An Enhanced Density Clustering Algorithm for Datasets with Complex Structures

Jieming Yang, Qilong Wu, Zhaoyang Qu, and Zhiying Liu

***Abstract*—There are several limitations of DBSCAN: 1) parameters have to be set; 2) the time consumption is intolerable in expansion; 3) the algorithm is sensitive to the density of starting points; 4) it is difficult to identify the adjacent clusters with different densities. In this paper, an enhanced and efficient density clustering algorithm for datasets with complex structures is proposed, named GISN-DBSCAN. Firstly, we propose an extended range query algorithm based on fixed-grids to reduce the time cost of searching the nearest neighborhood. Then the nearest neighbors (NN_k) and reverse nearest neighbors (RNN_k) are used to establish the *k*-influence space neighborhood of each point. Finally, a computational method of *k*-outlierness function is presented to distinguish the border points and noise points accurately. Experimental results demonstrate that GISN-DBSCAN can address the drawbacks of DBSCAN algorithm and identify the border points and noise points effectively. Moreover, the efficiency of GISN-DBSCAN has been greatly improved under the premise of guaranteeing the clustering quality.**

***Index Terms*—density-based clustering, extended range query, *k*-influence space neighborhood, border points detection**

I. Introduction

CLUSTERING is a fundamental problem and used in a variety of areas of computer science and related fields for data analysis [1], [2], such as pattern recognition, artificial intelligence, image segmentation, text analysis [3], [4], etc. The clustering algorithms can be roughly classified into partitional algorithms [5], hierarchical algorithms [6], density-based algorithms [7], grid-based algorithms [8] and model-based algorithms [9].

DBSCAN (Density-Based Spatial Clustering of Application with Noise) [10] is the clustering algorithm which implements the density-based strategy. The clusters with arbitrary shape and different sizes can be found by DBSCAN, and the noise points can be identified effectively. It does not require the users to specify the number of clusters. But there are following drawbacks: 1) parameters, *Eps* and *Minpts* in which *Eps* represents the maximum radius of a neighborhood and *Minpts* means the minimum number of data points contained in such a neighborhood, are difficult to be set without any prior knowledge. 2) The complexity of searching the nearest neighborhood of each point is high when dealing with the large dataset. 3) It is sensitive to the density of the starting point. Different starting points will result in various consequences. And 4) it is difficult to identify the adjacent clusters of different densities. A single dense cluster may consist of two adjacent areas with significantly different densities (both higher than a threshold) [11].

To overcome the drawbacks of DBSCAN mentioned above and identify the border points and noise points effectively, an enhanced density clustering algorithm for datasets with complex structures is proposed in this paper, named GISN-DBSCAN. Moreover, an extended range query algorithm based on fixed-grids is firstly proposed to reduce the time cost of searching the nearest neighborhood of each point. Then the nearest neighbors (NN_k) and reverse nearest neighbors (RNN_k) are used to establish the *k*-influence space neighborhood of each point, which can reduce the amount of parameters effectively and identify the adjacent clusters of different densities. At the same time, it is not sensitive to the density of the starting point. Finally, a computational method of *k*-outlierness function (OLF_k) is proposed to distinguish the border points and noise points effectively.

The rest of this paper is organized as follows. Related work is discussed in Section II, some definitions and details of GISN-DBSCAN are described in Section III. Several experimental studies are analyzed in Section IV and we summarize our paper in Section V.

II. Related Work

Clusters, obtained by DBSCAN algorithm, are the dense areas which are separated by the sparse clusters or blank areas. For each point *p* in dataset, the number of points contained in the ε-neighborhood needs to be compared with *Minpts*. And the points will be clustered into a cluster with density connected characteristic.

To improve the efficiency, a clustering algorithm based on density and grid for large data bases was proposed in [12], named GCHL. However, it is sensitive to the order of inputting data and two data-dependent parameters are difficult to be set. A concept of local outlier factor, named LOF, and outlier detection algorithm were proposed in [13]. The proposed algorithm could provide better results than DBSCAN algorithm. A computational method of local outlier factor (INFLO) was proposed in [14]. By calculating the

Manuscript received July 15, 2016; revised January 06, 2017. This work was supported by National Natural Science Foundation of China under Grant no. 51437003 and Project Development Plan of Science and Technology of Jilin Province under Grant No. 20160623004TC.

Jieming Yang is with College of Information Engineering, Northeast Electric Power University, Jilin, Jilin, China (e-mail: yjmlzy@sina.com).

Qilong Wu is with College of Information Engineering, Northeast Electric Power University, Jilin, Jilin, China (neduqlw@foxmail.com).

Zhaoyang Qu is with College of Information Engineering, Northeast Electric Power University, Jilin, Jilin, China (e-mail: qzywww@mail.nedu.edu.cn).

Zhiying Liu is with College of Information Engineering, Northeast Electric Power University, Jilin, Jilin, China (e-mail: 312545902@qq.com).

nearest neighbors (NN_k) and reverse nearest neighbors (RNN_k), INFLO could capture the degree of the outlierness of each point accurately. The NN_k and RNN_k were successfully applied to DBSCAN algorithm in [15], named IS-DBSCAN. However, the proposed algorithm requires large time overhead, and the complexity is high.

III. Algorithm Description

A. Definition

Let $D = \{p_1, \ldots, p_n\}$ denote a set of n points. Each point $p_i \in D$ is a d-dimensional vector $<p_i(1), p_i(2), \ldots, p_i(d)>$ where $p_i(\eta)$ represents the η-th dimension of the point and n is the total number of points in D.

Definition 1: (grid cell) Our algorithm is parameterized by ε. According to the ε, each dimension of the data space is divided into n' spaces, then the whole data space will be partitioned into grids and the length of each grid is ε.

Definition 2: (t-extended rectangular domain) We denote the grid cell as *Cell*, and the *Cell* are referred to as the center and extended t cells along the positive direction and the opposite direction of each dimension. So the area, covered by the grid set which is formed by the *Cell* and extended cells, is named the t-extended rectangular domain.

Example 1: Considered some of these 2-dimension data points in Fig 1. When $t = 0$, the t-extended rectangular domain is covered by {*Cell*}. When $t = 1$, the t-extended rectangular domain is covered by {*Cell*, $Cell_1$, $Cell_2$, $Cell_3$, $Cell_4$, $Cell_5$, $Cell_6$, $Cell_7$, $Cell_8$}.

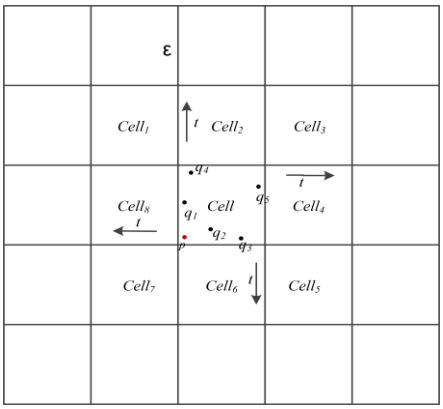

Fig 1. t-extended rectangular domain

Definition 3: ($k_{dist}(p)$) The k-distance of p where $p \in D$, denoted as $k_{dist}(p)$, is the distance $dist(p,q)$ between p and a point q in D, such that: 1) at least for k points $q' \in D$, it holds that $dist(p, q') \le dist(p, q)$ and 2) at most for (k-1) points $q' \in D$, it holds that $dist(p, q') < dist(p, q)$.

Definition 4: ($den_k(p)$) The density of p, denoted as $den_k(p)$, is the inverse of k-distance of p, i.e. $den_k(p) = 1/k_{dist}(p)$.

Definition 5: ($NN_k(p)$) The k-nearest neighborhood of a point p, denoted as $NN_k(p)$, is a set of points $o \in D$ with $dist(p, o) \le k_{dist}(p)$: $NN_k(p) = \{o \in D \setminus \{p\} | dist(p, o) \le k_{dist}(p)\}$, and $|NN_k(p)|$ is denoted as the number of points in $NN_k(p)$.

Definition 6: ($RNN_k(p)$) The reverse k-nearest neighborhood of a point p, denoted as $RNN_k(p)$, is defined as $RNN_k(p) = \{q \in D \mid p \in NN_k(q)\}$, and $|RNN_k(p)|$ is denoted as the number of points in $RNN_k(p)$.

Definition 7: ($ISN_k(p)$) The k-influence space neighborhood of a point p is defined as $ISN_k(p) = \{q \mid q \in NN_k(p) \cap RNN_k(p)\}$, and $|ISN_k(p)|$ is denoted as the number of points in $ISN_k(p)$.

Example 2: Considered the 2-dimension data points in Fig 2. Suppose that we would identify the ISN_k of each point with $k = 3$. Fig 2 shows that the k-distance of p is the distance $dist(p, q_3)$ between p and q_3. The k-nearest neighborhood of p is $NN_k(p) = \{q_1, q_2, q_3\}$. Similarly, $NN_k(q_1) = \{p, q_2, q_4\}$, $NN_k(q_2) = \{p, q_1, q_3\}$, $NN_k(q_3) = \{p, q_2, q_5\}$, $NN_k(q_4) = \{p, q_1, q_2, q_5\}$, $NN_k(q_5) = \{q_2, q_3, q_4\}$. After that, the reverse k-nearest neighborhood of each point can be obtained. $RNN_k(p) = \{q_1, q_2, q_3, q_4\}$, $RNN_k(q_1) = \{p, q_2, q_4\}$, $RNN_k(q_2) = \{p, q_1, q_3, q_4, q_5\}$, $RNN_k(q_3) = \{p, q_2, q_5\}$, $RNN_k(q_4) = \{q_1, q_5\}$ and $RNN_k(q_5) = \{q_3, q_4\}$. Finally, k-influence space neighborhood of each point will be obtained by combining NN_k and RNN_k together. Then, $ISN_k(p) = \{q_1, q_2, q_3\}$, $ISN_k(q_1) = \{p, q_2, q_4\}$, $ISN_k(q_2) = \{p, q_1, q_3\}$, $ISN_k(q_3) = \{p, q_2, q_5\}$, $ISN_k(q_4) = \{q_1, q_5\}$, $ISN_k(q_5) = \{q_3, q_4\}$.

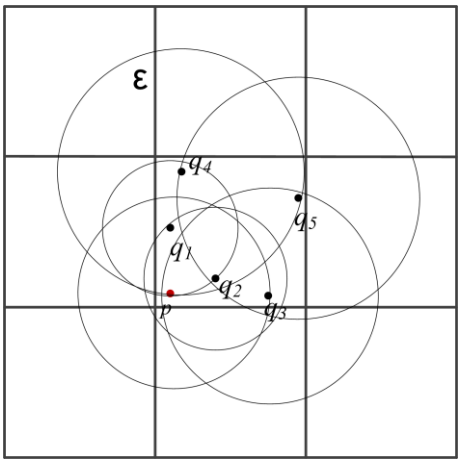

Fig 2. Identifying the ISN_k in t-extended rectangular domain

Definition 8: (core point) If the number of points in $ISN_k(p)$ is more than 2k/3 (i.e. $|ISN_k(p)| > 2k/3$) [15], then p is a core point.

Definition 9: (directly density reachable) Point q is directly density reachable from p if $q \in ISN_k(p)$ and $|ISN_k(p)| > 2k/3$.

Definition 10: (density reachable) If there is a chain of points $p_1, p_2, \ldots, p_n$, where $p_1 = q$, $p_n = p$, $p_i \in D$, $1 \le i \le n$, such that p_{i+1} is directly reachable from p_i and p_i is a core point (i.e. $|ISN_k(p_i)| > 2k/3$), then q is density reachable from p.

Definition 11: (density connected) If there is a point $o \in D$ such that p and q are density reachable from o, then a point p is density connected to a point q.

Definition 12: (border point, noise point) If a point p is not a core point and p is directly density reachable from a core point, then p is a border point. If p is neither a core point nor a border point, then p is a noise point.

Definition 13: (OLF_k) The k-outlierness function of a point p, denoted as $OLF_k(p)$, is defined as:

$$OLF_k(p) = \frac{\sum_{o \in ISN_k(p)} den(o)}{|ISN_k(p)| den(p) |RNN_k(p)|} \quad (1)$$

$OLF_k(p)$ is the ratio of the average density of points in $ISN_k(p)$ to $den_k(p)$ and $|RNN_k(p)|$. When a point p is the noise point, there is no points or only have noisy points in $RNN_k(p)$ and $ISN_k(p)$. Accordingly, when the number of points in the

k-influence space neighborhood of a point $p_i \in D$ is 0 (i.e. $|ISN_k(p_i)| = 0$), then p_i is the noise point. When the number of points in the *k*-influence space neighborhood of p_i is more than 0, but less than 2k/3 (i.e. $0 < |ISN_k(p_i)| < 2k/3$), then we calculate the $OLF_k(p_i)$. If $OLF_k(p_i) > 1$, then p_i is the noise point, otherwise, p_i is the border point.

B. Extended Range Query Algorithm based on Fixed-grids

In this section, the sequential version of the extended range query algorithm based on fixed-grids is introduced. If we calculate the density of each point in *D*, we need to execute range query and judgement of each point and calculate the distance over and over again. Therefore, the time complexity is $O(n^2)$. As the amount of data increasing, the time cost of DBSCAN algorithm will be enormous [16]. Accordingly, an extended range query algorithm based on fixed-grids is proposed in this section.

The *k*-distance of *p* is always constant in the whole data space, and the *k*-nearest neighborhood of *p* is composed of the nearest *k* points from *p*. Therefore, when we calculate $k'_{dist}(p)$ and $den'_k(p)$ in the areas which is composed of different grids, the minimum value of $k'_{dist}(p)$ is the *k*-distance of *p* (i.e. $k_{dist}(p) = k'_{dist}(p)_{\min}$), and the maximum of $den'_k(p)$ is the density of *p* (i.e. $den_k(p) = den'_k(p)_{\max}$). If the number of points in *t*-extended rectangular domain of *p* is less than *k*+1, we set $den_k(p) = 0$. The Algorithm 1 (see Table I) explains how the extended range query algorithm works.

Table I
PROCEDURE OF ALGORITHM 1

Algorithm1: Extended Range Query Algorithm(ε, *D*)

1: mark all points $p_i \in D$ as "*UNCLASSIFIED*" and $t = 1$
2: **for all** $p_i \in D$ **do**
3: **if** p_i is marked as "*UNCLASSIFIED*" **then**
4: **while** $den^{(j-1)}{}_k(p_i) < den^{(j)}{}_k(p_i)$ or $den^{(j-1)}{}_k(p_i) = den^{(j)}{}_k(p_i) = 0$ **do**
5: obtain the *t*-extended rectangular domain for the *t*-th expansion of (*t*-1)-extended rectangular domain
6: calculate the $den^{(j)}{}_k(p_i)$ of p_i in *t*-extended rectangular domain
7: *t*++
8: **end while**
9: calculate the $NN_k(p_i)$ and mark p_i as "*CLASSIFIED*"
10: **end if**
11: **end for**
12: **for all** $p_m \in D$ **do**
13: calculate the $ISN_k(p_m)$
14: **end for**
15: Output ISN_k

C. Procedure of GISN-DBSCAN Algorithm

In this paper, GISN-DBSCAN algorithm only need a parameter *k* to calculate the NN_k and RNN_k of each point. Compared with DBSCAN algorithm, it not only reduces the number of parameters, but also ensures the sensitivity to the change of local density with regard to each point. So GISN-DBSCAN algorithm can identify the adjacent clusters with different densities. In essence, the reverse nearest neighbor $RNN_k(p)$ is composed of all the points whose *k*-nearest neighbors include *p*. So NN_k and RNN_k are symmetric neighborhood, and ISN_k, obtained by calculation, is also symmetric. Therefore, GISN-DBSCAN algorithm is not sensitive to the densities of starting points.

The procedure of GISN-DBSCAN algorithm can be divided into three steps. The first step is to scan the whole datasets, mark all points as "*UNCLASSIFIED*" and calculate ISN_k of each point. If there is no point in $ISN_k(p_i)$ (i.e. $|ISN_k(p_i)| = 0$), the point $p_i \in D$ will be marked as "*NOISE*" and "*CLASSIFIED*". The second step is to identify whether the point p_i is the core point successfully. If $|ISN_k(p_i)| > 2k/3$, then p_i is the core point and marked as "*CLASSIFIED*" and current *ClusterID*. All the points in $ISN_k(p_i)$ with "*UNCLASSIFIED*" will be inserted into *SeedList*. If $|ISN_k(p_i)| \leq 2k/3$, then p_i is border point or noise point. Then we calculate the $OLF_k(p_i)$, if $OLF_k(p_i) > 1$, then p_i is noise point, otherwise, p_i is border point and marked as "*CLASSIFIED*" and current *ClusterID*. If p_i is border point, all the points in $ISN_k(p_i)$ which is marked as "*UNCLASSIFIED*" will be inserted into *SeedList*. The last step of GISN-DBSCAN is the expansion of clusters on the basis of the points in *SeedList*. Taking a point p_j in *SeedList*, and p_j will be processed by the method of second step. A cluster is successfully completed until there is no points in *SeedList*. The second step and third step are repeated until all the points are marked as "*CLASSIFIED*". The Algorithm 2 (see Table II) and Algorithm 3 (see Table III) explain how the GISN-DBSCAN algorithm works.

Table II
PROCEDURE OF ALGORITHM 2

Algorithm2: GISN-DBSCAN(*k*, *D*)

1: *ClusterID*=1, *SeedList* $\leftarrow \varphi$
2: *Extended Range Query Algorithm*(ε, *D*)
3: mark all points $p_i \in$ D as "*UNCLASSIFIED*"
4: **for all** $p_i \in D$ **do**
5: **while** p_i is marked as "*UNCLASSIFIED*" **do**
6: **if** $|ISN_k(p_i)| > 2k/3$ **then**
7: mark p_i as "*CLASSIFIED*", p_i is labeled as *ClusterID*
8: **for** each point $p_j \in ISN_k(p_i)$ **do**
9: **if** p_j is marked as "*UNCLASSIFIED*" and $p_j \notin$ *SeedList* **then**
10: add p_j into *SeedList*
11: **end if**
12: **end for**
13: **else**
14: *Noise Detecting*(p_i, *ClusterID*, *SeedList*)
15: **while** *SeedList* $\neq \varphi$ **do**
16: select the head point p_1
17: $p_i \leftarrow p_1$
18: remove p_1 from *SeedList*
19: **end while**
20: **end while**
21: *ClusterID*++
22: **end for**
23:Output Clusters

Table III
PROCEDURE OF ALGORITHM 3

Algorithm3: Noise Detecting(p_i, *ClusterID*, *SeedList*)

1: **if** $|ISN_k(p_i)| = 0$ **then**
2: mark p_i as "*NOISE*" and "*CLASSIFIED*"
3: **else**
4: calculate the $OLF_k(p_i)$
5: **if** $OLF_k(p_i) > 1$ **then**
6: mark p_i as "*NOISE*" and "*CLASSIFIED*"
7: **else**
8: mark p_i as "*BORDER*", "*CLASSIFIED*" and *ClusterID*
9: **for** each point $p_j \in ISN_k(p_i)$ **do**
10: **if** p_j is marked as "*UNCLASSIFIED*" and $p_j \notin$ *SeedList* **then**
11: add p_j into *SeedList*
12: **end if**
13: **end for**
14: **return** p_i, *ClusterID*, *SeedList*

D. Time Complexity

Firstly, the data spatial is divided into grids. The maximum and minimum of each dimension in the space need to be calculated, so the time complexity is $O(n*d)$, where n is the total number of points in D and d is the amount of dimensions. Secondly, all the points are mapped into the grid cells, so the time complexity is $O(n)$. Thirdly, in terms of existing spatial indexing structure R tree, ISN_k will be calculated in the extended rectangular domain of each point, so the time complexity is $O(n*\log m)$, where m is the total number of points in the extended rectangular domain of each point, and m is much less than n. Finally, each point in *SeedList* will be clustered and the time complexity is $O(1)$ which has nothing to do with n. Thus, the time complexity of GISN-DBSCAN is $O(n*d + n*\log m)$.

IV. Experimental Setup

A. Datasets

Six datasets are used to evaluate the performance of GISN-DBSCAN algorithm. Synthetic Dataset (in Fig 3) consists of 2000 points in 2 dimensions. Aggregation Dataset [17] (in Fig 4) consists of 788 points in 2 dimensions. Aggregation* Dataset (in Fig 5) is generated by Aggregation Dataset which adds in random noise points. Iris Dataset, consists of 150 points in 4 dimensions. Harberman's Survival Dataset consists of 306 points in 4 dimensions. Letter Recognition Dataset consists of 20,000 points in 16 dimensions. The last three datasets are publicly available from the UC Irvine Machine Learning repository [18].

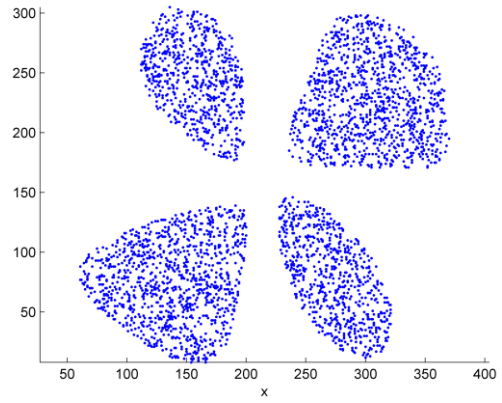

Fig 3. Synthetic Dataset

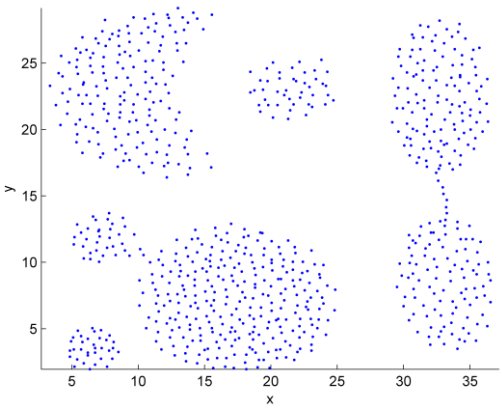

Fig 4. Aggregation Dataset

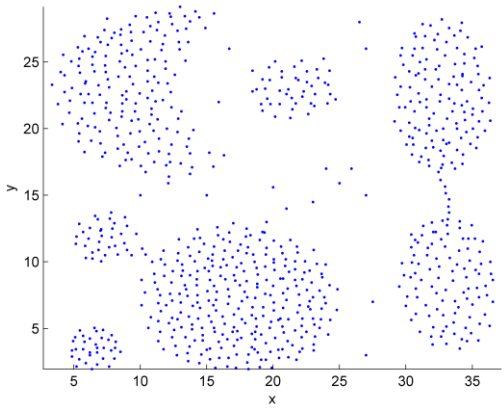

Fig 5. Aggregation* Dataset

The real cluster label of each point in the Synthetic Dataset and the real datasets of Iris, Harberman's Survival, Letter Recognition is known, then we use the correct rate to evaluate the performance of DBSCAN, IS-DBSCAN [15] and GISN-DBSCAN in the following experiments. The correct rate can be defined as: Correct rate = $(M / n) * 100\%$, where n is the number of points in dataset D, and M is the number of points which are correctly classified.

B. Performance Comparison

Synthetic Dataset is used in DBSCAN, IS-DBSCAN and GISN-DBSCAN in this subsection. The Synthetic Dataset has been increased by duplicating its original points, and the number of points is respectively 2,000, 6,000, 10,000, 14,000, 18,000, 22,000. To prevent from generating exactly the same points, we add a small random noise whose range is between -1 and 1 with regard to each coordinate of the points. Fig 6 and Fig 7 record the execution efficiency and correct rates of these algorithms respectively. Table IV indicates the correct rates of these algorithms in Iris, Harberman's Survival and Letter Recognition. The result of each experiment is the average obtained from these algorithms by running 20 cycles.

From Fig 6, the execution time of GISN-DBSCAN is significantly less than these of DBSCAN and IS-DBSCAN. When the data size is more than 10,000, the execution time of DBSCAN and IS-DBSCAN increases significantly, but GISN-DBSCAN still increases slightly. And this demonstrates that the extended range query algorithm can significantly improve the efficiency of GISN-DBSCAN, and substantially reduce the cost of range query. Fig 7 indicates that the correct rate of DBSCAN dropped significantly with the data size increasing, but the correct rates of IS-DBSCAN and GISN-DBSCAN, which are much higher than that of DBSCAN, dropped slightly. Table IV shows that there is no obvious difference between the correct rates of these algorithms in Iris and Harberman's Survival. The reason is that the data sizes and dimensions of Iris and Harberman's Survival are small. When dealing with Letter Recognition, the correct rate of GISN-DBSCAN is much higher than these of DBSCAN and IS-DBSCAN.

C. Comparison of Clustering Quality

Aggregation Dataset is used in this subsection. Fig 8 indicates the comparisons in identifying adjacent clusters with various densities. Fig 9 shows the comparisons of the different starting data points.

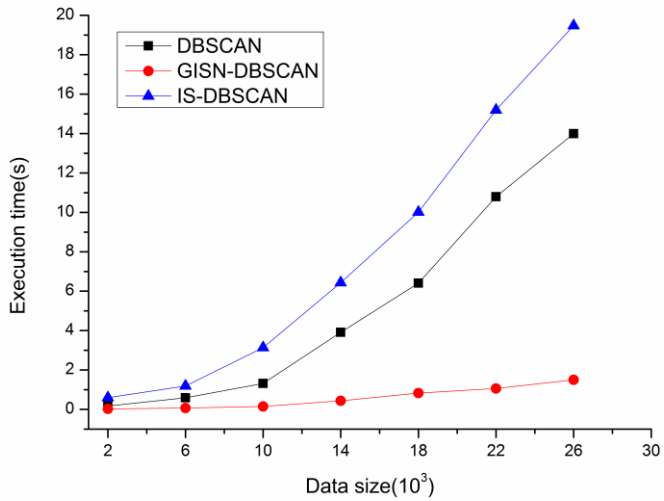

Fig 6. Comparison of execution time

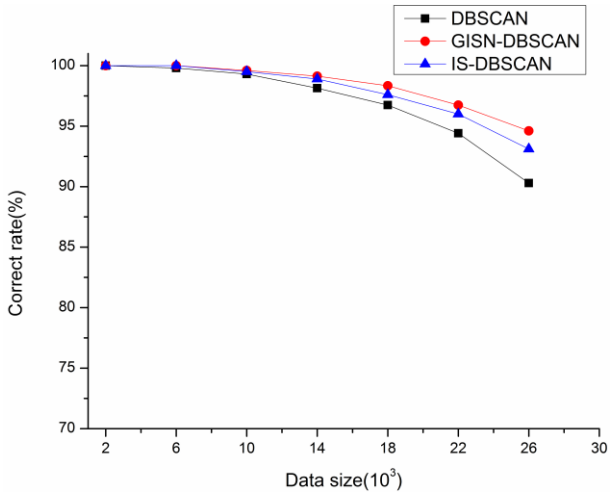

Fig 7. Comparison of correct rates

Table IV
COMPARISON OF CORRECT RATES IN VARIOUS DATASETS

Dataset	DBSCAN(%)	IS-DBSCAN(%)	GISN-DBSCAN(%)
Iris	99.68	99.74	99.81
Harberman's Survival	93.45	96.78	98.13
Letter Recognition	80.02	87.89	95.43

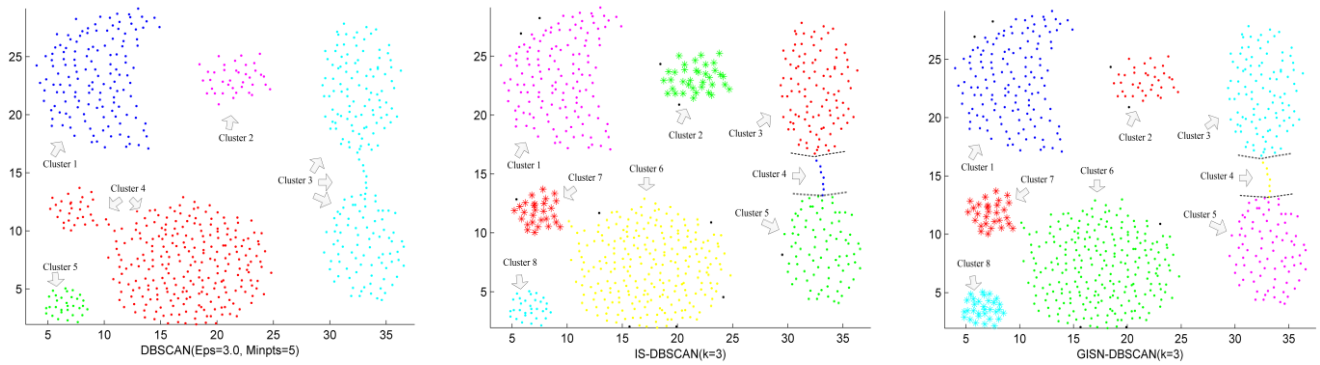

Fig 8. Comparison in identifying adjacent clusters of various densities of DBSCAN, IS-DBSCAN, and GISN-DBSCAN

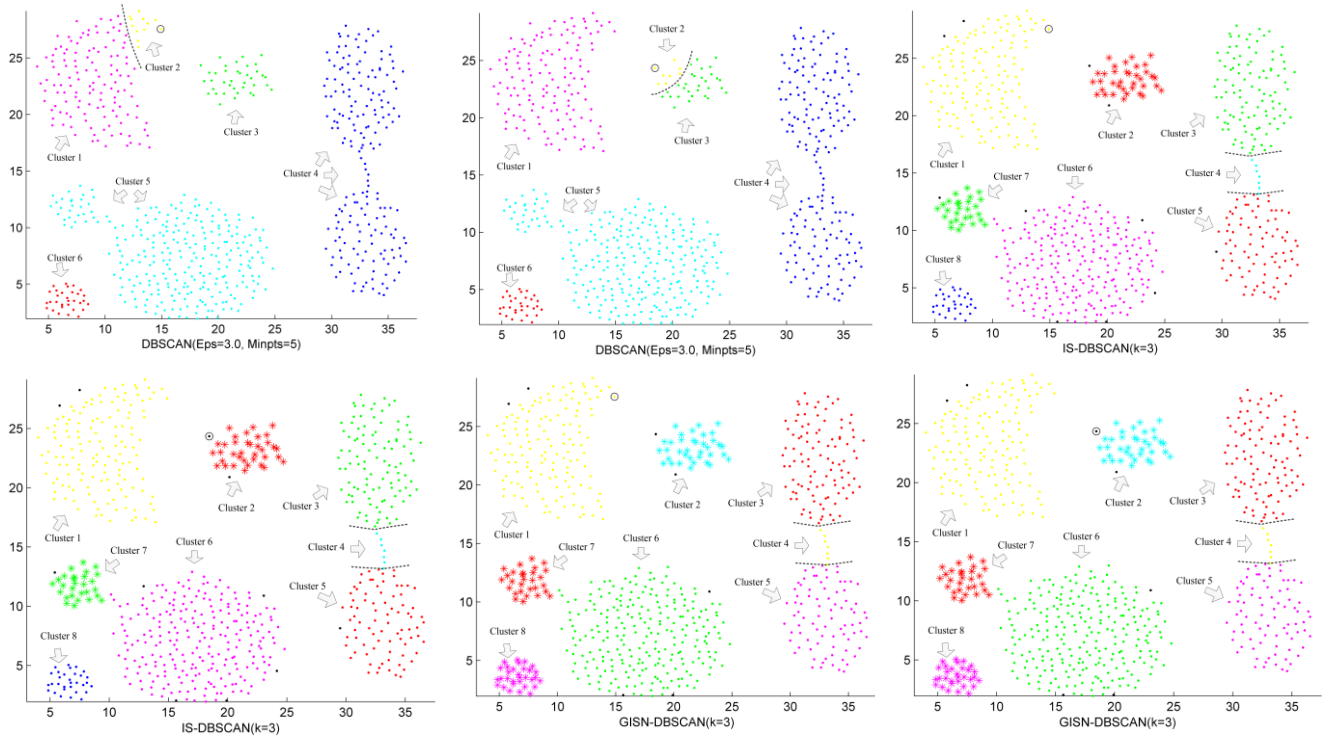

Fig 9. Comparison of the different starting data points of DBSCAN, IS-DBSCAN, and GISN-DBSCAN

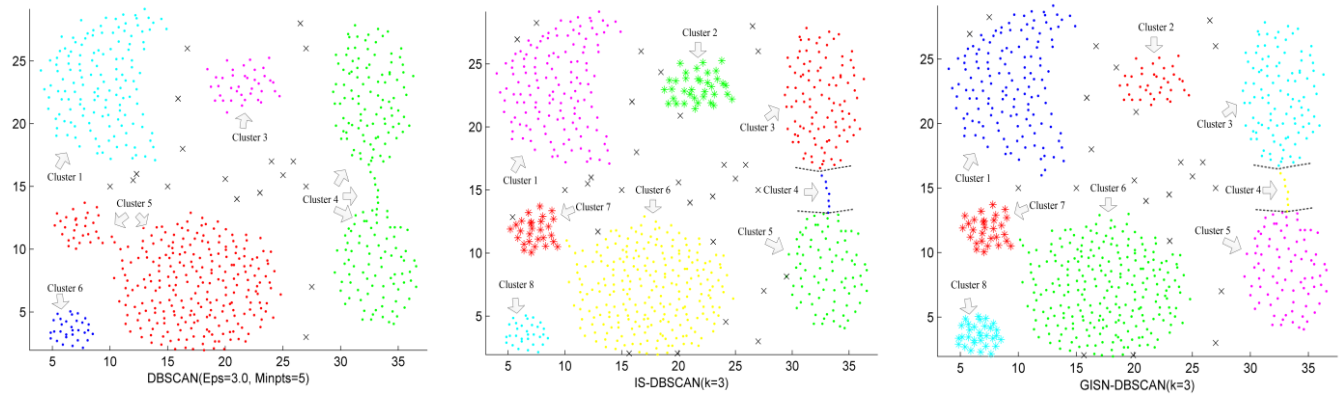

Fig 10. Comparison in detecting border points of DBSCAN, IS-DBSCAN, and GISN-DBSCAN

Fig 8 shows that IS-DBSCAN and GISN-DBSCAN are sensitive to the adjacent clusters with various densities which can be detected effectively. But DBSCAN cannot detect the adjacent clusters and gather the adjacent clusters with various densities into one cluster. The reason for this is that DBSCAN algorithm needs two global parameters *Eps* and *Minpts*. When the density of cluster changes, two global parameters cannot be applied to the clusters whose densities have changed. Fig 9 indicates that when the different starting points are selected (the black circles are the different starting points), the different results will be obtained by DBSCAN. However, the results obtained from IS-DBSCAN and GISN-DBSCAN are the same. The reason for this is that directly density reachable and density reachable are asymmetric for DBSCAN, that is if x_i is directly density reachable (or density reachable) from a core point x_j, maybe x_j is not directly density reachable (or density reachable) from x_i. This characteristic of DBSCAN can explain that different starting points will result in different effects. However, the points in ISN_k belong to the nearest neighbors and the reverse nearest neighbors, and ISN_k is used in GISN-DBSCAN. Moreover both the directly density reachable and density reachable have the characteristic of symmetric. Thus, GISN-DBSCAN always obtains the same clustering results even if different starting points are selected.

D. Border Points Detecting

Aggregation* Dataset is used in this subsection. Fig 10 indicates the comparisons in detecting border points.

Fig 10 shows that both DBSCAN and GISN-DBSCAN successfully detect border points and noise points whereas IS-DBSCAN fails in detecting them. The reason for this is that the number of points in *k*-influence space of p_i is less than k (i.e. $|IS_k(p_i)| < k|$), then p_i is considered to be noise point. However, when $|IS_k(p_i)| < k|$, maybe p_i is border point or noise point. Thus, compared with IS-DBSCAN, GISN-DBSCAN can detect border points with OLF_k successfully and improve the clustering quality significantly.

V. Conclusions And Future Works

In this paper, we have proposed an enhanced density clustering algorithm for datasets with complex structures, named GISN-DBSCAN. At the same time, an extended range query algorithm based on fixed-grids is presented firstly. In our work, the nearest neighbors (NN_k) and reverse nearest neighbors (RNN_k) are used to establish the *k*-influence space neighborhood of each point. Moreover, a computational method of *k*-outlierness function is proposed, named OLF_k. Experimental results demonstrate that GISN-DBSCAN can reduce the time overhead of searching the nearest neighborhood, identify the adjacent clusters with different densities and detect the border points and noise points effectively, and it is also not sensitive to the density of the starting point. Future works will be devoted to the parallel density-based clustering algorithm using MapReduce for big data.

Acknowledgment

The authors would like to thank the Intelligent Information Processing Group for supporting the implementation of proposed algorithms and tools.

References

[1] I. Cadez, P. Smyth and H. Mannila, "Probabilistic modeling of transaction data with applications to profiling, visualization, and prediction," in *Proc. 7th ACM SIGKDD Conf. Knowledge Discovery and Data Mining*, San Francisco, 2001, pp. 37–46.

[2] R. Xu and D. Wunsch, "Survey of clustering algorithm," *IEEE Transactions on Neural Networks*, vol. 16, no. 3, pp. 645–678, May. 2005.

[3] J. M. Yang, Z. Y. Liu and Z. Y. Qu, "Clustering of words based on relative contribution for text categorization," *IAENG International Journal of Computer Science*, vol. 40, no. 3, pp. 207–219, 2013.

[4] J. M. Yang, J. Wang and Z. Y. Qu, "Feature selection method based on the relative contribution," *Journal of Northeast Dianli University*, vol. 34, no. 4, pp. 62–68, Aug. 2014.

[5] A. Chaturvedi, P. E. Greed and J. D. Caroll, "K-modes clustering," *Journal of Classification*, vol. 18, no. 1, pp. 35–55, Jan. 2001.

[6] R. Gelbard, O. Goldman and I. Spiegler, "Investigating diversity of clustering methods: An empirical comparison," *Data & Knowledge Engineering*, vol. 63, no. 1, pp. 155–166, Oct. 2007.

[7] M. Nanni and D. Pedreschi, "Time-focused clustering of trajectories of moving objects," *Journal of Intelligent Information Systems*, vol. 27, no. 3, pp. 267–289, Nov. 2006.

[8] E. W. M. Ma and T. W. S. Chow, "A new shifting grid clustering algorithm," *Pattern Recognition*, vol. 37, no. 3, pp. 503–514, Mar. 2004.

[9] J. G. Sun, J. Liu and L. Y. Zhao, "Clustering algorithms research," *Journal of Software*, vol. 19, no. 1, pp. 48–61, Jan. 2008.

[10] M. Ester, H. P. Kriegel, S. Jiirg and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases," in *Proc. 2th International Conference on Knowledge Discovery and Data Mining*, Oregon, 1996, pp. 226–231.

[11] A. Ram, A. Sharma, A. S. Jalal and A. Agrawal, "An Enhanced Density Based Spatial Clustering of Applications with Noise," in *IEEE International Advance Computing Conference*, Patiala, 2009, pp. 1475–1478.

[12] A. H. Pilevar and M. Sukumar, "GCHL: a grid-clustering algorithm for high-dimensional very large spatial data bases," *Pattern Recognition Letters*, vol. 26, no. 7, pp. 999–1010, May. 2005.

[13] M. M. Breunig, H. P. Kriegel, R. T. Ng and J. Sander, "LOF: identifying density-based local outliers," in *Proc. 2000 ACM SIGMOD International Conf. on Management of Data*, New York, 2000, pp. 93–104.
[14] W. Jin, A. K. Tung, J. Han and W. Wang, "Ranking outliers using symmetric neighborhood relationship," in *Proc. 10th Knowledge Discovery and Data Mining*, Pacific, 2006, pp. 577–593.
[15] C. Cassisi, A. Ferro, R. Giugno, G. Pigola and A. Pulvirenti, "Enhancing density-based clustering: parameter reduction and outlier detection," *Information Systems*, vol. 38, no. 3, pp. 317–330, May. 2013.
[16] S. F. Liu, X. D. Meng and X. Y. Wang, "DBSCAN algorithm based on grid cell," *Journal of Jilin University (Engineering and Technology Edition)*, vol. 44, no. 4, pp. 1135–1139, Jul. 2014.
[17] A. Gionis, H. Mannila and P. Tsaparas, "Clustering aggregation," *Acm Transactions on Knowledge Discovery from Data*, vol. 1, no. 1, pp. 341–352, Mar. 2007.
[18] UCI Machine Learning Repository is free access is available at http://archive.ics.uci.edu/ml/

Jieming Yang received his MSc and PhD degree in computer applied technology from Northeast Electric Power University, China in 2008 and Computer Science and Technology from Jilin University, China in 2013, respectively. His research interests are in machine learning and data mining.

Qilong Wu is currently pursuing his MSc degree in computer applied technology from Northeast Electric Power University, China. His research interests are in data mining and pattern recognition.

Zhaoyang Qu received his MSc and PhD degree in computer science from Dalian University of Technology, China in 1988 and North China Electric Power University, China in 2012, respectively. His research interests are in artificial intelligence, machine learning and data mining.

Zhiying Liu received her MSc degree in computer applied technology from Northeast Electric Power University, China in 2005. Her research interests are in information retrieval and personalized recommendation.