TELKOMNIKA, Vol. 11, No. 12, December 2013, pp. 7302~7308
e-ISSN: 2087-278X

Clustering-boundary-detection Algorithm Based on Center-of-gravity of Neighborhood

Wang Gui-Zhi*, Zhang Jian-Wei
Department of Computer Application, Henan University of Animal Husbandry and Economy, Zhong zhou 450044, No.2, Yingcai Street, Hui Ji district, Zhengzhou, He Nan province, China
Phone: 0371-63515139
*Corresponding author, e-mail: 522829031@qq.com

Abstract

The cluster boundary is a useful model, in order to identify the boundary effectively, according to the uneven distribution of data points in the epsilon neighborhood of boundary objects, a boundary detection algorithm-S-BOUND is proposed. Firstly, all the points in the epsilon neighborhood of the data objects are projected onto the boundary of the convex hull of the neighborhood, and then calculate the center of gravity of the neighborhood. Finally, detect the boundary object according to the degree of deviation of the center of gravity of the neighborhood with the object. The experimental results show that the S-BOUND algorithm can accurately detect a variety of clustering boundary and remove the noises, the time of performance is also better.

Keywords*: cluster, boundary object, neighborhood, projection, neighborhood center of gravity*

1. Introduction

The so-called clustering is that according to some similar measure, divide the data object into a plurality of classes or clusters, making the objects in the same cluster has very high degree of similarity and the objects in different cluster is very different [1]. As an important method of data mining, clustering analysis has been widely used in pattern recognition, graphics and image processing, etc. In addition, it has been widely used in practical work. The collection of data objects located on the edge of the region of high-density of clustering, constitutes the cluster boundary. Boundary objects often have two or more features of clusters, of which the attribution is not clear [2]. To effectively extract cluster boundaries not only can improve the accuracy of clustering, but also study further the characteristic of the edge of cluster. Therefore, Clustering boundary has become one of the emerging areas of research of data mining.

DBSCAN [3] algorithm is the earliest to propose the concept of clustering boundary. DBSCAN algorithm is clustering algorithm based on density. The algorithm starts from a core object and iteratively seeks all objects that are directly density-reachable, and stops until meet non-core object iteration, to form Clustering. We can say, if a non-core object *p* can be directly density-reachable through core objects *q*, then *p* is the boundary point. However, DBSCAN algorithm just proposes clustering strategy, but does not give the detection method of boundary point.

BORDER [3] algorithm first proposes the use of reverse k-near neighbor of object to detect the boundary points. In the algorithm, if an object *p* is in the *k*-near neighbor of an object *o*, we call the object o is the reverse *k*-near neighbor of object p. Because the number of k-near neighbors of internal objects of clustering is more than that of clustering boundary objects, BORDER algorithm first calculates the number of reverse k-near neighbor of each object, then according to the number reverse k-near neighbors, in the order from small to large, arrange the whole data set, taking n objects as boundary points. However, since the number of reverse k-near neighbors of noise point is often smaller than that of reverse k-near neighbors of boundary points, when the entire data set is arranged, the noise points are also among the first *n* data objects, so BORDER algorithm cannot identify noise, and its significant deficiency lies in that it cannot effectively extract the boundary data set that contains noise.

Received June 24, 2013; Revised July 13, 2013; Accepted August 13, 2013

bDBSCAN [4] algorithm is to conduct projection and vector unitization of objects in the neighborhood of core objects, discriminate the balance of the neighborhood of core object and clustering the equilibrium-density-reachable objects of balance core objects. The algorithm defines imbalance objects in neighborhood as boundary object, effectively eliminating the type of noise of boundary sparse objects and improving clustering accuracy. However, the algorithm just conducts clustering, but does not conduct cluster boundary extraction. In recent years, the experts also propose many related cluster boundary detection algorithms, like IBORA [5] algorithm, BRIM [6] algorithm, Green [7] algorithm, BRINK [8] algorithm. These algorithms use different strategies to identify the boundaries of objects, which has validity, but also some limitations.

2. S-BOUND Algorithm

S-BOUND is a new algorithm, with the help of projection [9] to take boundary detection. It gives the definition of projection again, using the concept "center of gravity" in physics to discriminate the balance neighborhood core object, so as to identify boundary objects.

2.1. Theoretical Basis

In the physical mechanics, each part of an object should bear the role of gravity. The so-called center of gravity is that in the gravitational field, for the object in any position, the resultant force of gravity that constitute mass points, all through the point. The center of gravity of regular object of uniform density is its geometric center and the position of the center of gravity influences the balance and stability of the object.

Observing *ε* neighborhood of clustering internal objects, you can find its data object is relatively uniformly distributed, the center of gravity is the center of the neighborhood and the *ε* neighborhood is balanced; However, the data objects in *ε* neighborhood of clustering boundary object is unevenly distributed, with one side the high-density area, the other side the low-density areas. Therefore, the gravity center of its neighborhood is deviated from the neighborhood center and it is in the high density area. The *ε* neighborhood is uneven. Therefore, we can use the deviation degree of the center of gravity and geometric center of *ε* neighborhood of the core object to identify boundary objects, thereby to extract boundaries.

2.2. Related Definitions and Terminology

Definition 1: *ε*-neighborhood [3]. For any data object *p* in space *D*, the area within the radius *ε* is called *ε* neighborhood of the object, recorded as *Nε (p)*. Namely:

$N\varepsilon(p)=\{q|q \in D \text{ and } dist(p,q)\le\varepsilon\}$

Wherein *dist (p, q)* represents the Euclidean distance between *p* and *q*.

Definition 2: Core object [3]. Given *ε* and MinPts, if the number of objects of *ε* neighborhood *Nε (p)* of object p is $|N\varepsilon(p)|\ge MinPts$, then P is called the core object.

Definition 3: Projection. In the ε-neighborhood of core object, for any object $p \in N\varepsilon(q)$, draw a ray with *q* as the starting point to be through *q*, let the ray intersect with the *ε*-neighborhood boundary *q* at point *p*, namely $dist(p',q)=\varepsilon$. The process is called projection, wherein the point p 'called the subpoint of object p on the ε neighborhood.

For the d-dimensional data set *D*, in the *ε*- neighborhood of core object $q (x_{q1}, x_{q2}, ..., x_{qd})$, the projection of object $p (x_{p1}, x_{p2}, ..., x_{pd})$ is the projection of each attribute on every dimension. Assume the projection point as $p'(x_{p1'}, x_{p2'}, ..., x_{pd'})$, then:

$$x_{pi'}= \varepsilon \cdot \cos\theta_i \quad (1\le i\le d)$$

Wherein θi is the included angle of ray qp and unit vector Ii on the i- dimension [7].

And,

$$\cos\theta_i= \frac{x_{pi} - x_{qi}}{dist(p, q)}$$

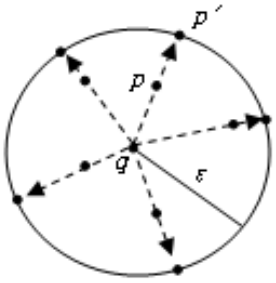


Figure 1. Neighborhood Projection of
two-dimensional data objects

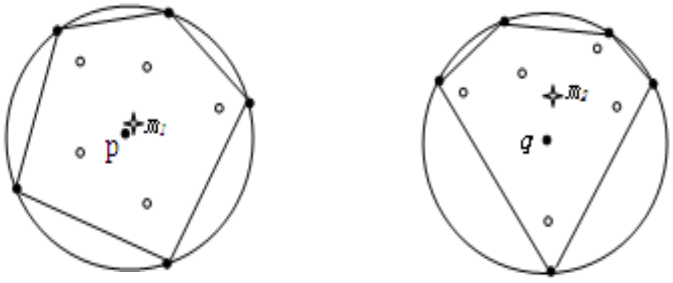
In bDBSCAN algorithm, in the neighborhood of two-dimensional space, object is projected onto the unit circle. In order to improve the calculation accuracy of the neighborhood equilibrium, the article put the projection point on the ε-circle, rather than the unit circle. As shown in Figure 1, in *ε-neighborhood* of two-dimensional space object q, object p is projected onto point *p′* of *ε*-circle. The other objects are projected in the same way onto the *ε*-circle.

Definition 4: Center of gravity of neighborhood. For the core object *q*, the projection points of all objects of its ε-neighborhood are sequentially connected, thus forming epiboly convex hull. The center of gravity of convex hull is defined as the center of gravity of *ε*-neighborhood of object q, recorded as point m.

For the d-dimensional data set *D*, projection point of object p_j in *ε*-neighborhood of object *q* is recorded as $pi'(x_{pj1'}, x_{pj2'}, \ldots, x_{pjd'})$.Its neighborhood center of gravity $m\ (x_{q1'}, x_{q2'}, \ldots, x_{qd'})$ is recorded as *m*, then:

$$x_{qi'} = \sum_{j=1}^{n} x_{pji}' \quad (1 \le i \le d,\ n = |N\varepsilon(q)|)$$

Obviously, the center of gravity *m* is closer to object *q*, indicates that the points in the *ε*-neighborhood of object *q* are more evenly distributed, otherwise the farther, more unevenly, namely neighborhood is unbalanced. Figure 2 is schematic diagram of center of gravity of *ε*-neighborhood of two-dimensional object *q*. The projection points (solid dots) of the objects (open dots) in the *ε*-neighborhood of object *p* in Figure 2(a) on ε-circle, form a convex polygon. The center of gravity m_1 of the polygon is very close to object *p*, so its neighborhood are balanced; However, the center of gravity m_2 of the convex polygon formed by subpoints of the objects in *ε*-neighborhood of object *q* in Figure 2(b), is very far from object *p* and Its neighborhood is uneven.



(A) Balanced neighborhood (B) Imbalanced neighborhood

Figure 2. Data Objects Neighborhood Equilibrium

Definition 5: Boundary Objects. when the distance between object *q* with center of gravity m of *ε*-neighborhood is greater than the given value *η*, object *q* is called boundary object.

3.3. Algorithm Steps Description

S-BOUND algorithm is according to the above ideas and definitions of boundary object to identify boundaries. The specific steps are descriptions are as follows:

(1) Input data set *D*, neighborhood radius *ε*, core object neighborhood threshold *MinPts* and boundary threshold *η*;

(2) Scan data set, repeat steps (2) - (3);

(3) Calculate the *ε*-neighborhood of data object *q* , if | *Nε (q)* | ≥ *MinPts*, execute step (4)-(6);

(4) Projection. Take object *q* as the center to establish coordinate system, project all objects in *Nε (q)* on the neighborhood boundary;

(5) Calculate center of gravity *m* of neighborhood according to definition 3;

(6) Calculate the distance between the center of gravity *m* and *q*, when | *mq* |> *η* , *q* is the boundary object;

(7) Output boundary object.

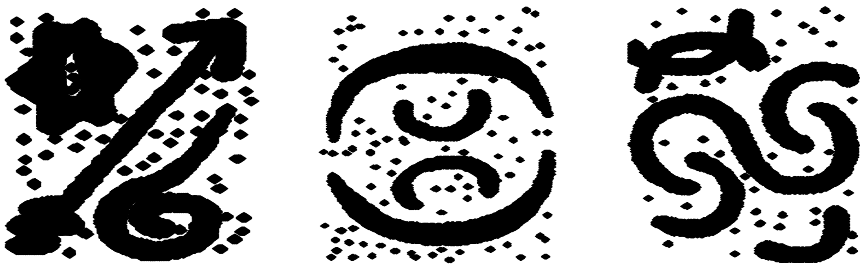
3. Experiment Situation Analysis

In order to verify the correctness and efficiency of algorithm, the article adopts a plurality of two-dimensional comprehensive data set to conduct experiment. For experimental environment, it use: PDC E5800 CPU, 3G memory, Windows 7 Home Edition operating system. Preparation and implementation of the algorithm is conducted in VC++ 6.0 environment.

3.1. Validity of the Algorithm

The article uses multiple synthetic data sets to test effectiveness of S-BOUND and compare with the typical boundary detection algorithm BORDER experimental results.

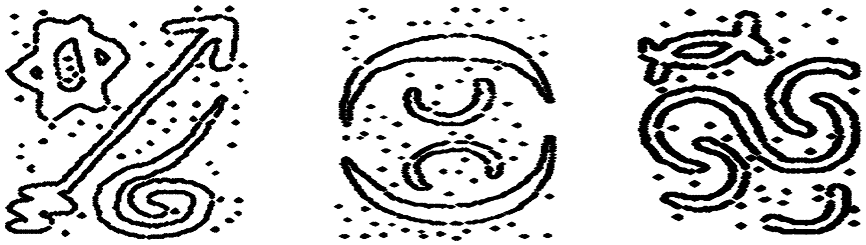
Figure 3 is three data sets with noise and uniform density. Wherein in (a) there are 12,917 data points, naturally forming three clusters; in (b) there are 8486 data points, naturally forming four clusters; in (c) there are 11,182 data points, naturally forming five clusters. The three data sets are with clear boundaries and various shapes, which could test the validation of algorithm.



 a b c

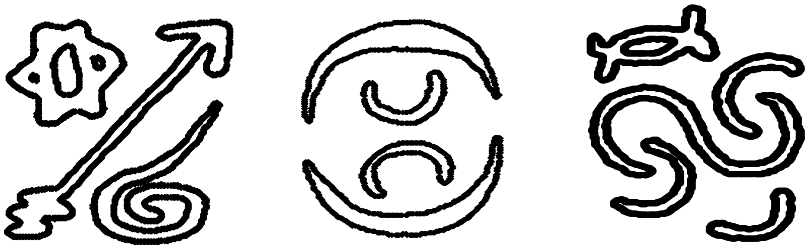
Figure 3. Original Date Set

Figure 4 is the computational results of the three data sets shown in Figure 3 by BORDER algorithm,wherein the parameters used by (a) are: the number neighbor *k=8*, the number of boundary points *n=1925*; the parameters used by (b) are: the number neighbor *k=8,* the number of boundary points *n=1390*; the parameters used by (c) are: the number neighbor *k=8*, the number of boundary points *n=1799*;As can be found from the figure, the "boundary points" detected BORDER algorithm include noise and the boundary point cannot be distinguished from the noise area.



（a） （b） （c）

Figure 4. BORDER Algorithm Results



（a） （b） （c）

Figure 5. S-BOUND Algorithm Results

Figure 5 is the computational results of the three data sets shown in Figure 3 by BORDER algorithm, wherein the parameters used by (a) are: neighborhood radius *ε=3*, core object neighborhood threshold *MinPts=12*, the boundary threshold *η=0.052*, the number of boundary points detected is 1756;the parameters used by (b) are: neighborhood radius *ε=2.3*, core object neighborhood threshold *MinPts=7*, the boundary threshold *η=0.075*, the number of boundary points detected is 1043; the parameters used by (c) are: neighborhood radius *ε=2.2*, core object neighborhood threshold *MinPts=7*, the boundary threshold *η=0.095*, the number of boundary points detected is 1546; It can be found from the diagram, S-BOUND algorithm can effectively detect the boundary objects in noise data set and the boundaries identified are clear, very well reflecting the shape characteristic of the boundary of each cluster.

（a） （b） （c）

Figure 6. Boundary Detection Case of Complex Data Sets

In the above three used data sets, the density of each cluster is very high and the data objects are evenly distributed, while the noise objects are relatively rare. In order to test the effectiveness of S-BOUND algorithm, we have also used complex data sets [7] shown in Figure 6(a) to conduct experiments. The original data set has 23 724 points and includes a lot of noise and "interfering line". The internal objects of each cluster are distributed very sparsely and not that evenly. (b) is the result detected by algorithm BORDER, the parameters used are: the number of neighbor *k=140*, the number of boundary points *n=4500*; (c) is the result detected by algorithm S-BOUND, the parameters used are: neighborhood radius *ε=8.6*, core object neighborhood threshold *MinPts=128*, boundary threshold *η=0.0064*, the number of

boundary points detected is 3837. It can be found by comparing the results of the two experiments that boundary points detected by BORDER algorithm contain a lot of noise and could not very well identify boundary outline of each cluster; While S-BOUND algorithm can effectively distinguish noise, remove the "interfering lines" and detect the true boundaries of each cluster. Thus, the effectiveness of the algorithm is verified.

3.2. Time Analysis of Algorithm

S-BOUND algorithm scans through the data set to calculate the ε-neighborhood of data objects and detect boundary objects by the projector, its time complexity is $O(n^2)$; while BORDER algorithms require scanning twice dataset, to calculate the reverse k-nearest neighbor of each data object, and sort all the data object according to the reverse k-nearest neighbor, algorithm's time complexity is $O(kn^2)$. Therefore BORDER algorithm's time complexity is higher than S-BOUND algorithms.

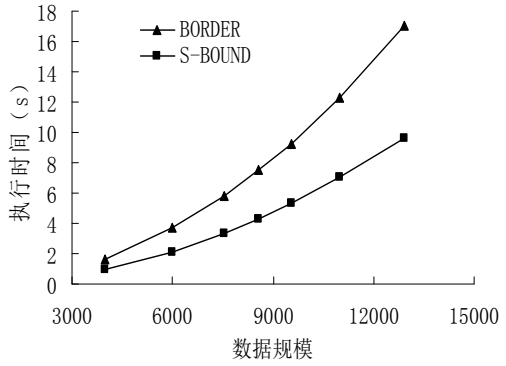


Figure 7. Time Efficiency Comparing Two Algorithms( )

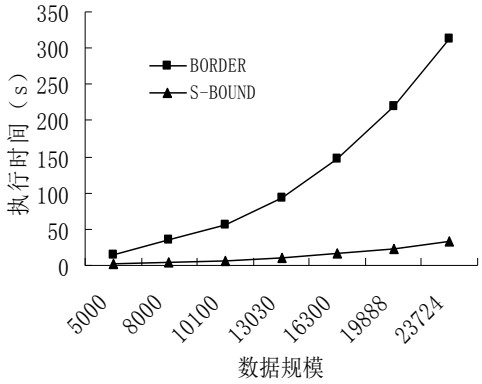


Figure 8. Time efficiency comparing two algorithms( )

In order to verify the time efficiency of the S-BOUND algorithm, we conducted experiments compared to the BORDER algorithm with the original data set shown in Figure 3 (a). 4000, 6000, 7550, 8550, 9550, 11000 and 12917 (all) data points were selected, the results shown in Figure 7. As can be seen, S-BOUND algorithm's time efficiency is more than BORDER algorithms.

In above experiments, the data points of each cluster is uniform and density, parameter values used in the two algorithms are small; in the data set shown in Figure 6, data objects distributed leaner and less evenly, parameter values used in the two algorithms are larger. To better validation S-BOUND algorithm's time performance, Experiments were executed with the data set shown in Figure 6. 5000, 8000, 10100, 13030,16300, 19888 and 23724 (all) data points were selected, the comparative results shown in Figure 8.

As can be seen from Figure 8, to S-BOUND algorithm, although the parameters are vary greatly in the two types of data sets, but the execution time is basically the same; However, the time complexity of BORDER algorithm is $O(kn^2)$, when used the data sets shown in Figure 6(a), the experiments need a larger value of the parameter ($k=140$), and the time performance is far below the S-BOUND algorithms.

4. Conclusion

In the article, all the data points are projected in the ε-neighborhood of the core object of the data set onto the convex hull of the epiboly and at the same time, identify boundary objects with the concept of center of gravity in physics. The experimental results show that the method can effectively identify cluster boundaries of various shapes in the data set that contains noise, and its time efficiency is greater than that of BORDER algorithm. Especially for the cluster with low density and not that evenly distributed, S-BOUND algorithm has more time advantage than BORDER algorithm. However, S-BOUND algorithm has an apparent defect - more parameters. when the algorithm is executed, in addition to the neighborhood radius ε and boundary

threshold *η*, it needs neighborhood threshold *MinPts* of Core object as the input parameter, in order to remove noise.

References

[1] MS Chen, JH Han, PS Yu. Data mining: an overview from a database perspective. *IEEE Trans KDE*. 1996; 8(6): 866-883.

[2] Xia Chenyi, Hsu W, Lee Mongli, et al. BORDER: efficient computation of boundary points. *IEEE Transon Knowledge and Data Engineering*. 2006; 18(3): 289-303.

[3] Ester M, Kriegel HP, Sander J, et al. *A density-based algorithm for discovering clusters in large spatial database with noise*. Proceedings of 2nd International Conference on Knowledge Discovering and Data Mining (KDD-96). Portland: Oregon. 1996: 226-231.

[4] Wu Jiawei, Li Xiongfei, Sun tao, etc. A Density-Based Clustering Algorithm Concerning Neighborhood Balance. *Journal of Computer Research and Development*. 2010; 47(6): 1044-1052.

[5] WU SHOUR, Silamu, LI Fengjun, TAO Mei. IBORA: an Improved Efficient Detection of Boundary Points. *Journal of Chinese Computer Systems*. 2008; 29(10): 1845-1848.

[6] Qiu Baozhi, Yue Feng, Shen Junyi, etc. *BRIM: An Efficient Boundary Points Detecting Algorithm.* Proc.Of Advances in Knowledge Discovery and Data Mining. Heidelberg: Springer. 2007; 761-768.

[7] Qiu Baozhi, YueFeng. Gravity-based Boundary Points Detecting Algorithm. *Journal of Chinese Computer Systems*. 2008; 29(9): 279-282.

[8] Qiu Baozhi, Yang Yang, Du Xiaowei. BRINK: An Algorithm of Boundary Points of Clusters Detection Based On Local Qualitative Factors. *Journal of Zhengzhou University (Engineering Science)*. 2012; 33(3): 117-120.

[9] Zhou Jingbo, Yin Jun, Jin Zhong. New Ensemble Constructor Based on Locality Preserving Projection for High Dimensional Clustering. *Computer Science*. 2011; 38(9): 177-181.