Application of DBSCAN Algorithm in Data Sampling

Dingsheng Deng1 *

1 Sichuan University Nationalities, Kangding, Sichuan, 626001, China

*Corresponding author’s e-mail: dds0904@scun.edu.cn

Abstract: Data sampling is to sample hidden, previously unknown knowledge and rules that have potential value for decision-making from massive data. Cluster analysis is an important research topic in the field of data sampling. How to extract the information and knowledge that people care about, unknown, and help to analyze the decision-making process from massive data is a problem that people urgently need to solve. In this paper, after using the genetic algorithm-based method to obtain a better initial clustering center, the DPDGA algorithm divides the data set according to the obtained initial clustering center point. For each local data set obtained by division, the parameters MinPts of each local data set are calculated, and then each local data set is clustered using the DBSCAN algorithm, and finally the clustering results of each local data set are merged. Aiming at the shortcomings of the DBSCAN algorithm, this paper proposes a DBSCAN algorithm that uses particle swarm optimization to divide data and uses the MapReduce model to perform parallel calculations. The DPDPPO algorithm first uses the particle swarm optimization algorithm to obtain the optimal initial clustering center, and then partitions the data set according to the optimal initial clustering center. After partitioning, the DBSCAN's own k-dist graph is used to determine the ε and MinPts, finally merge each partition according to certain rules, and merge data points that may be mistaken for noise points. Research shows that when the amount of data reaches about 7M, the clustering method used in this article may take up to 1 hour. If the number of iterations is less, the time it takes will be less.

1. Introduction

With the rapid development of computer and communication technology, various types of data collected and stored are constantly increasing at an explosive rate. People's ability to use information technology has greatly improved, and tens of thousands of databases have been used for business management, government offices, scientific research and engineering development. At the same time, people are facing such a contradiction: on the one hand, people have huge amounts of data; on the other hand, people are worried about the lack of information and knowledge. How to extract the information and knowledge that people care about, unknown, and help to analyze the decision-making process from massive data is a problem that people urgently need to solve. Data sampling and knowledge discovery technology came into being.

Rahmah proposed a DBSCAN algorithm based on data sampling for the memory and I/O bottlenecks of the DBSCAN algorithm when dealing with large-scale databases. The algorithm uses data sampling to extend the DBSCAN algorithm, making it effective for cluster analysis of large-scale databases [1]. Among them, a fast cluster labeling method is adopted, so that the cluster calculation of the sampled data and the cluster labeling of the unsampled data can be quickly and synchronously performed, thereby greatly increasing the speed and speed of the entire clustering process. Effectiveness.
After partitioning the data set, this article uses the Hadoop cloud computing platform to perform parallel calculations on each partition. By designing a reasonable MapReduce programming model, the true parallelization of DBSCAN is achieved, which effectively solves the dependence of the DBSCAN algorithm on memory. It also improves the running time of the DBSCAN algorithm. Analyze the shortcomings of the DBSCAN algorithm. When the data distribution is uneven, the clustering quality is not good due to the use of global variables. When the data set is large, the demand for main memory is high. In this paper, the DPDGA algorithm is proposed. The algorithm divides the data set according to the obtained initial clustering center, calculates the parameters MinPts of each divided local data set separately, then uses the original DBSCAN algorithm to cluster, and finally merges the clustering results of each local data set.

2. Application Research of Dbscan Algorithm in Data Sampling

2.1. Data Types in Cluster Analysis

(1) Interval scale variables

The interval scale variable is a continuous variable with a roughly linear scale. Typical examples include weight and altitude, longitude and latitude, and atmospheric temperature [2].

The selected measurement unit will directly affect the results of cluster analysis. In general, the smaller the selected unit, the larger the possible value range of the variable, and thus the greater the impact on the clustering result. Therefore, in order to avoid the dependence of clustering results on unit selection, the data should be standardized [3-4]. After the standardization process, the dissimilarity between objects is calculated based on the distance. The most commonly used distance measurement method is Euclidean distance, which is defined as:

\[ d(i, j) = \sqrt{|x_{i1} - x_{j1}|^2 + |x_{i2} - x_{j2}|^2 + \ldots + |x_{ip} - x_{jp}|^2} \]  

(1)

Here \( i = (x_{i1}, x_{i2}, \ldots, x_{ip}) \) and \( j = (x_{j1}, x_{j2}, \ldots, x_{jp}) \) are two \( p \)-dimensional data objects. When using Euclidean distance, special attention should be paid to the selection of the measured values of the sample. It is a feature that effectively reflects the attributes of categories [5].

Two other well-known measures are Manhattan distance:

\[ d(i, j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \ldots + |x_{ip} - x_{jp}| \]  

(2)

Distance from Minkowski:

\[ d(i, j) = (|x_{i1} - x_{j1}|^q + |x_{i2} - x_{j2}|^q + \ldots + |x_{ip} - x_{jp}|^q)^{1/q} \]  

(3)

It can be seen from the Minkowski cluster: when \( q=1 \), it represents the Manhattan distance; when \( q=2 \), it represents the Euclidean distance.

(2) Binary variables

A binary variable has only two states: 0 or 1, 0 indicates that the variable is empty, and 1 indicates that the variable exists. For example, given a variable smoker describing the patient, 1 means that the patient smokes, and 0 means that the patient does not smoke [6]. If the binary variables have the same weight, this table reflects the possibility of the variable values of the two objects. In the table, \( q \) is the number of variables where both object \( i \) and object \( j \) have a value of 1, \( r \) is the number of variables where object \( i \) has a value of 1 and object \( j \) has a value of 0, and \( s \) is the value of object \( i \) with a value of 0 and object \( j \) has a value of 1. The number of variables, \( t \) is the number of variables whose objects \( i \) and \( j \) are 0 [7-8]. The total number of variables is \( p \), \( p = q + r + s + t \).

If the two states of a binary variable are of equal value and have the same weight, the binary variable is symmetrical. At this time, evaluating the difference between the two objects \( i \) and \( j \) is the most famous simple matching coefficient, which is defined as follows:

\[ d(i, j) = \frac{r+s}{t+\frac{r+s}{t}} \]  

(4)

If the output of two states of a binary variable is not equally important, then the binary variable is asymmetric. For example: a positive and negative result of a disease check. According to convention, we will output the more important output, usually the result with a low probability of encoding as 1, and the other result as 0. Given two asymmetric binary variables, two values of 1 are considered more
meaningful than two values of 0 [9]. At this time, the evaluation coefficient of the dissimilarity is the Jaccard coefficient:

\[ d(i, j) = \frac{r + s}{1 + r + s} \]  

(5)

2.2. Cluster Center Sampling Method Based on Particle Swarm

Particle swarm optimization can optimize various functions simply and effectively. It can also be said that the algorithm is somewhere between genetic algorithm (GA) and evolutionary programming [10].

For example, in a D-dimensional target search space, there are m particles in the space forming a whole population, each particle i has an initial position \( x_{id} \), and each position may be an optimal solution-through the target The function calculates the fitness value of the initial position to determine whether it is the optimal solution. Suppose the flight speed of the i-th particle is \( v_{id} \). Remember that the current optimal position of the particle is \( p_{id} \), and the current optimal position of the entire population is \( p_{gd} \), then each particle adjusts its position according to the following formula:

\[ v_{id} = w \cdot v_{id} + c_1 r_1 (p_{id} - x_{id}) + c_2 r_2 (p_{gd} - x_{id}) \]  

(6)

\[ x_{id} = x_{id} + v_{id} \]  

(7)

Where \( i = 1, 2, \ldots, m \), \( d = 1, 2, \ldots, D \). The acceleration constants \( c_1 \) and \( c_2 \) are non-negative numbers; \( r_1^k \) and \( r_2^k \) are random functions taking values between (0,1).

3. Experimental Research on Dbscan Algorithm in Data Sampling

3.1. Experimental Environment

This article makes a complete set of executable software packages for the DBSCAN algorithm. In order to compare DPDPSO and DBSCAN, we used the Java language to write the entire DPDPSO algorithm, and re-implemented the DBSCAN algorithm. Its stand-alone hardware environment is Intel(R) Pentium(R) CPU 1.87GHZ, Windows 7 system with 2G memory, JDK version adopts the latest JDK. At the same time, in order to perform parallel computing on the data of each partition, the Hadoop cloud computing platform deployed by the university is used.

3.2. Processing Uneven Distribution Data Set Comparison

In this paper, two sets of data are used for experiments: the first set is 100 two-dimensional data; the second set is 500 two-dimensional data, and both sets of data are randomly generated. The original DBSCAN algorithm, the DBSCAN algorithm (DPDGA) improved by genetic algorithm in this paper and the DPDPSO algorithm in this paper are used for experiments on the two sets of data. Among them, the crossover probability of genetic algorithm is 0.5. The probability of variation is 0.001. 3.3 merge the clustering results of each partition.

When we get the final clustering result, we also need to merge the clustering results of each partition, because partitioning may cause a cluster to be split into two different clusters in different partitions. It can be seen from the foregoing data partitioning principle that in order to prevent the object from being mis-segmented, the \( \varepsilon \) neighborhood is introduced in the partitioning algorithm. This is for processing when the object is just at the critical point during data partitioning, that is, when merging partitions Calculated by the given formula \( \varepsilon \)-The objects in the neighborhood meet certain rules, then the two clusters belonging to the objects belonging to this area are merged.

4. Experimental Research and Analysis of DBSCAN Algorithm in Data Sampling

4.1. Memory Performance Comparison

The content of the experiment is to compare the time required for DBSCAN and DPDPSO to process the same data scale. In the experiment, the DBSCAN algorithm needs to operate on the entire data set. When the amount of data gradually increases, the time required by the algorithm will increase exponentially and there will be insufficient memory. At runtime, the memory of the JVM is set to 1G, and the JVM used by the Hadoop platform is also set to 1G. The data comes from the user's GPS
latitude and longitude collected based on a certain product. During the experiment, the data is gradually increased, and the time spent in the experiment is observed. The experimental conditions are shown in Table 1 (τ_1 is the time required by the DBSCAN algorithm, and τ_2 is the DPDPSO on the Hadoop platform. It takes time, of which 1~3 is two-dimensional data, and 4~6 is three-dimensional data). As shown in Table 1.

| Experiments | Sample size/KB | Number of samples/10^5 | τ_1/s | τ_2/s |
|-------------|----------------|------------------------|-------|-------|
| 1           | 421.4          | 0.4                    | 11    | 13    |
| 2           | 872.4          | 0.8                    | 23    | 21    |
| 3           | 2436.0         | 2.4                    | 45    | 43    |
| 4           | 7184.2         | 4.8                    | 624   | 589   |
| 5           | 13583.6        | 8.6                    | 1824  | 1800  |
| 6           | 26864.8        | 17.2                   | 8292  | 8431  |

It can be seen from the above experimental results that the computational complexity of DBSCAN is quite high. When the data volume reaches about 7M, the clustering time may reach 1 hour. Of course, the clustering time is also related to the number of iterations. If the fewer the number of iterations, the less time will be spent. But in general, the time taken by DBSCAN is growing much faster than the growth rate of data, and the hardware resources consumed are becoming larger and larger, and even there will be insufficient memory. The experimental results of DPDPSO can be seen that when the data is small, the computing power is basically the same as DBSCAN, and sometimes it is even worse than the DBSCAN algorithm. This is because the DPDPSO algorithm needs to be partitioned before parallel processing can be performed. However, the same can be seen It is found that when the DBSCAN algorithm has insufficient memory, DPDPSO can still perform clustering operations very efficiently. It can be seen that DPDPSO solves the deficiencies of DBSCAN when processing large-scale data, thereby effectively reducing DBSCAN's dependence on memory.

4.2. Treatment of Unevenly Distributed Data Set Comparison

We took 500 two-dimensional data points for the experiment. The experimental data is shown in Figure 1, and the steps are the same as the first group of data. As the amount of data increases, different algorithms present different clustering results. For the convenience of observation, the data belonging to a certain category is filled with colors, and different categories are displayed with different colors. Data that does not belong to any category are noise points.
The original DBSCAN algorithm treated the relatively sparse points at the bottom right as noise, which obviously was not true. The DPDGA algorithm and the DPDPSE algorithm in this paper select different values of \( \epsilon \) and Minpts for each partition, so that data with relatively low density can be accurately clustered.

When the amount of data is small, the clustering quality of the DPDGA algorithm and the DPDPSE algorithm is not much different and both are significantly better than the original DBSCAN algorithm. However, as the amount of data increases, the DPDGA algorithm will also fall into the local optimal solution, which splits the data on the left that originally belonged to one class into two classes. It can be seen that particle swarm optimization has better global optimization ability than genetic algorithm. The DPDPSE algorithm can get better clustering results and is better at handling data sets with uneven data distribution.

5. Conclusions
This paper proposes an improved DBSCAN algorithm-DPDPSE algorithm in view of the above defects of DBSCAN. The algorithm partitions the data set, sets different initialization parameters for each partition, uses the Hadoop cloud computing platform for efficient calculation, and finally merges the clustering results of each partition according to certain rules. Experiments show that the DPDPSE algorithm is not only superior to the standard DBSCAN algorithm, but also superior to the DPDGA algorithm. The DBSCAN algorithm has a class that can find any shape, but the DBSCAN algorithm process continuously calculates the matrix between two objects. When the amount of data is large, it is easy to cause memory overflow. If the data set is not all loaded into memory, then it will also result in higher I/O consumption; when DBSCAN handles uneven data distribution, due to the human factors of the initial input parameters and the factors that cannot be modified after determination, the correct clustering results are often not obtained.

Acknowledgments
This paper was financially supported by the Key Project of Natural Science of Sichuan Provincial Education Department (No. 17ZA0295), 2017 Applied Demonstration Course Project of Sichuan University for Nationalities (No. sfkc201705) and Key Project of Natural Science of Sichuan University for Nationalities (No. XYZB19001ZA).

References
[1] Rahmah N, Sitanggang I S. Determination of Optimal Epsilon (Eps) Value on DBSCAN Algorithm to Clustering Data on Peatland Hotspots in Sumatra[J]. Iop Conference, 2016, 31(1):012012.
[2] Sukmasetya P, Sitanggang I S. Outlier Detection on Hotspots Data in Riau Province using DBSCAN Algorithm[J]. Iop Conference, 2016, 31(1):012035.
[3] Schubert E, Sander J, Ester M, et al. DBSCAN revisited, revisited: Why and how you should (still) use DBSCAN[J]. ACM Transactions on Database Systems (TODS), 2017, 42(3):19.
[4] Wang B, Zhang C, Song L, et al. Design and optimization of DBSCAN Algorithm based on CUDA[J]. Computer ence, 2015, 40(5):553-556.
[5] Nasiboglu R, Tezel B T, Nasibov E. Learning the stress function pattern of ordered weighted average aggregation using DBSCAN clustering[J]. International journal of entelligent systems, 2019, 34(3):477-492.
[6] Huang H, Liu F, Zha X, et al. Robust bad data detection method for microgrid using improved ELM and DBSCAN algorithm[J]. Journal of Energy Engineering, 2018, 144(3):04018026.1-04018026.11.
[7] Lee S. A Hybrid Framework using Fuzzy if-then rules for DBSCAN Algorithm[J]. International journal of computational intelligence research, 2018, 14(5):403-412.
[8] Zhang S K, Shi G Y, Liu Z J, et al. Data-driven based automatic maritime routing from massive AIS trajectories in the face of disparity[J]. Ocean Engineering, 2018, 155(MAY 1):240-250.
[9] Sun X, Zheng H, Li H, et al. Bad Data Identification for Leakage Reactance Parameters of Transformer Based on Improved DBSCAN Algorithm[J]. Automation of Electric Power Systems, 2017, 41(9):96-101.

[10] Gong L, Sato H, Yamamoto T, et al. Identification of activity stop locations in GPS trajectories by density-based clustering method combined with support vector machines[J]. Journal of Modern Transportation, 2015, 23(003):202-213.