DBSCAN-KNN-GA: a multi Density-Level Parameter-Free clustering algorithm

Bin Mu*, Meng Dai and Shijin Yuan

School of Software Engineering, Tongji University, 4800 Caoan Road, Shanghai, China

*E-mail: binmu99@163.com

Abstract. DBSCAN is a popular tool to analyse datasets which can effectively discover clusters with arbitrary shapes. However, it requires two input parameters which are difficult to be determined, according to the fact that the performance of clustering result depends heavily on user-specified parameters. In addition, it uses global parameters which are not appropriate to those multi-density datasets. Aiming at these problems, we propose a parameter-free algorithm to perform DBSCAN with different density-level parameters. We select some classical datasets and a TLC taxi trip record used for experiments to compared our proposed algorithm with the original DBSCAN to evaluate the performance of our improved DBSCAN. The results show that the proposed algorithm is capable for efficiently and effectively detecting clusters automatically with variable density-levels. Compared with original DBSCAN, the proposed algorithm can discover more noise points and its execution accuracy is higher.

1. Introduction

Clustering is a powerful unsupervised learning technique and one of the fundamental tasks in data mining which is used in applications such as machine learning, pattern recognition, image analysis, information retrieval, bioinformatics, data compression, and computer graphics. Clustering is a method of dividing data into different groups such that objects in the same group have the similar characteristics while are dissimilar to each other when they are not in different groups. There are many kinds of clustering algorithms which can be classified into partitioning, hierarchical, density, model based and grid-based methods [1]. K-means and k-medoids are famous partitioning clustering algorithms. The two algorithms are sensitive to k, the number of partitions. It is difficult to get the best value of k, and it is easier to get locally optimal result by using iteration. Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH) [2] are hierarchical clustering algorithm, which clusters dataset by CF tree and is suitable to process large dataset. Model-based clustering algorithms attempt to optimize the fit between the given data and some mathematical models, e.g. decision trees and neural networks [3]. The grid-based clustering approach using a multiresolution grid data structure, e.g. STratistical INformation Grid-based method (STING) [4] and CLustering In QUEst (CLIQUE) [5], processes data very fast and is typically independent of the number of data objects, yet dependent on only the number of cells in each dimension in the quantized space [6]. DBSCAN [7], one of density-based clustering algorithm, does not need the number of clusters like k-means and convex sample set like BIRCH and it can discover clusters of arbitrary shape and distinguishing noises. However, it needs two parameters Eps and minPts, and the speed of processing data is not effective enough.
The parameters Eps and minPts play the significant role in DBSCAN on which the clustering results heavily depend, so it is significant to determine the exact values of them. Many scholars are devoted to discovering a general approach to determine Eps and minPts. Martin Ester and Hans-Perter Kriegel [7] apply k-dist plot to determine the parameter Eps and minPts. The main idea is to gain k-th nearest neighbor matrix. Their experiments indicate that there is no difference when k is larger than 4, so they draw 4-dist graph and discover the “valley” in a graphical representation to get the threshold point. They suggest users estimate the percentage of noise and use the threshold point to calculate Eps [7]. Hou and Gao presented a parameter-free algorithm based on the DSets (Dominant sets) and DBSCAN algorithm. They use histogram equalization to the pairwise similarity matrix of input data and make DSets clustering results, expand the clusters from DSet with DBSCAN to generate the clusters of arbitrary shapes without any parameter input. Actually, they don’t need to set the value of Eps, but minPts is set by default [8]. Gaonkar and Sawant, at first, get the k-dist values of every point in the dataset and then, unlike other authors, they sort it in ascending order and find a sharp change between two smooth curves from graph to determine whether the slope can be Eps. They cluster the datasets with different densities [9]. Wang et al. introduce adaptive DBSCAN in accordance with data. Their idea is based on Gaonkar and they make an improvement on calculating minPts [3]. Instead of calculating the number of data points in the eps-neighborhood of every point in the dataset, they select points with the same density level, ignoring the points in denser region.

Similar to the above improvements, we are also dedicated to solving the problem of acquiring Eps and minPts. Our proposed algorithm DBSCAN-KNN-GA combining KNN algorithm, genetic algorithm and DBSCAN, can acquire Eps and minPts automatically without any human factors. KNN algorithm [10] is proposed by Cover is one of a simple but effective classification methods in data mining. The core idea of KNN algorithm is that provided that most of the k nearest samples in the feature space of cluster belongs to a certain category, the cluster also belongs to this category, so we can apply KNN to classify dataset as several small clusters to discover different densities in DBSCAN rather than the global density to discover more noises in the dataset. By drawing k-dist plot, we can gain the rough range of Eps may exist by observation or calculates slopes. To acquire the exact value of Eps, we can use genetic algorithm [11] which find the optimal solution by imitating the natural selection and genetic mechanism. Genetic algorithm has good global search ability and can search all solution space quickly without failing into fast descent trap of local optimal solution. We can set the fitness function to find the optimal solution of Eps by iteration. Thus, DBSCAN-KNN-GA can automatically obtain Eps and minPts according to datasets and then pass them to DBSCAN to get the clustering results. We conduct several experiments to validate our proposed algorithm. We select some public well-known datasets to verify the feasibility of our proposed algorithm, that is whether it can automatically acquire Eps and minPts according to the different datasets and effectively cluster data. Then we select taxi trip record datasets to compare our proposed algorithm with original DBSCAN and GA-DBSCAN [12]. On one hand, the dataset is large enough to compare the efficiency of the algorithm; one the other hand, it is an uneven density distribution dataset so that we are able to verify whether the proposed algorithm can acquire multiple set of Eps and minPts and compare the accuracy of clustering results.

The rest paper is organized as follows. Section 2 gives a detailed description of traditional DBSCAN. In Section 3, we present our proposed algorithm called DBSCAN-KNN-GA in detail. Section 4 shows experimental results of proposed algorithm to verify its superiority. Finally, Section 5 concludes the paper and some directions for future research.

2. DBSCAN
Density Based Spatial Clustering of Application with Noise is a density-based clustering method. It fits in processing spatial dataset on account of its capability of detecting clusters of arbitrary shape at different scales. The main idea is: For each class of data objects in the cluster, for a given value minPts, this object Eps field must be greater than or equal to this number [13]. That is, the neighbourhood within Eps has to contain at least the number of minPts points. A brief description of DBSCAN is
given below. Database is D, the radius of each point is Eps, and the minimum number of points in a field is minPts.

Definition 1. Eps-neighbourhood
For each point in the database, the neighborhood of a point P within a given radius Eps is called eps-neighbourhood of P, denoted as:

\[ N_{\text{eps}}(P) = \{ Q \in D | \text{dist}(P, Q) \leq \text{Eps} \} \]  \hspace{1cm} (1)

where \( \text{dist}(P, Q) \) is the distance between point P and point Q.

Definition 2. Core point
If P could be core point, its eps-neighbourhood should have contained at least minPts points.

\[ |N_{\text{eps}}(P)| \geq \text{minPts} \]  \hspace{1cm} (2)

Definition 3. Boundary point
Suppose P is a core point, and Q is in the eps-neighbourhood of P, but Q is not a core point. In this case, Q is a boundary point.

Definition 4. Noise
If P is neither a core point, nor a boundary point, P will be noise, or outlier point. Figure 1 show the concept map of core point, border point and noise.

![Figure 1. Core point, border point and noise.](image)

Definition 5. Directly density-reachable
If P is a neighbour of Q and Q is a core point, then the relationship from P to Q is called directly density-reachable.

Definition 6. Density-reachable
Given a chain \( P_1, P_2, \ldots, P_n \), where \( P = P_1, Q = P_n \), if any relationship from \( P_i \) to \( P_{i+1} \) is directly density-reachable, then the relationship from P to Q is called density-reachable. Figure 2 shows the concept map of directly density-reachable and density-reachable.

Definition 7. Density-connected
If P is density-reachable to Q and Q is density-reachable to P, then the relationship between P and Q is density-connected.

![Figure 2. Directly density-reachable (a) Density-reachable (b).](image)

The steps of DBSCAN is given below:
Step 1: Find an unvisited point P, mark P as visited.
Step 2: Get \( N_{\text{eps}}(P) \), if \( |N_{\text{eps}}(P)| < \text{minPts} \), mark P as noise and go to Step 1.
Step 3: Create a new collection C, put P into C. Traverse each point Q in \( N_{\text{eps}}(P) \), and remove Q from \( N_{\text{eps}}(P) \). If Q is not visited, mark Q as visited. Otherwise, go to Step 5.
Step 4: Get \( N_{\text{eps}}(Q) \), if \( |N_{\text{eps}}(P)| \geq \text{minPts} \), put \( N_{\text{eps}}(Q) \) into \( N_{\text{eps}}(P) \).
Step 5: If Q is not yet a member of any collection, put Q into C. If is not empty, go to Step 3. If exist an unvisited point P, go to Step 1.

The traditional methods are based on experiences to determine the values of Eps and minPts. Some improved methods are based on k-dist graph to find the valley in the graph sorting points in the descending order or the knees sorting points in the ascending order. The latter which can discover multiple Eps is better than the former because an important property of many datasets is that their intrinsic cluster structures are unable to be characterized by global density parameters [3]. Figure 3 shows that the cluster could not be detected by global density parameter. Under this circumstance may global density parameters are big enough to cluster several small groups into a large one. This paper use KNN to observe multi density-level of eps, utilize genetic algorithm to adjust parameter to acquire the accurate eps and minPts.

![Figure 3. Cluster with different level of density.](image)

3. DBSCAN-KNN-GA

DBSCAN-KNN-GA is aimed to determine the value of Eps and minPts to improve DBSCAN. The main strategy of DBSCAN-KNN-GA is to narrow the range of density radius by means of KNN (k-nearest neighbor), utilize fitness function to approach the value of density radius and terminate in some constraints. The concrete method is as follows: initialize the distance matrix D, where presents the distance between point i to point j. The matrix D is symmetric diagonally.

$$D = \begin{bmatrix}
0 & d_{12} & \cdots & d_{1n} \\
\vdots & \ddots & \ddots & \vdots \\
d_{1n} & d_{2n} & \cdots & 0
\end{bmatrix}$$

Then we will draw k-dist graph in ascending order. In k-dist graph, it only considers the distance of a point from its k-th nearest neighbor point, instead, we calculate the average of the distance from a point to all its k-nearest neighbor (KNN). Considering all of the KNN neighbors of a point and the average leads to a smooth curve with the removal of noise and it is easier to detect the threshold of the density levels. For density levels [a, b], we can transform D to D’ which contains only zero and one.

$$d'_y = \begin{cases}
1, a \leq d_y \leq b \\
0, \text{others}
\end{cases}$$

(3)

In matrix D’, we can discover the candidate core points. We can suppose all the points in D as a population, and each row in represent the gene of point in this population, and each value can be thought of as a gene fragment. We can calculate the frequency of each gene fragment, and donate it as its fitness.

$$f_y = \begin{cases}
\frac{d'_y}{\sum_{i=1}^{n} d'_y} = 1 \\
0, d'_y = 0
\end{cases}$$

(4)
Due to the particularity of genes that each gene segment is equivalent, we can simply consider the sum of each gene fragment’s fitness as each point’s fitness.

\[ f_i = \sum_{j=1}^{n} f_{ij} (i = 1, 2, \ldots, n) \]  

(5)

The fitness of whole population is the maximum of each point’s fitness.

\[ f = \max_{i=1}^{n} (f_i) \]  

(6)

Our strategy is to find core points and remove them from the population. The probability of the selected core point is \( P_i \).

\[ P_i = \frac{f_i}{\sum_{j=1}^{n} f_j} (i = 1, 2, \ldots, n) \]  

(7)

We will set a value to determine the max times of iteration \( K \). In every iteration, will calculate the maximum of the population fitness, marked as \( f_k \),

\[ f_k = \max_{i=1}^{n} (f_i) \]  

(8)

where \( i \) is the value of times of iteration. Record the max population fitness \( f \) in all iterations. If \( f_k - f > \beta \), it means the iteration is moving on, \( f = f_k \). Otherwise, it means we have found the optimal core point. Table 1 The procedure of DBSCAN-KNN-GA shows the procedure of DBSCAN-KNN-GA.

| Table 1. | The procedure of DBSCAN-KNN-GA. |
|----------|---------------------------------|
| **Step 1:** Draw KNN graph and determine the range of Eps may exist. |  |
| **Input:** raw dataset | **Output:** range of Eps \((a, b)\)  |
| **Process:** |  |
| (1) Read raw dataset and save as point matrix \( D \), remove the repeated points. |  |
| (2) Calculate distance matrix \( Dist \) for every two different point. |  |
| (3) Sort distances for every point to the other points in \( Dist \) as new matrix \( SortedDist \). |  |
| (4) Normalize the matrix \( SortedDist \), discover k-nearest distances for every point and draw KNN graph. |  |
| (5) Find valley or knee from KNN graph according to slope between adjacent points in the graph. For each valley or knee, calculated corresponding range \((a, b)\) in which Eps exists. |  |
| **Step 2:** Determine two parameter Eps and minPts in DBSCAN. |  |
| **Input:** \( Dist \), Iteration \( K \), threshold \( \beta \), range of Eps \((a, b)\) | **Output:** Eps, minPts  |
| **Process:** |  |
| (1) Read distance matrix \( Dist \) and convert it to special matrix which only contains zero or one according to range of Eps \((a, b)\). Set the iteration number \( k \) as 0. |  |
| (2) Calculate fitness for every point as matrix \( Fitness \) and then acquire probability matrix \( Prob \) which represents the probability of each point to become candidate core point. Find the point with the highest probability and remove it from the group. |  |
| (3) Acquire the maximum value from \( Fitness \) as the fitness \( f_k \) of whole group. If \( f_k - f > \beta \land k < K \), the value of \( k \) increments and return to process 2. |  |
(4) Determine the threshold $Eps$ by calculating average of $d_{ij}$ of candidate core points. $MinPts$ is assigned by average of $Eps$-neighborhood of every point.

Step 3: DBSCAN algorithm
Input: $Dist$, a list of $(minPts, Eps)$
Output: (point, cluster, visited, isCorePoint, isNoise)
Process:
1) Select a point $p$ from raw $Dist$ to start its neighborhood searching and mark it visited. If $p$ is not a core point, mark it as noise point, else mark it as core point.
2) If $p$ a core point, then create a cluster $c$, adding all of neighbors of $p$ in $c$ for recursive calls.
3) Repeated process 2 and 3 until all the point is marked visited.
4) Select other $Eps$ and $minPts$ while $Eps$ is bigger than previous. Repeat process 1, 2 and 3 for those point marked as noise.

4. Experiments
In this section, we verify our proposed algorithm and evaluate the performance comparing it with DBSCAN. We perform the experiments in two steps: the first step is to utilize several classical datasets to support and verify DBSCAN-KNN-GA algorithm, the second step is to apply it to taxi trip dataset to analyse and evaluate the clustering performance.

In first step, we choose three datasets in this paper, one is classical iris dataset, one is from Compound and Aggregation of Computing School of the East Finland University [14], the other is the experimental data of Chameleon algorithm to verify the feasibility of DBSCAN-KNN-GA. To estimate the range of the $Eps$ value change, the $k$-dist graphs were constructed for the studied data. The $k$-dist graphs of Iris and Synthetic are shown in figure 4.

![Figure 4. K-dist graph of Iris, Aggregation and t5.8k.](http://glaros.dtc.umn.edu/gkhome/cluto/cluto/download)

Table 2. DBSCAN-KNN-GA on different inputs on Iris, Aggregation and t5.8k.

| Eps  | $MinPts$ | Total Points | Cluster Points | Noise Points | No. of Clusters |
|------|----------|--------------|----------------|--------------|----------------|
| 0.25 | 4        | 150          | 145            | 5            | 3              |
| 0.085| 3        | 788          | 788            | 0            | 5              |
| 3.6  | 4        | 8000         | 7153           | 847          | 6              |

In short to find all possible $Eps$ values we have to calculate the slopes at regular interval and then find the difference between slopes values at the same regular interval. By setting certain threshold value we can get different $Eps$ values automatically based on this threshold value while discarding those with higher thresholds [3]. In figure 4, some knees could be mistaken for as their sharp slope. for points of the same density level, the range of variation will not be huge while a sharp change is expected to see between two density levels. Thus, there will be several smooth curves connected by greatly variation ones [6]. After obtaining the ranges of $Eps$, we can acquire $Eps$ and $minPts$ using fitness function shown in table 2 DBSCAN-KNN-GA on different inputs on Iris, Aggregation and t5.8k. Figure 5 shows the results of DBSCAN-KNN-GA algorithm.
In second experiment, we adopt TLC trip record data as our data source. We select six datasets with increasing number of data points, which are 4162, 8481, 13761, 25667, 41682, 86473, to carry out the clustering simulation experiments. We perform several experiments to observe the relationship between the accuracy and the parameter. The relationship between the accuracy and the parameter is shown in figure 6.

Table 3. Parameter adaption value.

| Parameter | β       | Eps | minPts |
|-----------|---------|-----|--------|
| 4162      | 0.3     | 0.266 | 4      |
|           | 0.3     | 0.246 | 4      |
| 8481      | 0.3     | 0.266 | 5      |
|           | 0.3     | 0.187 | 5      |
| 13761     | 0.35    | 0.167 | 4      |
|           | 0.35    | 0.274 | 4      |
|           | 0.3     | 0.143 | 6      |
| 25667     | 0.3     | 0.185 | 4      |
|           | 0.35    | 0.347 | 5      |
| 41682     | 0.3     | 0.175 | 5      |
|           | 0.35    | 0.315 | 5      |
|           | 0.3     | 0.148 | 4      |
| 86473     | 0.35    | 0.266 | 6      |
|           | 0.3     | 0.382 | 4      |

Table 4. Accuracy of the algorithm execution (%).

| Data       | Accuracy  |
|------------|-----------|
|            | 4162      | 8481      | 13761     | 25667     | 41682      | 86473      |
| DBSCAN     | 95.32     | 92.16     | 89.62     | 86.45     | 83.16      | 81.43      |
| GA-DBSCAN  | 97.63     | 95.45     | 93.31     | 90.41     | 88.42      | 87.43      |
| DBSCAN-KNN-GA | 98.52     | 96.47     | 95.46     | 93.94     | 91.86      | 90.78      |

Footnote: 1 https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page
Figure 7. Contrast map with percentage of noise points.

We select DBSCAN algorithm and GA-DBSCAN as comparison. These clustering methods are experimented in the selected six datasets. The values of β, Eps and minPts of DBSCAN-KNN-GAs in experiments are recorded in table 3. We can observe that datasets are clustered with different densities. The accuracy rate and percentage of noise points are shown in table 4 and figure 7. We can see that the accuracy of the K-GA-DBSCAN algorithm execution is obviously higher than the other algorithms. The percentage of noise points is higher than the other algorithms because the minimum of local Eps is usually smaller than the global Eps, so many points may become core point or border point for those DBSCAN algorithm using global Eps, but are recognized as noise point in our proposed algorithm. In this way, we can distinguish the areas with concentrated density from those with sparse density. The execution time of each method on each dataset in the above experiments is recorded and their average execution time is compared. We can find that DBSCAN-KNN-GA saves manual calculating time and eliminate the interference of human factors and it can be further optimized through parallel and distributed computing.

5. Conclusion
This paper focuses on how to automatically determine multiple set of parameters Eps and minPts. The original DBSCAN requires Eps and minPts that are difficult to be determined and it simply uses the global Eps and minPts so that the clustering result of multi-density database is inaccurate. For these problems, we proposed a new algorithm using KNN algorithm and genetic algorithm. Based on k nearest neighbor matrix, we draw k-dist plot and reduce the value range of Eps, then with the iteration of genetic algorithm, we can select the optimal population and determine accurate Eps and minPts and finally pass the parameters to DBSCAN algorithm. We select classic datasets and real taxi trip record dataset for experiments, we compare our proposed algorithm with original DBSCAN algorithm and GA-DBSCAN in terms of clustering result accuracy, percentage of noises points and algorithm running speed. The experiments result shows that our proposed algorithm acquires more accurate parameters automatically has better clustering effect especially for those clusters of very different densities.

The proposed algorithm is lack of efficiency when encountering large datasets as it will cost a lot to establish distance matrix and perform DBSCAN algorithm. In the future research, we are looking forward to using paralyzed calculating or distribute calculating to reduce the algorithm execution time and improve processing efficiency.

References
[1] Tan, Pang-Ning, Michael Steinbach, Vipin Kumar. Introduction to Data Mining. (2005)
[2] T. Zhang, R. Ramakrishnan, M. Livny. BIRCH: An Efficient Data Clustering Method for Very Large Databases. ACM SIGMOD Record, 25, 2 (1996)
[3] Gaonkar, M. Naik, K. Sawant. AutoEpsDBSCAN: DBSCAN with Eps Automatic for Large Dataset. ICACTE, 2(2), 11-16 (2013).
[4] W. Wang, J. Yang, R. Muntz. STING: A Statistical Information Grid Approach to Spatial Data Mining. VLDB, 41 (1997)

[5] A. Rakesh, J. Gehrke, D. Gunopulos, P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. ACM, 27 (1998)

[6] S. Suman, P. Rani. A Survey on STING and CLIQUE Grid Based Clustering Methods. IJARCS, 8(5) (2017)

[7] E. Martin, H. Kriegel, J. Sander, X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. KDD, 96(34), 226-231 (1996)

[8] H. Jian, H. Gao, X. Li. DSets-DBSCAN: a parameter-free clustering algorithm. IEEE T IMAGE PROCESS, 25(7), 3182-93 (2016)

[9] W. Wang, Y. Wu, C. Tang, M. Hor. Adaptive density-based spatial clustering of applications with noise (DBSCAN) according to data. ICMLC, 1, 445-451 (2015)

[10] G. G. Enas, S. C. Choi. Choice of the smoothing parameter and efficiency of k-nearest neighbor classification. Statistical Methods of Discrimination and Classification. Statistical Methods of Discrimination and Classification, 235-244 (1986)

[11] D. Jong, K. Alan. Analysis of the behavior of a class of genetic adaptive systems (1975)

[12] X. Hu, L. Liu, D. Yang, M. Li. A MapReduce-based improvement algorithm for DBSCAN. J Algorithm Comput Tech, 12(1), 53-61 (2018)

[13] Y. He, H. Tan, W. Luo, H. Mao, D. Ma, S. Feng, J. Fan. Mr-dbscan: an efficient parallel density-based clustering algorithm using mapreduce. ICPADS, 473-480 (2011)

[14] A. Gionis, H. Mannila, P. Tsaparas. Clustering aggregation. TKDD, 1(1), 4 (2007)