Volume 6, Issue 2, February 2016 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Implementation of Optimised K-Means Clustering on Hadoop Platform

E. Ezhilan, A. Vignesh Kumar, G. Arvind Kumar, A. Afzal Khan
Computer Science and Engineering, Bannari Amman Institute of Technology,
Sathyamangalam, Tamilnadu, India

Abstract: ***Cluster is defined as a collection of data members having similar characteristics. The process by which a relation is established or an information is derived from raw data by performing some operations on the data set like clustering is known as data mining. K-means clustering algorithm is one of the most widely used clustering algorithms and it is being applied in many fields of science and technology. A major problem of the k-means clustering algorithm is that the results in different types of clusters depends on the initial centroid which is choose at random.In this paper, we propose an optimized K-means clustering algorithm in which the initial centers are based on data dimensional density. This method improves the stability of the K-means clustering by avoiding the deficiency of the random initial centers.Thispaperdiscusses the implementation of the K-Means Clustering Algorithm over a distributed environment using Apache Hadoop. The key to the implementation of the K-Means Algorithm is the design of the Mapper and Reducer codes which has been discussed in the later part of the paper.***

Keywords: K-Means Clustering, MapReduce, Hadoop, Initial Center, Density.

I. INTRODUCTION

In recent years, with the development of inflation technology, the inflation network generated a large amount of data all the time. Faced with these large data, if put to good use, we can extract many important information. Clustering is one of the most widely used method for exploratory data analysis. K-means clustering as one of the traditional and widely used clustering algorithms has demonstrated good results. However, with the explosive growth of data, K-means clustering running on separate servers cannot meet the demand.

Data mining can be defined as subject that discovers data relations by applying principles of AI, database systems, statistics and likewise. In addition to just analysis this facilitates visualization, data management aspects, data modeling, complexity considerations. Distributed Computing is a method aimed at solving computational problems mainly by sharing the computation over a network of interconnected devices. Each individual system connected on the network is called a node and the collection of many nodes that form a network is called a cluster.

Apache Hadoop is one such open source framework that supports distributed computing. It came into extant from Google's MapReduce and Google File Systems projects. It is a platform that can be used for high data applications which are processed in a distributed environment. It follows a Map and Reduce programming model where the splitting of data is the elementary step and this fragmented data is fed into the distributed network for the purpose of processing. The processed data is then integrated as a whole.

To solve the problem, this paper proposes a K-means clustering with optimized initial centers based on data dimensional density implemented on the Hadoop platform. This proposed concept avoids the affects of the random initial centers and improves the stability of the K-means clustering.

II. RELATED WORK

A. Introduction of K-means clustering

K-Means Clustering is a method used to classify semi structured or unstructured data sets. This is one of the most commonly and effective methods to classify data because of its simplicity and ability to handle voluminous data sets. It accepts the number of clusters and the initial set of centroids as parameters. The distance of each item in the data set is calculated with each of the centroids of the respective cluster. The item is then assigned to the cluster with which the distance of the item is the least. The centroid of the cluster to which the item was assigned is recalculated. One of the most important and commonly used methods for grouping the items of a data set using K-Means Clustering is calculating the distance of the point from the chosen mean. This distance is usually the Euclidean Distance though there are other such distance calculating techniques in existence. This is the most common metric for comparison of points. Suppose there the two points are defined as P = (x1(P), x2(P), x3(P) …..) and Q = (x1(Q), x2(Q), x3(Q) …..). The distance is calculated by the formula given by

$$d(P,Q) = \sqrt{((x1(P)-x1(Q))^2 + (x2(P)-x2(Q))^2 + ...)}$$
$$= \sqrt{\left(\sum_{j=1}^{p} (xj(P)-xj(Q))^2\right)} \quad \text{eq. (1)}$$

The next important parameter is the cluster centroid. The point whose coordinates corresponds to the mean of the coordinates of all the points in the cluster. The data set may or better said will have certain items that may not be related to any cluster and hence cannot be classified under them, such points are referred to as outliers and more often than not correspond to the extremes of the data set depending on whether their values or extremely high or low. The main objective of the algorithm is to obtain a minimal squared difference between the centroid of the cluster and the item in the dataset.

$$|x_i^{(j)} - c_j|^2 \quad \text{eq. (2)}$$

Where xi is the value of the item and cj is the value of the centroid of the cluster.

The Algorithm is discussed below:

- The required number of cluster must be chosen. We will refer to the number of clusters to be 'K'.
- The next step is to choose distant and distinct centroids for each of the chosen set of K clusters.
- The third step is to consider each element of the given set and compare its distance to all the centroids of the K clusters. Based on the calculated distance the element is added to the cluster whose centroid is nearest to the element.
- The cluster centroids are recalculated after each assignment or a set of assignments.
- This is an iterative method and continuously updated.

B. Introduction of MapReduce programming model

*M*apReduce is a programming paradigm used for computation of large datasets. A standard MapReduce process computes terabytes or even petabytes of data on interconnected systems forming a cluster of nodes. MapReduce implementation splits the huge data into chunks that are independently fed to the nodes so the number and size of each chunk of data is dependent on the number of nodes connected to the network. The programmer designs a Map function that uses a (key,value) pair for computation. The Map function results in the creation of another set of data in form of (key,value) pair which is known as the intermediate data set. The programmer also designs a Reduce function that combines value elements of the (key,value) paired intermediate data set having the same intermediate key.

Map and Reduce steps are separate and distinct and complete freedom is given to the programmer to design them. Each of the Map and Reduce steps are performed in parallel on pairs of (key,value) data members. Thereby the program is segmented into two distinct and well defined stages namely Map and Reduce. The Map stage involves execution of a function on a given data set in the form of (key,value) and generates the intermediate data set. The generated intermediate data set is then organized for the implementation of the Reduce operation. Data transfer takes place between the Map and Reduce functions. The Reduce function compiles all the data sets bearing the particular key and this process is repeated for all the various key values. The final out put produced by the Reduce call is also a dataset of (key,value) pairs. An important thing to note is that the execution of the Reduce function is possible only after the Mapping process is complete. Each MapReduce Framework has a solo Job Tracker and multiple task trackers. Each node connected to the network has

III. ALGORITHM DESIGN AND IMPLEMENTATION

In the previous section, we have introduced the related technology used in our algorithm. In this section, we will begin to introduce our algorithm design and implementation.Before we introduce the procedure of the algorithm, we define some concepts as follows:

A. Basic ideas and concepts

In this paper, we propose a K-means clustering with optimized initial centers based on data dimensional density, Considering that the algorithm has parallel characteristics, we implement it in the MapReduce programming model on Hadoop platform. The basic idea of the algorithm is that we choose the farthest k (where k is the number of clusters, given in advance) points in the high density areas as the initial center. Using this idea the algorithm can not only exclude some isolated points affecting the final clustering results, but also achieve final stable clustering results faster.

1) Average distance R

In our algorithm, we only consider the numeric dataset. Therefore, we treat each tuple as a point and represent it as a vector. Then we calculate the distance between any two points of the dataset. The average distance R means the average distance between any two points of the dataset. (The distance is defined as the euclidean distance)

2) Density index

The density index is a very important parameter in the algorithm. It is defined as the average count of the points in each cluster. As we know, points can not be uniformly assigned in each cluster, so the density index must float down 20%-40% based on the experience. For example, we have a dataset containing 100 points and the given number of clusters is 2. The average count of the points in each cluster is 100/2, and the appropriate number must foat down 20%-40%, so the density index of the dataset is 30-40.

3) High density area

According to our previous definition, the high density area will be defined as follows:

Given a point, we create a ball which uses the point as the center and the average distance R as the radius, calculate the count of points in the ball. If the count of points are greater than the density index, the point is in the high density area, on the contrary, the point is in the low density area.

B. Steps of the algorithm

Based on the basic concepts above, we proposed the K-means algorithm with optimized initial centers, The steps of the algorithm are described as follows:

1) Calculate the distance of each point between all other points in the dataset. The distance is defned as Euclidean distance. The distance computation formula is shown as Eq. 2. In theEq, 2, the $d_{i,j}$ means the distance between the point X_i and the point X_j in the dataset. And we assume that the point in the dataset has d dimensions, so the X_{il}means the value of first dimension of the point

$$d(P,Q) = \sqrt{((x1(P)-x1(Q))^2 + (x2(P)-x2(Q))^2 + \ldots)} = \sqrt{\left(\sum_{j=1}^{p} (xj(P)-xj(Q))^2\right)} \quad \text{eq. (3)}$$

2) Calculate the average distance R of the dataset. The calculation formula is shown as Eg. 3. In the Eg, 3, the denominator means the sum of distance between all points, and the molecular means the total number of the distances between points.

$$R = \frac{\sum_{i<=n, i<j<=n} d_{i,j}}{n*(n-1)/2} \quad \text{eq. (4)}$$

Calculate k-nearest neighborhood distance named distance (p, i) which is defned as the direct connection distance between object p and i,

$$distance = \sqrt{(x^1 - y^1)^2 + (x^2 - y^2)^2 + \cdots (x^n + y^n)^2} \quad \text{, n eq. (5)}$$

Eq. (5) is the dimension of the dataset.

3) Calculate the density index of the dataset. The density of object pwhich reflects the distribution of the data near is defned by the reciprocal of k-nearest neighbor mean. It is described as follows:

$$lrd(p) = \frac{1}{\frac{1}{k}\sum_{i=1}^{k} distance(p,i)}. \quad \text{eq. (6)}$$

Where lrd (p) is the local density of p.

4) Test whether a point is in the high density area, If the point is in the high density area, assigned the point to high density point set. On the contrary, assigned the point to low density point set.

5) Select K(K is the number of the cluster) points from the high density point set which have the farthest distances as initial cluster centers. The specific step is to sort the distances of all points in the high density point set, and select the two points which have the farthest distances assigned to the initial center set. Then we compare the distance of points in the initial center set to the other points, fmd the farthest distance to the points in the initial center set and assign the point which has the farthest distance to the initial center set until the number of points in the initial center is equal to the number of cluster.

In summary, through the steps above, we find the initial center set as fast as possible. Because of the farthest distance between these points, we confirm that these points have thebiggest difference between clusters. The final clustering results will be more stable compared to the initial center using a random method. But adding the optimized algorithm will bring a new problem. If the dataset has more data or higher dimensions, the computational time will have a substantial increase.

C. Algorithm implementation

After the analysis above, we migrate the judging process to a cloud computing platform, such as Hadoop. Because Hadoop has helped us to do a series of configuration work, we only need to prepare the corresponding map function and reduce function.

The map function of the judging process using MapReduce is shown in Figure 1, and this function runs on each mapper in the cluster. This function compares the each distance of the point which we want to judge with average distance R and count the number of which distance is less than average distance R. Then we assign the point to the corresponding set according to the comparison result of the count and the index density.

The reduce function of the judging process using MapReduce is shown in Figure 2, and this function runs on each reducer in the cluster. This function divides all the points into two sets, a set of points which are in the high density area, and the other set of points which are in the low density area. The operation is completed in the framework of MapReduce automatically.

Map Function
Input:<key, value> pairs, key is the serial number of the point that we want to judge, and

value is the union of the distance between this point and the rest points. **Output:**<key, value> pairs, the key of output just have two kinds of values, number 1 if the point is in the high density area, number 2 if not. The value of output is the serial number of the point that we want to judge. **Steps:** 1. Count+O 2. For each distance E input. value 3. If distance < R 4. Count++ 5. End if 6. End for 7. If count >= density index 8. Output. Collect (1, input. key) 9. Else 10. Output. Collect (2, input. key)

Reduce Function
Input:<key, List<value» pairs, key is the same as the key of map fnction output, List<value> is the union of the points in the high density area if key is number 1, or in the low density area if key is number 2. **Output:**<key, List<value» , the same as the input of reduce function **Steps:** l. Output. Collect (input. key, input. value)

For the test of the clustering stability, we used one dataset which have the labels. And make experiments with random initial centers of different randomly seeds 20 times and proposed method. The results of the experiments are shown in Fig. 1.

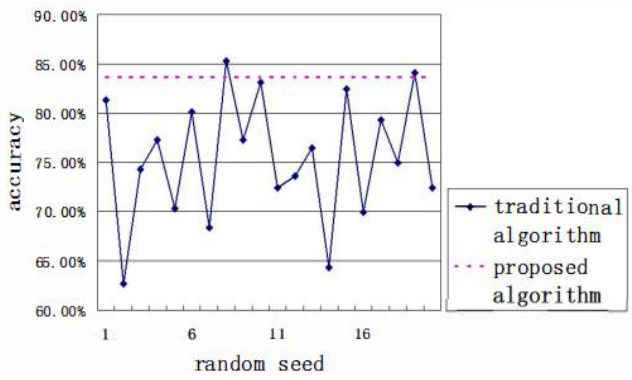

Fig. 1.

Fig.1 shows the comparison of accuracy between traditional algorithm and proposed algorithm. Through the data in the figure can be seen, the accuracy of traditional algorithm fluctuates significantly and unstable. Compared with the proposed algorithm, the accuracy of proposed method is close to the maximum accuracy of traditional algorithm. It means that the proposed algorithm is more stable than the traditional algorithm using random initial centers.

IV. CONCLUSION

Data Mining is one of the most important tools in information retrieval. The rate of information exchange today is growing very fast and so there is a need for processing huge volume of data. To respond to this need we try and implement the vital algorithms used for data mining on a distributed or a parallel environment to reduce the operational resources and increase the speed of operation.

With the explosive growth of data and the development of information technology, traditional clustering algorithmscan not meet the demand. More and more traditional clustering algorithm are implemented on the cloud computing platform. In this paper, we design a K-means clustering with optimized initial centers based on data

dimensional density and introduce the implementation of our algorithm under the Map Reduce programming model. Through the comparison with proposed algorithm and traditional algorithm on performance we demonstrate that our algorithm effectively improve the stability of the algorithm. As for future work, we will pay attention to improve our MapReduce programming model to reduce the time-consuming of algorithm.

REFERENCES

[1] J. Dean, S. Ghemawat, "MapReduce: Simplifed Data Processing on Large Clusters", Communications of the ACM, January, 2008, Vol. 51, No.1, pp. 107-113

[2] Hadoopofcial site. http://hadoop.apache.org/.

[3] L. Guang, W. Gong-Qing, H. Xue-Gang, Z. Jing, L. Lian and W. Xindong, "K-means clustering with bagging and mapreduce," in Proceeding of the 2011 44th Hawaii International Conference on System Sciences. Washington, DC, USA: IEEE Computer Society, 2011, pp. 1-8

[4] R. M. Esteves, R. Pais and C. Rong, "K-means Clustering in the Cloud - A Mahout Test," in Processing of the 2011 IEEE Workshops of International Conference on Advanced Information Networking and Applications, 2011, pp.514-519.

[5] Zhang, YongJun, and Enxiu Cheng. "An Optimized Method for Selection of the Initial Centers of K-Means Clustering." Integrated Uncertainty in Knowledge Modeling and Decision Making. Springer Berlin Heidelberg, 2013, pp.149-156

[6] S. Ghemawat, H. Gobioff, S. Leung. "The Google file system," In Proc.of ACM Symposium on Operating Systems Principles, Lake George, NY, Oct 2003, pp 29–43.

[7] Fahim A M,Salem A M,Torkey F A, "An efficient enhanced k-means clustering algorithm" Journal of Zhejiang University Science A, Vol.10, pp:1626-1633,July 2006.

[8] J. Ekanayake, S. Pallickara and G. Fox, "Mapreduce for data intensive scientific analyses," in IEEE 4th International Conference on e-Science, December, 2008, pp. 277-284.

[9] C. Ranger, R. Raghuraman, A. Renmetsa, G. R. Bradski, and C.Koztrakis. "Evaluating MapReduce for Multi-core and Multiprocessor Systems," Proc. International Symposium on High-Performance Computer Architecture (HPCA), 2007, pp.13-24

[10] Prajesh P. Anchalia, Anjan K. Koundinya, Srinath N. K., "MapReduce Design of K-Means Clustering Algorithm," icisa, pp.1- 5, 2013 International Conference on Information Science and Applications.

[11] JuanyingXie, Shuai Jiang, weixinXie, XinboGao, "An Efficient Global K-means Clustering Algorithm", Journal of computers, vol.6, no.2, pp.271-279, February 2011

[12] Tan P N, Steinbach M, Kumar V, "Introduction to data mining", MA, USA: Addison-Wesley Longman Publishing Co., Inc. Boston, 2010

[13] P. Indira Priya, D.K.Ghosh,PhD, "K-means Clustering Algorithm Characteristics Differences based on Distance Measurement", International journal of computer applications, December 2012, pp. 12-14.

[14] Han Lingbo, Wang Qiang, Jiang Zhengfeng, "Improved k-means initial clustering center selection algorithm", Computer engineering and application, 2010, 46 (17), pp: 150-152.

