framework[C]//OSDI. 2014, 14: 599-613.

[6] TIAN Xuan, LIU Xi-yu, MENG Qiang.Study on Document Clustering Based on BP Neural Network [J] .Computer Science, 2002, 29 (8): 93-95.

[7] Sang Yingbin. K classification algorithm based on K neighborhood [D]. Chongqing University, 2009.

[8] Journal of Jilin University, Information Science Edition, 2010, 28 (1): 68-76. [J]. Journal of Jilin University, 2010, 28 (1): 68-76. [J]. Journal of Jilin University, 2010, 28 (1): 68-76.

[9] Angiulli F. Fast condensed nearest neighbor rule[C]//Proceedings of the 22nd international conference on Machine learning. ACM, 2005: 25-32.

[10] Wilson D L. Asymptotic properties of nearest neighbor rules using edited data[J]. IEEE Transactions on Systems, Man, and Cybernetics, 1972, 2(3): 408-421.

[11] Ghosh A K, Chaudhuri P, Murthy C A. On visualization and aggregation of nearest neighbor classifiers[J]. IEEE transactions on pattern analysis and machine intelligence, 2005, 27(10): 1592-1602.

[12] Yokoyama T, Ishikawa Y, Suzuki Y. Processing all k-nearest neighbor queries in hadoop[C]//International Conference on Web-Age Information Management. Springer Berlin Heidelberg, 2012: 346-351.

[13] He Q, Zhuang F, Li J, et al. Parallel implementation of classification algorithms based on

MapReduce[C]//International Conference on Rough Sets and Knowledge Technology. Springer Berlin Heidelberg, 2010: 655-662.

[14] Ghoting A, Kambadur P, Pednault E, et al. NIMBLE: a toolkit for the implementation of parallel data mining and machine learning algorithms on mapreduce[C]//Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2011: 334-342.