J Grid Computing (2020) 18:239–250
https://doi.org/10.1007/s10723-019-09503-0

Improved K-Means Clustering Algorithm for Big Data Mining under Hadoop Parallel Framework

Weijia Lu

Received: 24 April 2019 / Accepted: 6 December 2019 / Published online: 20 December 2019

Abstract In order to improve the accuracy and efficiency of the clustering mining algorithm, this paper focuses on the clustering mining algorithm for large data. Firstly, the traditional clustering mining algorithm is improved to improve the accuracy, and then the improved clustering algorithm is parallelized to improve the efficiency. In order to improve the accuracy of clustering, an incremental K-means clustering algorithm based on density is proposed on the basis of K-means algorithm. Firstly, the density of data points is calculated, and each basic cluster is composed of the center points whose density is not less than the given threshold and the points within the density range. Then, the basic cluster is merged according to the distance between the two cluster centers. Finally, the points that are not divided into any cluster are divided into the clusters nearest to them. In order to improve the efficiency of the algorithm and reduce the time complexity of the algorithm, the distributed database was used to simulate the shared memory space and parallelize the algorithm on the Hadoop platform of cloud computing. The simulation results show that the clustering accuracy of the proposed algorithm is higher than that of the other two algorithms by more than 10%.

W. Lu (✉)
Information Center, Affiliated Hospital of Nantong University, Nantong, Jiangsu, China
e-mail: tdfylwj@126.com

Keywords Improved k-means clustering algorithm · Big data mining · Hadoop parallel framework · Shared storage space · Parallel computing · Parallelization · Distributed database

1 Introduction

Using unsupervised methods such as clustering to mine hidden patterns in big data is one of the effective means of data mining, which has been widely used in tasks such as data modeling and data preprocessing [1]. Clustering is an unsupervised learning method, which is used to divide data into different clusters. Data in each cluster are similar to each other, while data in different clusters are obviously different. When data is not marked, clustering algorithm can be used for data mining, which can build independent data model or lay a foundation for other data modeling and analysis, and is widely applied in many research fields such as social network, bioinformatics and image processing [2, 3]. K-means clustering algorithm is one of the most widely used clustering algorithms at present. This algorithm adopts the idea of partition and uses the criterion of least mean square error to divide data into different clusters. It has advantages of unsupervised and efficient in performing, and has better clustering effect when the data obeying the normal distribution. However, when it comes to big data mining applications, the efficiency of k-means clustering algorithm is difficult to meet the requirements [4].

There are some improvements to the problems in K-means clustering applications. For example, Literature [5] discussed an algorithm for processing K-means in Hadoop by changing data sets and clustering centers. Then we compare parallel and sequential execution, keeping other factors unchanged. The experimental results show that the algorithm can effectively deal with large data sets in Hadoop environment. Literature [6] aimed at the research of DBSCAN clustering of AI data. Combining with Hadoop platform, a DBSCAN clustering algorithm for large-scale AI data based on Hadoop platform is proposed. Under the framework of MapReduce parallel computing, DBSCAN clustering algorithm makes full use of Hadoop's advantages in processing large data through HDFS distributed storage and MapReduce distributed computing package, which greatly improves the efficiency of the algorithm. The literature [7] proposed an algorithm to solve the random selection of the initial clustering center, which may causes unreasonable clustering result, this algorithm is developed by combining with the correlation of the underlying data structure, which can improve the choice of initial clustering center strategy, thereby improving the convergence speed of clustering and the clustering efficiency. The literature [8] proposed two parallel versions of the OClustR algorithm, specifically tailored for GPUs and multi-core CPUs, which enhance the efficiency of OClustR in problems dealing with a very large number of documents. However, this algorithm is still lower in clustering efficiency in data mining applications.

In order to improve the accuracy of data mining algorithm and reduce the time complexity, this paper synthesizes the advantages of various literature algorithms, and through the improvement of existing algorithms, proposes an incremental K-means clustering algorithm based on density, in order to improve the efficiency of the algorithm and reduce the time complexity of the algorithm. Parallelization of the algorithm is carried out and simulation experiments are carried out on the algorithm.

2 Big Data Mining Technology

2.1 Definition of Big Data Mining

Data mining refers to the data processing process of discovering existing valuable information from the data set, and its purpose is to "panning for gold" from the data. The process of data mining is composed with several steps. Figure 1 shows the general steps of data mining.

The specific tasks of each step are as follows:

(1) Data selection: select correct and valid data from the data set. Because the data is generated randomly, the data set must contain a large number of interfering data. Data interference will have a negative impact on mining results, so the selection of data related to mining targets is an indispensable step in effective data mining.
(2) Data preprocessing: conduct simple normalization processing of data, so that data can meet the processing requirements of data mining algorithm. Data preprocessing mainly includes data variable conversion, missing value processing, data normalization and data attribute selection.
(3) Pattern discovery: pattern discovery is the core step of data mining. Pattern discovery refers to the selection of specific data mining algorithms according to the purpose of data mining and the discovery of knowledge from data sets after data selection and data preprocessing.
(4) Pattern assessment: the purpose of pattern assessment is to evaluate the quality of discovered knowledge. Mode evaluation mainly includes two methods, judging whether the accuracy of knowledge discovered meets the given standard and whether the knowledge discovered conforms to the expected result.
(5) Knowledge representation: the purpose of knowledge representation is to present the discovered knowledge intuitively. Knowledge can be visually presented to the user through visualization techniques or other means that enable the user to

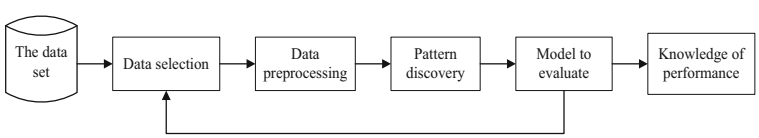

Fig. 1 General steps of data mining

understand and participate in the pattern discovery process to a certain extent.

With the popularity of smart devices (sensors, medical devices and smart phones) [9, 10] and the increase of smart buildings, data volume is growing rapidly, and the measurement unit of these data has reached PB or even TB. In addition, semi-structured and unstructured data account for the majority of the whole data in the rapidly growing. Therefore, how to mine useful knowledge from massive, high-speed and heterogeneous big data sets is a major challenge. Big data mining technology describes the new generation of data mining technology and framework. Big data mining refers to the process of discovering value from massive and diverse data sets by using high-speed acquisition, discovery and analysis technology. Due to the complexity of big data itself, the preferred method of big data mining should be in the cloud computing environment.

2.2 The Challenge of Big Data Mining

Big data is both a once-in-a-lifetime opportunity and a huge challenge for enterprises. The speed of the development of contemporary enterprises and the big data created by the digital world require the use of new methods to mine useful knowledge from big data [11, 12]. Potential business value which effectively mining from big data can help enterprises to formulate, improve and reformulate business plans, find operational faults, simplify supply chain, thereby better understand customer needs, develop new products, and win new development opportunities in the wave of big data. Although enterprises have a clear understanding of the usefulness of big data, they still face huge challenges in extracting useful knowledge from big data [13]. The main problems of big data mining are as follows.

(1) Diversity of data types: since semi-structured data and unstructured data account for 80% to 90% of the whole large data set, big data mining algorithm must be able to process data sets containing structured data, semi-structured data and unstructured data.
(2) The problem of large-scale data sets: large scale is one of the characteristics of big data. Therefore, big data mining algorithms should be able to effectively mine large-scale data within an acceptable time and space.
(3) (3)High-speed stream data problem: the maximization of benefits can only be achieved by mining high-speed stream data in a specific time. Therefore, the big data mining algorithm should be able to effectively complete the mining of high-speed stream data in a specific time with the shortest time.
(4) Evaluation of data mining results: given a large data set, due to its large amount of data and diverse data types, the distribution characteristics of the entire data are often unknown, which leads to many difficulties in the evaluation of the results of big data mining.
(5) Privacy protection: the era of big data has brought many challenges to traditional data analysis technologies, such as privacy protection. Therefore, privacy protection in the era of big data is facing the dual pressure of talents and technology.

3 Improved K-Means Clustering Algorithm

3.1 K-Means Algorithm

K-means clustering algorithm is a classic and commonly used clustering algorithm. Its main operation is to find the mean of the elements in each cluster subset and set it as the cluster center. The main method of the algorithm is to divide the element set into different clusters after repeated iterations, and apply the criterion function of the evaluation cluster classification. When the function is optimal, the iteration is terminated [14].

3.1.1 K-Means Algorithm Flow

The flow chart of k-means algorithm is as follows:

Step 1: Randomly select K objects from the elements as the initial cluster center:

$$S_j(I), j = 1, 2, \cdots, K \tag{1}$$

Step 2: Get the distance between all elements and $S_j(I)$ in the cluster:

$$D\left(x_i, S_j(I)\right), i = 1, 2, \cdots, n; j = 1, 2, \cdots, K \quad (2)$$

Then the object will be allocated to the nearest cluster, if satisfied

$$D(x_i, S_k(I)) = \min\{D(x_i, S_k(I))\} \quad (3)$$

Then $x_i \in C_k$

Step 3: The error square sum criterion function can be obtained:

$$J_c = \sum_{j=1}^{C} \left\| x_k^j - S_j(I) \right\|^2 \quad (4)$$

Step 4: If $|J_c(I) - J_c(I-1)| < \xi$, stop and output the cluster result. Otherwise, continue to iterate, calculate the clustering center $S_j(I) = \frac{1}{n} \sum_{j=1}^{n_j} x_k^j$, and go to Step2 until $|J_c(I) - J_c(I-1)| < \xi$.

The flow chart of k-means algorithm is as follows Fig. 2:

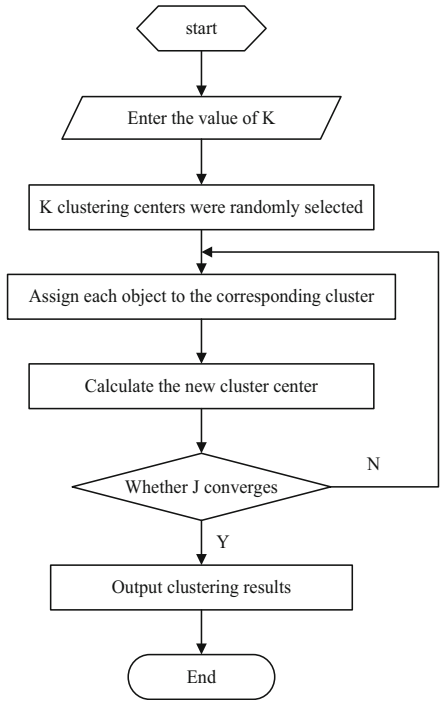

Fig. 2 The flow chart of k-means clustering algorithm

3.1.2 The Shortcomings of the K-Means Algorithm

(1) The K value of the clustering number should be selected in advance. However, when this algorithm is applied, the selection of this K value is very difficult to estimate. The uncertainty of the clustering number K is one of the disadvantages of k-means clustering algorithm.
(2) Center selection also plays a decisive role in the initial cluster selection. In the k-means algorithm, an initial classification should also be obtained by relying on the initial clustering at first, and then subsequent clustering processing should be carried out for the classification. If the required selection in the initial classification is biased, the algorithm will easily wander in the local optimal solution or the result is wrong [15, 16].
(3) It is susceptible to noise data. In each recalculation of the k-means algorithm for the cluster, the mean value is obtained from all data points in each cluster. If there are noise points in the data set, the clustering result will be easily affected by these noise points, leading to different clustering results. Therefore, this algorithm is very sensitive to noise data [17].
(4) It is not competent for clustering problems with a large amount of data. From the steps of this algorithm, we know that this algorithm needs to conduct sample division and debugging again and again, and the new data after debugging is obtained again and again. If the data volume is very large, the cost of the algorithm is very high.

3.2 An Improved K-Means Clustering Algorithm for Big Data Mining

It is a hot topic to improve and parallelize the traditional clustering algorithm. The main research content of this paper is to put forward the improved clustering algorithm on the basis of k-means clustering algorithm,

parallelize the improved clustering algorithm, and finally apply the improved clustering algorithm to the actual scene.

3.2.1 The Basic Idea of the Algorithm

For the d-dimensional mixed attribute data set $X = \{x_1, \cdots, x_i, \cdots, x_n\}$ with n nodes, where $x_i \in R_d$, the basic idea of the clustering algorithm is as follows:

Firstly, a data set Y, $Y \subset X$, is randomly selected from the set X, and the number of data in Y is much smaller than n; for any point $y_i \in Y$, calculate its neighborhood Eps, which is equal to the mean value of distance between y_i and other data points in set Y..

Secondly, for any point $y_i \subset Y$, if the number of data points contained in the Eps neighborhood of y_i is not less than a given threshold minPts, that is, the density of y_i is not less than minPts, then y_i and its points within the Eps radius are combined into one basic Cluster; In the neighborhood of y_i, select the data points furthest from y_i and record their distance as D^i. Again, for any two clusters i and j, if the distance between their center points y_i and y_j is not greater than 2*max(D^i,D^j), the two clusters are merged. Calculate the center point y_k of this new cluster, find the data points farthest from y_k, and record their distance as D^k.; repeat this step until no more clusters are merged. Then, for the point y_m in set Y that is not divided into arbitrary clusters, y_m is divided into clusters having the smallest dissimilarity-degree, and the center point of the cluster is updated at the same time.

Finally, for any of the remaining points x, $x \in X - Yx \subset X - Y$, calculate the distance between x and all cluster center points, find the minimum distance $D_{\min}$, and calculate the distance mean D_{ave}.. If $D_{\min}$ is not greater than D_{ave}, x is divided into the cluster closest to it, otherwise a new cluster is generated with x as the center point, update the center point of the cluster when its number of data points changes.. Repeat this step until all data has been processed.

3.2.2 Calculation of Data Dissimilarity-Degree

The traditional clustering algorithm that uses Euclidean distance to calculate the dissimilarity-degree between data cannot effectively process the data set with the following conditions: data dimension is greater than 20;. A high-dimensional mixed attribute data set containing continuous, discrete, and text attributes in the attributes of the data [18, 19] . Therefore, under the premise of not reducing the data dimension, using the data normalization method to calculate the dissimilarity-degree between the data is one of the necessary conditions for the traditional clustering algorithm to effectively process the high-dimensional mixed attribute data set.

Classification of Data Attributes From the perspective of clustering, the attributes of the data can be divided into numerical attributes, binary attributes, subtyped attributes and ordinal type attributes. The four types of attributes of the data are described in detail below:

(1) Numeric attributes: attribute types represented by numerical values, which are generally divided into continuous attributes and discrete attributes. An attribute that can take any value within a given range is called a continuous attribute, and its value is continuous, and any two values can be infinitely divided [20, 21] . An attribute that can only take a natural number or an integer unit within a given range is called a discrete attribute. For example, you can abstract each person into a data point, which contains three attributes: age, height, and weight. Age is a discrete attribute, and height and weight are continuous attributes.

(2) Binary Attributes: attributes that can only take any one of the two values given are called binary attributes. Two selectable values of a binary attribute are called two states, generally represented by 0 and 1. For example, in the circuit, 0 is used to indicate a low state and 1 is a high state.

(3) Subtype attribute: subtype attribute is a generalization of a binary attribute. The number of selectable values of a subtype attribute is greater than two. Different values represent different classifications. For example, there are seven basic colors, namely, red orange, yellow, green, blue, and purple, and each of the seven basic colors is represented by 1 to 7 respectively [22, 23] . The value of the subtype attribute is determined in advance, and there is no order size between the attribute values.

(4) Ordinal attribute: the selectable value of the attribute is not less than two. Different values represent different levels, and there is a level between the values. For example, university teachers have three levels of professors, associate professors, and lecturers. There are high and low levels between the three levels.

Since the values of binary, sub-type and ordinal attributes cannot be measured by size, simply using the Euclidean distance formula cannot effectively calculate the dissimilarity-degree between data.

Calculation of Data Dissimilarity-Degree The calculation of the dissimilarity-degree between data is the key to the clustering algorithm. The quality of the clustering result depends on the calculation method of the dissimilarity-degree between the clustering algorithm and the data. The calculation of the dissimilarity between the high-dimensional mixed attribute data, firstly, normalize all the data attributes, and then calculate the degree of dissimilarity between the data. The so-called normalization means that the value range of all attributes is mapped in the range [0, 1], thereby, there is a certain comparability and computability between the same attributes, avoiding the problem of large numbers "eat" decimals [24–26].

Based on the existing research results, this paper improves the original data normalization method and presents a simplified data normalization method.. For a d-dimensional mixed attribute data set $X = \{x_1, \cdots, x_i, \cdots, x_n\}$ containing n points, here $x_i \in R_d$, each attribute dissimilarity-degree is calculated as follows:

(1) Numeric attributes: Numeric attributes are further divided into continuous attributes and discrete attributes. The formula for calculating the dissimilarity-degree of continuous attributes is as follows:

$$d_{x_i,x_j}(f) = \frac{|x_{if} - x_{jf}|}{\max_f - \min_f} \tag{5}$$

Here $d_{x_i,x_j}(f)$ represents the dissimilarity-degree between the *f*th attributes of data points x_i and x_j. In formula (5), *f* and G respectively represent the maximum and minimum values that can be obtained for the *f*th attribute in the entire data set, and x_{if} represents the value of the *f*th attribute of point x_i.

The dissimilarity-degree of discrete attributes can be calculated according to the specific case using the classification type attribute or the ordinal type attribute dissimilarity-degree calculation formula.

(2) Ordinal type attribute: Assuming that the desirable field value of the ordinal type attribute *f* is a set of ordered sequences of 1, 2...M, the formula for calculating the ordinal type attribute is as follows:

$$d_{x_i,x_j}(f) = \frac{|x_{if} - x_{jf}|}{M_f - 1} \tag{6}$$

In the formula (6), M_f represents the maximum value that *f* can take.

(3) Binary and subtype attributes: Since the values of binary and subtype attributes only represent different states or different categories therefore, their formulas are as follows:

$$d_{x_i,x_j}(f) = \begin{cases} 0 & x_{if} \neq x_{jf} \\ 1 & x_{if} = x_{jf} \end{cases} \tag{7}$$

It is 1 when the *f*th attribute values of x_i and x_j are the same, otherwise 0.

According to formulas (5), (6) and (7), the calculation of the dissimilarity-degree of any two d-dimensional mixed attribute points x_i and x_j is as follows:

$$d(x_i, x_j) = \frac{\sum_{f=1}^{d} \delta_{x_i,x_j}^{(f)} \cdot d_{x_i,x_j}^{(f)}}{\sum_{f=1}^{d} \delta_{x_i,x_j}^{(f)}} \tag{8}$$

Here $\delta_{x_i,x_j}^{(f)}$ represents the indication value, it is equal to 0 if and only if the *f*th attribute of x_i or x_j does not exist or both are 0, otherwise the value is 1; $\delta_{x_i,x_j}^{(f)}$ is a calculated dissimilarity-degree betwwen x_i and x_j, which is calculated according to their *f* attribute and the formula (5),(6) and (7).

Normalizing the data not only avoids the problem that large number"eat"decimals when calculating the dissimilarity with the Euclidean formula, moreover, the clustering algorithm can effectively process the data set of high-dimensional mixed attributes without reducing the data dimension.

3.2.3 Description of the Algorithm

K-means clustering algorithm is mainly divided into two parts. The first part conducts density-based

clustering on a small part of data in the entire data set. In the second part, based on the clustering results of the first part, the remaining data are divided into corresponding clusters according to the idea of incremental clustering. The overall flow of the algorithm is shown in Fig. 3 below.

Combining the basic idea of K-means clustering algorithm with the flow chart of K-means clustering algorithm, using formula (8) as the calculation formula of dissimilarity-degree, the proposed pseudo code of K-means clustering algorithm is as follows:

Algorithm 1：K-means clustering algorithm

Input: Data set X, density threshold minPts

Output: Several clusters of arbitrary shape

Step:

Randomly take a small part of the data set Y, $Y \subset X$ from the set X, and calculate the Eps at the same time;

List centerPoint = null， clusters = null;

for i = 1 to |Y| // |Y| represents the size of the data set Y, and the for loop is used to calculate the density of the data points.

{

p = getPoint（*i*, Y）； //Take the i-th record from data set Y

if centerPoint is empty then

p is placed in the centerPoint as the center point of a cluster；

else

if the dissimilarity-degree between p and the cluster center point is not greater than Eps

Divide p into clusters that are least different from each other；

else

p is placed in the centerPoint as the center point of a cluster；

}

for i = 1 to centerPoint.size() //Points with a density not less than minPts and points within the density range are combined into clusters

{

p = centerPoint.get（i）； //Get the ith center point from centerPoint

if the density of p is not less than minPts, then

Take p as the center, the points in the Eps neighborhood of p were combined into clusters, and the D^i of the points farthest from p in the cluster was calculated, and p and its cluster-related attributes were put into clusters;

}

for i = 1 to centerPoint.size() //Points with a density no less than minPts and points within their density range are combined into clusters

{

p = centerPoint.get（i）；

If the density of p is not less than minPts, then

Take p as the center, the points in the neighborhood of p point Eps were combined into clusters, and the D^i of the points farthest from p in the cluster was calculated, and p and its cluster-related attributes were put into clusters.

}

do {

flag = false；//Flag variable that indicates whether a cluster has been merged

for i = 1 to clusters.size()

{

for j = i +1 to clusters.size()

{

if the dissimilarity-degree between the center points of cluster I and j is no more than 2* Max（D^i， D^j）

{

Merge clusters I and j， flag = true；break；

}

}

if flag == true

break；

}

}while（flag）

The points in the set Y that are not divided into any clusters are divided into the clusters with the closest dissimilarity-degree to them；

Calculate the mean distance D_{ave} between the center points of the cluster；

for i = 1 to |X-Y| //The points in the set x-y are divided into clusters with the least dissimilarity-degree with them

{

p = get（X-Y，i）；

The dissimilarity-degree between p and each cluster was calculated, and the value $D_{\min}$ of the minimum dissimilarity-degree was obtained；

if $D_{\min}$ is greater than the D_{ave}

Take p as the center point, generate a new cluster and add it to cluster；

else

Divide p into clusters with the least dissimilarity-degree with it；

Recalculate the value of D_{ave}；

}

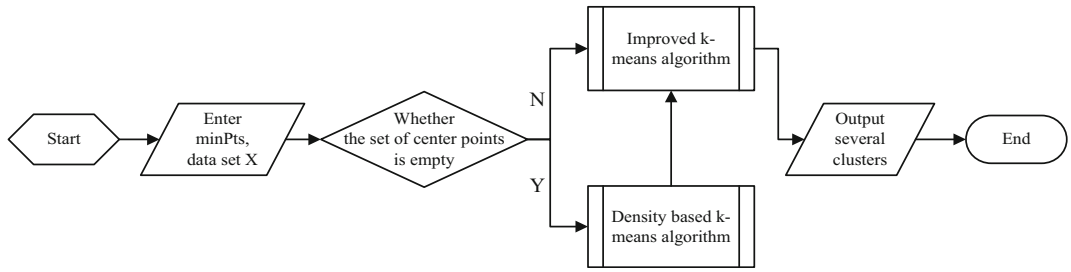

Fig. 3 Flow chart of improved K-means clustering algorithm

The proposed clustering algorithm can find a number of clusters of arbitrary shapes. In terms of accuracy and time complexity, the algorithm is not sensitive to the data input order and minPts parameters. Using formula (8), the algorithm can effectively process the data set of high-dimensional mixed attributes.

4 K-Means Clustering Comparison Experiment Based on Hadoop Cloud Platform

The k-means algorithm is not competent for clustering problems with a large amount of data. According to the steps of the proposed algorithm, we know that the algorithm needs to conduct sample division and debugging again and again, and the Hadoop distributed cloud platform after debugging is used to analyze the data.

4.1 Hadoop Cloud Platform Environment Construction

The environment required to build Hadoop cloud platform is as follows:

4 servers (or PCS)
Linux operating system: Centos6.5
Java environment: JDK 1.7
Hadoop version: Hadoop −2.6.0

The platform is set up by four servers, one as master node and three as slave node. Linux based on Centos6.5 is installed on each server; Server configuration information is shown in Table 1:

The steps of Hadoop configuration mainly include:

1. Unzip and install the Linux version of JDK 1.7 on each node for JAVA compilation.
2. SSH is configured between each node to make it convenient for master to login slave nodes without secret keys to achieve communication between nodes.
3. Unzip and install Hadoop and configure the files for Hadoop.
4. After configuring Hadoop, you should first format the system through the formatting command, and then start the command sbin/start-all. sh, you can see whether Hadoop is set up through the JPS process and the end of the browser.

4.2 Mahout Data Mining Tool

Once the Hadoop cloud platform is set up, you can install and configure Mahout data mining tools on the platform. In Mahout, a large number of classical algorithms are encapsulated, K-means is one of them. The main steps for Mahout installation and configuration are:

1. Download the Mahout

It can be downloaded from the official Mahout website. The version used in this article is: Mahout-0.10.0

2. Unzip the files
3. Configure environment variables
4. Configure the Hadoop environment variables required for Mahout

Table 1 Hadoop platform configuration information

The name of the node	The processor	IP
Master	1	192.168.1.100
slave	1	192.168.1.101
slave	1	192.168.1.102
slave	1	192.168.1.103

Table 2 Comparison of time consumption between Hadoop platform and single machine classical k-means algorithm

The data set	Number of data sets	Comparison of k-means running time(s)	
		Single machine	Hadoop platform
data set 1	15,000	7.9	7.2
data set 2	30,000	20.5	11.6
data set 3	45,000	43.2	21.4

5. Verify that Mahout is installed successfully

Enter Mahout on the terminal. If several algorithms appear, success is indicated.

4.3 Hadoop Cloud Platform and Stand-Alone Comparison Experiment

After setting up the Hadoop platform, conduct k-means clustering analysis on the data with the help of Mahout data mining tool, extract 10% KDD99 data set, and divide it into three subsets for experiments, each with 15,000, 30,000 and 45,000 records. Table 2 shows the comparison of the running time of data on Hadoop platform and single machine classical k-means algorithm.

In order to analyze the experimental results more intuitively, Fig. 4 shows the time comparison between the Hadoop cloud platform and the stand-alone classical k-means algorithm.

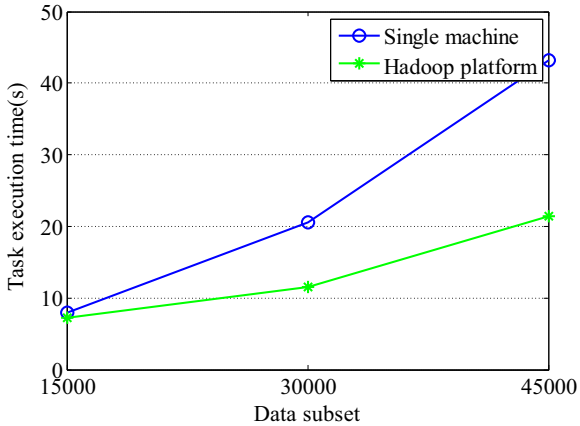

Fig. 4 Time-consuming comparison of Hadoop cloud platform and stand-alone classic K-means algorithm

According to Fig. 4, k-means clustering algorithm based on Hadoop cloud platform is more efficient and less time-consuming than the typical method of single machine. In the results of dataset 2 and dataset 3 with more data, the gap is becoming larger and larger. This is because the two methods have different processing methods for data. The classic method of single machine is serial processing, which has a large time complexity, while Hadoop processes data in parallel, which greatly shortens the operation time.

4.4 Compare with Other Algorithms

In this paper, the DBSCAN algorithm [6]、the classical k-means clustering algorithm (abbreviated as KM) [8] and the improved k-means clustering algorithm (abbreviated as MKM) in literature [9] were selected for comparison experiment. The big data set used in the experiment is the HIGGS data set, which contains 11 million records, each record having 28 attribute characteristics. The common parameter settings of the three comparison algorithms are the same, specifically, the number of clusters k = 2, and the upper limit of iteration times t_{max}= 1000. For different errors, the changes in clustering time and clustering accuracy of different algorithms are tested, and the results are shown in Figs. 5 and 6 respectively.

As can be seen from Figs. 5 and 6, with the increase of error, the clustering time of the three algorithms will decrease, and the clustering accuracy will also decrease. Because the larger the error is, the faster the clustering algorithm converges, and the clustering time decreases accordingly. However, rapid convergence may also cause that the solution of iterative search is not the

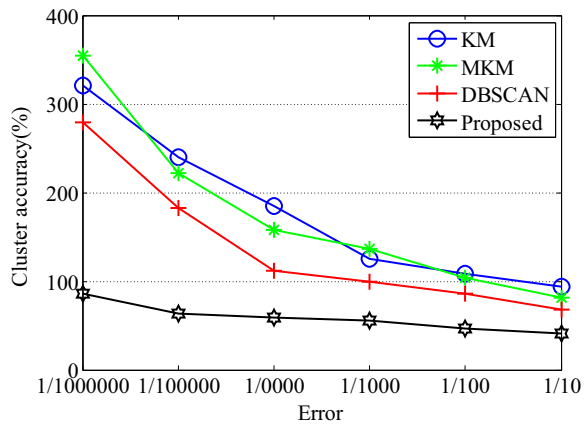

Fig. 5 The curve of clustering time varying with error

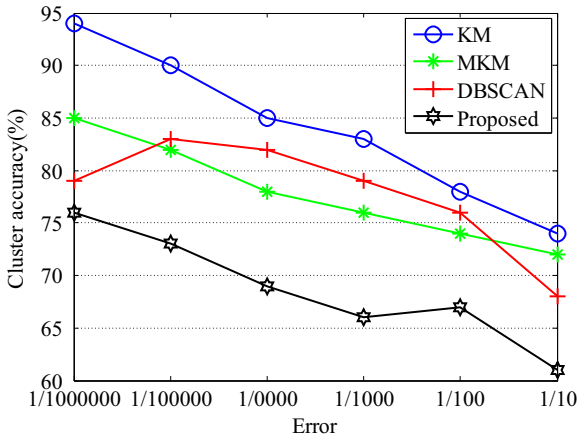

Fig. 6 The variation curve of clustering accuracy with error

optimal solution, so the clustering accuracy may decline. The algorithm in this paper improves the operation efficiency of KM algorithm obviously, because the algorithm in this paper distributes the clustering of big data to multiple nodes, thus reducing the time of clustering. In addition, the clustering time of the algorithm in this paper is also less affected by errors, so the curve is relatively flat.

Contrast in Fig. 6 clustering accuracy curves of four kinds of algorithm, visible clustering accuracy of the four kinds of algorithms were similar, and the error is bigger when the clustering algorithm in this paper accuracy slightly higher than the other two algorithms, this is because the clustering algorithm in this paper will be divided into big data blocks, each block of data when using weighted k-means clustering and k-means algorithm improve the clustering accuracy weighted fusion, weaken the error parameter's influence on the overall clustering accuracy. In conclusion, under the premise of maintaining the clustering accuracy, the algorithm in this paper greatly improves the operation efficiency of k-means clustering algorithm in big data clustering.

5 Conclusion

In order to improve the efficiency of clustering mining algorithm for large data, an incremental K-means clustering algorithm based on density is proposed on the basis of K-means algorithm, and the algorithm is designed in parallel. The difference degree of discrete attributes can be calculated according to the specific situation by using the calculation formula of the difference degree of classification attributes or ordinal attributes. This algorithm divides large data into block clustering, and uses weighted K-means and weighted fusion K-means algorithm to improve clustering accuracy, which weakens the influence of error parameters on the overall clustering accuracy. Experiments show that the improved K-means clustering algorithm can achieve better acceleration ratio with the help of cloud platform when dealing with large data. In the improved algorithm, the selection of data sampling and initial two clustering centers is an important factor affecting the final clustering effect. These two steps take a certain amount of time. Therefore, how to sample data quickly and how to select initial clustering centers quickly need further research. In addition, due to the limitation of experimental conditions, this experiment only uses four computers, lacking of large-scale cluster testing, and the testing of larger data sets needs further verification.

Acknowledgements This work was supported by the Nantong natural science foundation project (No. MS12017026-3).

References

1. Cai, Z., Lee, I., Chu, S.C., et al.: SimSim: a service discovery method preserving content similarity and spatial similarity in P2P mobile cloud. J. Grid Comput. **17**(3), 1–17 (2019)
2. Saeed, Z., Abbasi, R.A., Maqbool, O., et al.: What's happening around the world? A survey and framework on event detection techniques on twitter. J. Grid Comput. **17**(2), 1–34 (2019)
3. Righi, R.D.R., Lehmann, M., Gomes, M.M., et al.: A survey on global management view: toward combining system monitoring, resource management, and load prediction. J. Grid Comput. **17**(9), 1–30 (2019)
4. Salabat, K., Amir, K., Muazzam, M., et al.: Optimized Gabor feature extraction for mass classification using cuckoo search for big data E-healthcare. J. Grid Comput. **17**(2), 239–254 (2019)
5. Bandyopadhyay, S.S., Halder, A.K., Chatterjee, P., et al.: HdK-means: Hadoop based parallel K-means clustering for big data IEEE Calcutta Conference, pp. 452–456 (2018)
6. Chen, Z., Guo, J., Liu, Q.: DBSCAN algorithm clustering for massive AIS data based on the Hadoop platform 2017 International Conference on Industrial Informatics - Computing Technology, Intelligent Technology, Industrial Information Integration (ICIICII). IEEE Computer Society, pp. 25–28 (2017)
7. Ye, K., Jiang, X., He, Y., et al.: vHadoop: a scalable Hadoop virtual cluster platform for mapreduce-based parallel machine learning with performance consideration. IEEE International Conference on Cluster Computing Workshops, pp. 152–160 (2012)

8. Soler, L.J.G., Suárez, A.P., Chang, L.: Efficient overlapping document clustering using GPUs and Multi-core systems. Iberoamerican Congress on Pattern Recognition Ciarp, pp. 264–271 (2014)
9. Bousbaci, A., Kamel, N.: A parallel sampling-PSO-multi-core-K-means algorithm using mapreduce. International Conference on Hybrid Intelligent Systems, pp. 129–134 (2015)
10. Kim, J., Kim, M.H.: An efficient parallel processing method for skyline queries in MapReduce. J. Supercomput. **74**(2), 1–50 (2018)
11. Suresh Kumar, N., Thangamani, M.: Multi-ontology based points of interests (MO-POIS) and parallel fuzzy clustering (PFC) algorithm for travel sequence recommendation with Mobile communication on big social media. Wirel. Pers. Commun. **103**(11), 1–20 (2018)
12. Tripathi, A.K., Sharma, K., Bala, M.: Dynamic frequency based parallel k-bat algorithm for massive data clustering (DFBPKBA). Int. J. Syst. Assur. Eng. Manag. **9**(1), 1–9 (2018)
13. Xing, X., Shimada, A., Taniguchi, R.I., et al.: Coupled dictionary learning and feature mapping for cross-modal retrieval. IEEE International Conference on Multimedia & Expo, pp. 1–6 (2015)
14. Wang, J., Li, G., Peng, P., et al.: Semi-supervised semantic factorization hashing for fast cross-modal retrieval. Multimed. Tools Appl. **76**(3), 1–19 (2017)
15. Yonggui, W., Cui, P., University L T: An efficient K-means parallel algorithm based on MapReduce. J. Liaoning Tech. Univ. **36**(11), 1204–1211 (2017)
16. Xiao-Yu, L.I., Li-Ying, Y.U., Lei, H., et al.: The parallel implementation and application of an improved K-means algorithm. J. Univ. Elect. Sci. Technol. China. **46**(1), 61–68 (2017)
17. Gao, B., Qin, Y., Xiao, X.M., et al.: K-means clustering analysis of key nodes and edges in Beijing subway network. Jiaotong Yunshu Xitong Gongcheng Yu Xinxi/J. Transp. Syst. Eng. Inf. Technol. **14**(3), 207–213 (2014)
18. Tripathi, A.K., Sharma, K., Bala, M.: Dynamic frequency based parallel k-bat algorithm for massive data clustering (DFBPKBA). Int. J. Syst. Assur. Eng. Manag. **9**(1), 1–9 (2017)
19. Wang, H., Wang, Q., Wang, W.: Text mining for educational literature on big data with Hadoop. 166–170 (2018)
20. Agarwal, R., Singh, S., Vats, S.: Implementation of an improved algorithm for frequent itemset mining using Hadoop. International Conference on Computing, pp. 13–18 (2017)
21. Afrati, F., Stasinopoulos, N., Ullman, J.D., et al.: SharesSkew: an algorithm to handle skew for joins in MapReduce. Inf. Syst. **77**(2018), 129–150 (2018)
22. Ye, H., Meng, C., Wang, Y.: Frequent pattern mining algorithm based on MapReduce. J. Nanjing Univ. Sci. Technolo. **42**(1), 62–67 (2018)
23. Ma, K., Dong, F., Bo, Y.: Large-scale schema-free data deduplication approach with adaptive sliding window using MapReduce. Comput. J. **58**(11), 3187–3201 (2018)
24. Qureshi, N.M.F., Siddiqui, I.F., Unar, M.A., et al.: An aggregate MapReduce data block placement strategy for wireless IoT edge nodes in smart grid. Wirel. Pers. Commun. **106**(2), 2225–2236 (2018)
25. Takizawa, S., Matsuda, M., Maruyama, N., et al.: A scalable multi-granular data model for data parallel workflows. International Conference on High Performance Computing in Asia-pacific Region, pp. 1–10 (2018)
26. Zhou, Z., Zhao, X., Zhu, S.: K-harmonic means clustering algorithm using feature weighting for color image segmentation. Multimed. Tools Appl. **77**(12), 15139–15160 (2018)