Advances in Engineering Research, volume 138
2nd International Conference on Machinery, Electronics and Control Simulation (MECS 2017)

An Optimization Algorithm of Selecting Initial Clustering Center in K - means

Tianhan Gao[1, a], Xue Kong[2, b,*]

[1]School of Software, Northeastern University, Shenyang 110819, China

[2]School of Software, Northeastern University, Shenyang 110819, China

[a]gaoth@mail.neu.edu.cn, [b]2513697428@qq.com

Abstract: The traditional stand-alone K-means clustering algorithm has the limitation of time consumption and memory overflow when dealing with large-scale data. Although this problem is solved with the help of MapReduce framework. However, the clustering accuracy effect is not stable due to the selection of initial clustering center. Therefore, this paper presents an algorithm for optimizing the initial clustering center in K-means by using several equal-scale sampling, calculating the local density and selecting the optimal initial clustering center. The experimental results show that the optimized algorithm shortens the clustering time and improves the accuracy and stability of clustering procedure in K-means.

Keywords: K-means, Initial clustering center, MapReduce, Local density

1. Introduction

With the rapid development of information technology, data scale is increasing continually. The large-scale data set for effective mining analysis can promote the development and progress of information technology. Clustering analysis is an important data processing technology which is widely used in data mining, information retrieval, and other research. The main goal is to divide the data set into similar category in the same subset under the guarantee that similarity between different subsets is small. K-means algorithm is one of the classical clustering algorithms in the partition method that owns the characteristics of simple operation and fast convergence[1]. With the continuous expansion of data size, the traditional stand-alone clustering algorithm is unable to meet the needs of large data clustering. The parallel K-means clustering algorithm[2] is thus proposed that combines the stand-alone k-means clustering algorithm and MapReduce to process large-scale data. However, the parallel K-means algorithm still has some flaws: (1) Pre-given the categories and easy to fall into local optimization; (2) Sensitive to edge and isolated points. Aiming at the above issues, a method is proposed to determine K initial clustering centers based on data density[3], which can select clustering center by calculating the density of the data to avoid random selection. Unfortunately, the method must calculate the density of all data, the workload and time consumption is immeasurable.

This paper presents an algorithm for optimizing the initial clustering center in K-means. First, equal-scale sampling is adopted to ensure that no duplication of the sample set. Second, the local density and average density of the sample is calculated and sorted where the larger local density of data is selected as the initial clustering center. Finally, the optimization algorithm is implemented under MapReduce framework. The experiment and performance analysis show that the proposed algorithm can effectively reduce the iterations and clustering time and owns high clustering stability.

The rest of the paper is organized as follows. Section 2 mainly introduces the related technology and work. Section 3 presents the optimization algorithm in detail. The experiment and performance analysis is given in section 4. The conclusions are made in section 5.

2. Related Technology and Work

2.1 K-means Clustering Algorithm

K-means clustering algorithm is a typical distance-based algorithm, which mainly employs distance as the evaluation criterion of similarity between data. A cluster is a group of data objects that describes a closer distance. The goal of the algorithm is to cluster the datasets into independent and compact clusters. The specific algorithm is as follows.

Algorithm 1: K-means algorithm

① K instances are randomly selected as initial clustering centers

② Calculate the remaining data to the center of each Euclidean distance and cluster to the nearest center point to form K clusters

③ Each cluster updates the center point by calculating the average of the distance between data

④ Utilize the error squared sum criterion function to compare the old and the new center, if the error change greatly, repeat step ② and step ③ until the center point no longer change or the fluctuation is not big, forming convergence

2.2 MapReduce Model

MapReduce is a distributed parallel programming model[4], whose core idea is "divide and rule". The large-scale data set is divided into a small data set owned by the main node under the management of the sub-nodes. Then the intermediate results are integrated to get the final result. The workflow of MapReduce is as below: (1) Get the input data for fragmentation and parse into key-value pairs as input to the Map function for partitioning. (2) Copy the key-value pairs output from the Map as input to the Reduce function for reduction.

2.3 Related Work

At present, there are a lot of improvements in the selection of initial clustering centers for K-means algorithm [5-7]. (1) The method of maximum and minimum distance is used to select the clustering center point then calculate the K value and find a reasonable center point. But it is easier to locate the isolated point or edge data as the center point which will affect the final clustering results. (2) Density - based K - means algorithm. The center of the data sample is selected by the data density, and the combination of the sub-cluster is introduced to avoid falling into local optimization. However, the time consuming is large. (3) Utilize the distribution density function of the data to avoid random selection. But the algorithm density is high. In order to improve the algorithm, we can find the requirement of k-means clustering center: (1) The selected data is representative of its surroundings. (2) The distance between cluster centers is the largest. Therefore, this paper proposes a method of data sampling combined with local density to determine the K value, which not only enhances the clustering effect but also shortens the clustering time effectively.

3. The Optimization Algorithm

3.1 Algorithm Design

The main idea is to extract samples without intersections through large-scale data sets. The local density and average density of samples are calculated by MapReduce with the data average density as the limit. Finally, find a local density data point as the center point.

3.1.1 Data Sampling

In this paper, let large data sets *D* be data sampling. The extracted data set is denoted by D_i and the data size is f_i, *N* times are taken. The sampling criterion is given by equation (1):

$$\boldsymbol{D_i \cap D_j = \varnothing,\ f_i \approx f_j \quad \text{and} \quad Nf_i \ll D} \tag{1}$$

Where $i \in \{1, N\}$, $i \neq j$ .The sampling process needs to meet the following conditions:1) each extracted dataset intersects an empty set; 2) the size of each extracted data is equal;3) *N* multiplies the size of the extracted data is much smaller than the total large data sets.

The calculation formula of sample size is shown in equation (2), where *e* is the size of the sample, *f* is the type of the data, and δ is the extraction probability sets $0.5 \le \delta \le 1$. Data types are more common about ten or so[8], $0 \le f \le 0.1$.

$$e = f * N * \delta \tag{2}$$

The improved data sampling approach avoids random selection of the center point and reduces the calculation of duplicated data, thus saving the system resources.

3.1.2 Local Density

The local density of data is described by the number of neighbors around the data. First, the local density of sample data is calculated and sorted quickly. The maximum local density of the data is the first center point. If the distance from the previous center is greater than $2D_e$, it is considered to be the center (D_e is the radius of the range around the intercepted data). The operation is ended until the local density of the searched data is smaller than the average local density (*Avg*).

Assuming that the data set *M* has *m* data, any data object has *n* attributes.Calculate the distance between data object *i* and *j* as (3):

$$D_{\text{ij}} = \sqrt{(i_1 - j_1)^2 + (i_2 - j_2)^2 + \cdots + (i_n - j_n)^2} \tag{3}$$

Find the local density of the data object *i* and the average local density as (4):

$$\rho_{\text{i}} = \sum_{j=1}^{m} \lambda(D_{ij} - D_e) \quad , \quad \text{Avg} = \frac{1}{m}\sum_{i=1}^{m} \rho_i \tag{4}$$

Where D_e is the range around the intercepted data *i*. If the data *j* is in the neighbor range of data *i*, then the value of λ is 1, otherwise the value of λ is 0.

3.2 Parallel Implementation under MapReduce

The optimized K-means algorithm is that each iteration calculation corresponds to a MapReduce computation, which includes calculating the distance of other data to the cluster center and the determination of the new cluster center. The algorithm is composed of initial cluster center selection, Map function, combine function, as well as Reduce function.

3.2.1 Algorithm Description

The related algorithms are depicted as Algorithm2~5.

Algorithm 2: initial cluster center selection

Input: data $= \{s_1, s_2, \cdots s_n\}, D_e$

Calculate $\rho_1, \rho_2, \cdots, \rho_n$, remember c_1=1

Quicksort $\rho_1, \rho_2, \cdots, \rho_n$

While $\rho > Avg$ if $dis(s_i, c_j) > D_e \times 2,\ j = 1, 2, \cdots, k-1$ c_k=*i*

Output: c_k

This algorithm calculates the local density and average density of the sample data and selects K initial clustering centers.

Algorithm 3: Map function

Input: $K=\{k_1,k_2,\cdots,k_k\}$, $D=\{x_1,x_2,\cdots,x_n\}$

$\text{int}\, i=1, \min dis = dis(x_1,k_1)$

$for(\text{int}\, i=2; i<=n; i++)$

$for(\text{int}\, j=1; j<=k; j++) if\,(dis(x_1,k_1) < \min dis) \min dis = dis(x_1,k_1)$

Output: $(x_1,x_2,\cdots,x_n) \Leftrightarrow (k_1,k_2,\cdots,k_k)$

The Map function calculates the distance from the remaining data to each center point and clusters to the nearest center point.

Algorithm 4: Combine function

Input: The key-value pairs generated by Map function

$if\,(< x_i \Leftrightarrow k_i >==< x_j \Leftrightarrow k_j >)$

$dis+=dis(x_i,x_j)$

$count++$

Output: The sum of the data object distances as *dis* and the number of objects as *count*.

The Combine function computes the distance of the data object with the common cluster center and records the number of objects.

Algorithm 5: Reduce function

Input: The output key-value pairs from Combine function

$while(i<k) k_i \Rightarrow dis_i, count_i$

$if\,(\frac{dis_i}{count_i} != k_i) newk_i = \frac{dis_i}{count_i}$

Output: New clustering centers

The Reduce function calculates the mean value of each cluster as the new cluster center point and utilizes the criterion function to judge whether converge or not.

4. Experiment and Performance Analysis

We build the Hadoop cluster environment as shown in Table1. The experimental data set is selected in the UCI iris dataset. There are150 sample data, which are divided into 3 class. Each class has 50 packets and each data contains 4 attributes. The accuracy and time consumption are analyzed among PK-Means[3], DK-Means[4], and the proposed algorithm(OK-Means).

Table 1 Node Configuration

Master/Slave:
CPU: Intel(R)Core(TM)i7-4790 3.60GHz
Memory: 8GB
Hard disk: SATA 500G
Operating system: Centos6.5
Experimental platform and development tools: Hadoop-2.7.1，MyEclipse
JDK environment：Jdk1.7

4.1 Accuracy and Time Consumption Analysis

The accuracy is defined as the percentage of the number of data objects that compute the correct number of clusters. The time consumption is the running time of the algorithm.

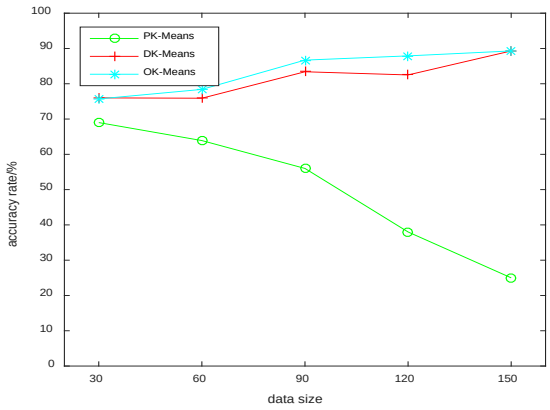

Fig. 1 Comparison of accuracy

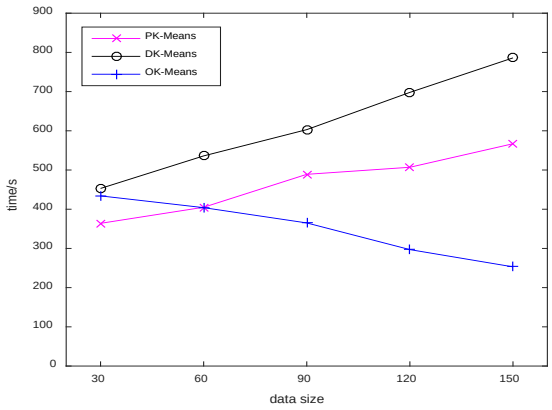

Fig. 2 Comparison of time consumption

The compared results are shown in Fig1 and Fig2. It can be seen that the clustering effect of the optimized K-means algorithm(OK-Means) is better than the other two algorithms with the increase of data size.

5. Conclusions

In terms of the accuracy and stability of clustering large-scale data in K-means, an optimized initial clustering center selection algorithm is proposed in this paper under MapReduce framework. The initial clustering center is optimized by several equal-scale sampling methods, and the calulation of local density of the data. The error square sum function is introduced to judge the fluctuation of the center point in the iterative process. Compared with other existing approachs, the proposed optimization algorithm can reduces the iterations and time consumption significantly when clustering large-scale data, and owns better stability and accuracy.

6. Acknowledgments

This work was financially supported by National Natural Science Foundation of China (No. 61402095).

References

[1] Jigui S, Jie L, Lianyu Z, et al, Research on clustering algorithm[J], Journal of Software, 2008, 19(1):48-61.
[2] Weizhong Z, Huifang M, Yanxiang F, et al, Research on Parallel K - means Clustering Algorith m Based on Cloud Computing Platform Hadoop[J], Computer Science, 2011, 38(10):166-168.
[3] Weiben Z, Yuexiang S, Optimization Algorithm of K - means Clustering Center Based on Densi ty[J], Application Research of Computers, 2012, 29(5):1726-1728.
[4] LI Jian-Jiang, J Cui, D Wang, L Yan, et al. Survey of MapReduce paraller programming model [J], Tien Tzu Hsueh Pao/acta Electronica Sinica, November 2011,39(11):2635-2642.
[5] Boukhdhir A, Lachiheb O, Gouider M S. An improved mapReduce design of kmeans for clusteri ng very large datasets[C]// Ieee/acs, International Conference of Computer Systems and Application s. 2015.
[6] Anchalia P P. Improved MapReduce k-Means Clustering Algorithm with Combiner[C]// Uksim-Amss, International Conference on Computer Modelling and Simulation. IEEE, 2014:386-391.
[7] Cui X, Zhu P, Yang X, et al. Optimized big data K-means clustering using MapReduce[J]. The J ournal of Supercomputing, 2014, 70(3):1249-1259.
[8] Guha S, Rastogi R, Shim K. CURE: an efficient clustering algorithm for large databases[C]// AC MSIGMOD International Conference on Management of Data. ACM, 1998:73-84.