Proceedings of the Second International Conference on Machine Learning and Cybernetics, Xi'an, 2-5 November 2003

CUBN: A CLUSTERING ALGORITHM BASED ONDENSITY AND DISTANCE

LI WANG, ZHENG-OU WANG

Institute of Systems Engineering, Tianjin University, Tianjin, China 300072
E-MAIL: wangli9966@eyou.com, zowang@public.tpt.tj.cn

Abstract:

In data mining, clustering is used to discover groups and identify interesting distribution in the underlying data. Traditional clustering algorithms favor clusters with spherical shapes and similar sizes. We propose a new clustering algorithm called CUBN that integrates density-based and distance-based clustering. Firstly, CUBN finds border points by using erosion operation that is one of the basic operations in mathematical morphology, then, it clusters the border points and inner points according to the nearest distance. Our experimental results show that CUBN can identify clusters having non-spherical shapes and wide variances in size, and its computational complexity is O(n). Therefore, this algorithm facilitates the clustering of a very large data set.

Key words:

Data mining, Clustering, Erosion operation

1 Introduction

Clustering is the process of grouping the data into clusters so that objects within a cluster have high similarity, but are very dissimilar to objects in other clusters. The similarity of the objects is determined by a cardinal similarity measurement over the object attributes. The term "clustering" originates from many areas, including data mining, statistics, biology, and machine learning. In data mining, clustering is a very useful tool when the analyst is faced with a large, complex data set containing many variables and a complicated unknown structure. At the beginning of a data mining operation, clustering would often be the best technique to use. Once automatic cluster detection has discovered regions of the data space that contain similar records, other data mining tools and techniques could be used to discover rules and patterns within the clusters [1].

Clustering algorithms have been investigated previously and cover a vast range of methodologies. There exist a large number of clustering algorithms in the literature. In general, major clustering methods can be classified into distance-based methods and density-based methods by the clustering criteria. The goal of distance-based methods is that the distance of the objects within a cluster is small, whereas that of the objects among different clusters is large. The typical distance-based methods, such as k-means and k-medoids, are relatively scalable and efficient in processing large data sets, but they only work well for finding clusters with spherical shapes and similar sizes. To discover clusters with arbitrary shape, many improvement algorithms have been proposed recently. These algorithms, such as CURE[2] and BIRCH[3], produce high-quality clusters in the existence of outliers, allowing clusters of complex shapes and different sizes. However, the computational complexity of them are $O(n^2)$. The density-based clustering methods regard clusters as dense regions of objects in the data space that are separated by regions of low density. Those algorithms, such as DBSCAN[4] and OPTICS[5], can discover clusters with arbitrary shape. However the computational complexity are $O(n^2)$. Another algorithm CLIQUE[6] partitions the n-dimensional data space into grid units, and identifies the dense units among these. The computational complexity of the algorithm is O(n), but the accuracy of the clustering result may be degraded at the expense of the simplicity of the method.

In this paper, we propose a new clustering method named CUBN (Clustering Using Border and Nearest). The algorithm integrates density-based and distance-based clustering. Firstly, CUBN finds border points in data space, and then it clusters the border points and inner points according to the nearest distance. CUBN can identify clusters having non-spherical shapes and wide variances in size, and it's computational complexity is O(n).

The remainder of the paper is organized as follows. Section 2 introduces the mathematics theory of our algorithm, and we present CUBN algorithm in detail and discuss the algorithm effectiveness in section 3. Then in section 4, we present the results of our experiments, which support our claim about CUBN's clustering ability. Finally concluding remarks are made in section 5.

2 Mathematics theory of CUBN

Clustering is the process of grouping the data in

0-7803-7865-2/03/$17.00 ©2003 IEEE

d-dimension space into clusters, and cluster is the dense region of objects in the data space. Obviously, if the borders of each cluster are detected, the clusters are identified. The erosion operation in mathematical morphology[7] gives the elicitation to detect the border.

2.1 Erosion operation

Erosion operation is one of the seven basic operations in mathematical morphology, and it has visual meanings for geometry. In Fig.1, ellipse A is the researched object; circle B is the "structure element" or "template". Let B move along the A's border in the inside of A, then the part surrounded by the track of B's center is the result of A eroded by B. As shown in Fig.1, the result (the shadow part) retains the basic shape of A, and the roughness of the border is eliminated.

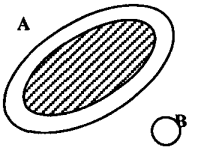


Figure.1 The erosion operation

The following is the mathematics description of erosion operation:

When B is a single vector, the erosion operation is defined as:

$A \ominus B = \{ a \mid a \in A, n>t \text{ and } n = \#\{ w \mid w \in O(a+B, r) \cap A \} \}$

where t is a threshold; O is a super-sphere; a+B is the center of the sphere, and r is the radius of the sphere.

When B is a vector set (B1, B2, . . . , B_n), where B_i (i = 1, 2, . . . , n) is a single vector, the erosion operation is defined as:

$$A \ominus B = (A \ominus B_1) \cap (A \ominus B_2) \cap \ldots \cap (A \ominus B_n)$$

i.e. $$A \ominus B = \bigcap_{i=1}^{n} A \ominus B_i$$

2.2 Detect the cluster border

If A is a cluster, then the border points set C is defined as

$$C = A - \overline{A \ominus B} = A \ominus B = \bigcup_{i=1}^{n} \overline{A \ominus B_i}$$

where $\overline{A \ominus B_i} = \{a \mid a \in A, n<=t \text{ and } n = \#\{ w \mid w \in O(a+B, r) \cap A\}$. In CUBN algorithm, let

$$B = \begin{bmatrix} p & 0 & \cdots & 0 \\ -p & 0 & \cdots & 0 \\ 0 & p & \cdots & 0 \\ 0 & -p & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & p \\ 0 & 0 & \cdots & -p \end{bmatrix}_{2d \times d}$$, where d is the dimension of

data space. B_i is B's row vector, that is, A is eroded by positive and negative direction in each dimension. Let t=1, that is, a is a border point if the super sphere is empty.p is the size of B_i, and p should satisfy the following condition:

$$\max(\text{distance}(x_i, x_j)) < p+r < \min(\text{distance}(A_i, A_j))$$

where distance (x_i, x_j) is the distance between point x_i and x_j in a same cluster, distance (A_i, A_j) is the distance between cluster A_i and A_j. It can be computed as

$$\text{distance}(A_i, A_j) = \min_{x \in Ai, y \in Aj} |x-y|$$

if p+r > min (distance (A_i, A_j)), CUBN algorithm can not detect all border points; if p+r < max (distance (x_i, x_j)), CUBN algorithm may regard an inner point as a border point.

To detect the different cluster border, the super-sphere radius r is taken to be the value $p/\sqrt{2}$. Fig.2 illustrates the reason using an example in 2-dimesional space. In Fig.2, the right of the skew line is the inside of a cluster. The dashed line is the axis. The point (marked as shaded) is a border point, and the hollow point is an inner point. The circle is the super-sphere, a+p is its center, and r is its radius.

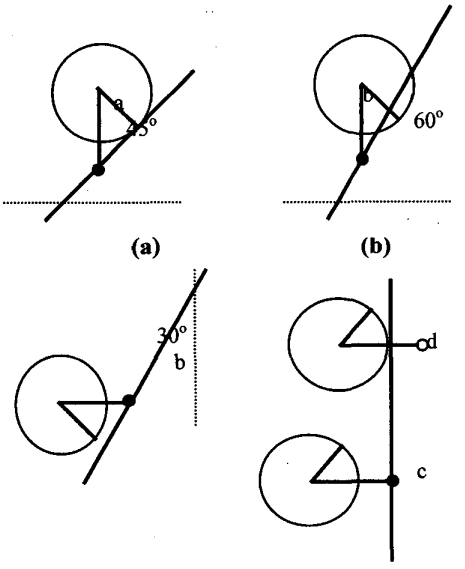


Figure.2 the radius of super-sphere

In Fig.2 (a), the circle and the cluster intersect at a, so a must be a border point. In Fig.2 (b), b is not been detected in vertical direction because the super-sphere is not empty. However it can be detected in horizontal direction as shown in Fig.2 (c). Therefore the border must be detected if the angle formed by cluster border and coordinate axis is not above 45°. Because B takes the value of positive and negative direction in each dimension, the condition must be satisfied. The point d in Fig.2 (d) is not a border point, but the algorithm may regard it as a border point, that is, detect border layer (from the border point c to the inner point d). The thickness of the border layer is p-r.

3 CUBN algorithm

In this section we introduce our new clustering algorithm CUBN. CUBN algorithm consists of 3 algorithms.

3.1 Algorithm description

Algorithm 1 applies the method described in section 2. The following is the details of the algorithm1.

Algorithm1: find border points
Input: the data set S
(n points in d-dimensional space)
p—the value in the matrix B
Output: C—a set of border points
Method:
• $r = p/\sqrt{2}$;
• for each column b in matrix B
• for each point x in S
{ • $k=\#\{ y \mid y \in O(x+b, r) \cap S \}$
• if k=0
• $C \leftarrow C \cup \{x\}$ }

The algorithm CUBN needs the user to give the input parameter p. Parameter p must satisfy the condition discussed in section 2.2, that is, p satisfies the following condition:

$\max(\text{distance}(x_i, x_j)) < p+r < \min(\text{distance}(A_i, A_j))$ in this range, the algorithm can get a good result. In practice, this range is difficult to decide, so we choose a small value (below 0.05) according to our experience.

Algorithm 2 applies the nearest method to cluster border points. The following is the details of the algorithm2.

In algorithm 2, the critical distance e is taken as the value of p+r. If a border point has no near point, it is regarded as an inner point or outlier, which will be disposed in algorithm 3 respectively.

Algorithm2: clustering border points
Input: C—a set of border points
Output: PC—a set of border clusters
Method:
• regard the C as the initial PC, i.e., PC ← C;
• repeat
{ • for each PC_i and PC_j
• if $\exists u \in PC_i, \exists v \in PC_j$ and distance (u,v)≤e
• merge PC_i and PC_j, i.e., $PC_i \leftarrow PC_i \cup PC_j$ }
• until no change

Algorithm 3 also applies the nearest method to cluster inner points. Firstly, it finds the inner point clusters that are nearest to all border points, and then regard these inner points as border points and repeats this operation until all inner points are finished. If some inner points do not belong to any cluster, they are outliers. The following is the details of the algorithm 3.

Algorithm3: clustering inner points
Input: PC—a set of border cluster
Output: RC—a set of clusters
Method:
• Z ←S −C, i.e. Z is the set of all inner points
• RC←∅
• while Z≠∅
{ • I =∅
• for each inner point x
• for each PC_i
• if $\exists y \in PC_i$ and distance (x , y)≤e
{ • $I_i \leftarrow I_i \cup \{x\}$;
• Z←Z−{x} }
• PC ← I
• RC ← RC∪I }

To enhance the cluster quality of CUBN algorithm, there are two improvements in algorithm 3.

• after the first operation, x_i , $x_j \in I$, where I is the inner points cluster, if max (distance (x_i , x_j)) > p+ r , then I is to be split (see experiment 1).

• after the last operation, I_i , I_j is two inner points cluster, if min (distance (I_i , I_j)) < r ,then merge I_i and I_j (see experiment 2).

3.2 Algorithm evaluation

We next examine the time complexity of our clustering algorithm for n input data objects. Suppose there are m border points, then there are n−m inner points. In the following, we discuss the three algorithms' time complexity, respectively.

In algorithm 1, the key calculation is the operation that

calculates the distance between two points and compares the distance with r, which is called a calculation. Each border point needs a calculation with other points, that is, all border points need m×n calculations. Algorithm 1 does the erosion in positive and negative direction of each dimension in d-dimension space. Therefore, the time complexity of algorithm 1 is 2×m×n×d.

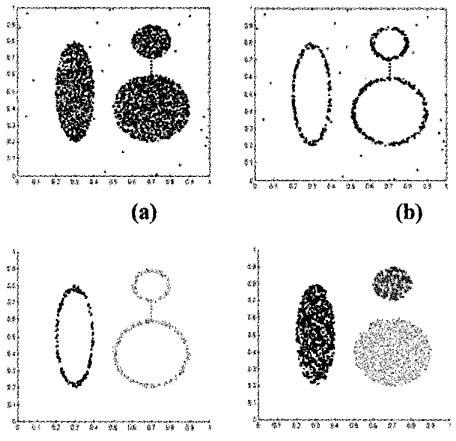
The time complexity of algorithm 2 is independent of n. The time complexity of algorithm 3 is (n−m)×m. Therefore, the time complexity of CUBN is O (nmd).

4 Experimental results

In order to study the performance of CUBN and demonstrate its effectiveness for clustering, we did experiments with two data set containing points in two dimensions.

4.1 Experimental 1

The data set1 is shown in Fig.3 (a). Let n=5000 and p=0.05. Fig.3 (b) shows the result after running algorithm 1, the points in the figure are the border points. Fig.3 (c) shows the result after running algorithm 2. Because some outlier points link the big circle and small circle, all border points are clustered to two clusters. Fig.3 (d) shows the result after running algorithm 3. The big circle and small circle are split.

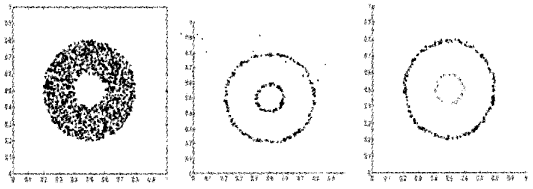


(a) (b) (c) (d)

Figure.3 data and result of experiment 1

4.2 Experiment 2

The data set is shown in Fig.4 (a). Let n=5000 and p=0.05. Fig.4 (b) shows the result after run algorithm 1. Fig.4 (c) shows the result after running algorithm 2, there are two clusters. After run algorithm 3, the two clusters are merged, and all data are in one cluster.



(a) (b) (c)

Figure.4 data and result of experiment 2

5 Conclusions

In this paper, we propose a novel clustering method: CUBN. The following is the characteristic of CUBN.

- Detect clusters having non-spherical shapes and wide variances in size.
- Deal with outliers successfully.
- Insensitive to the order of input data.
- Computing complexity is O(n), which allows CUBN to handle large data sets efficiently.
- Easy to determine the input parameters, do not require user to give the number of clusters.

Acknowledgements:

This work was supported by the National Science Foundation of China (Grant No 60275020).

References

[1] Han J, M. Kambr. Data Mining: Concepts and Techniques. Peking, China: Higher Education Press, 2001

[2] S. Guha, R. Rastogi, K. Shim. CURE: an efficient clustering algorithm for large database. Information Systems. 26(1): 35-58, Jan 2001

[3] T. Zhang, R. Ramakrishnan, M. Livny. Birch: An efficient data clustering method for very large database. In Proceedings of the ACM SIGMOD Conference on Management of Data, Montreal, Canada, 103-114, ACM Press (1996)

[4] M. Ester, H. P. Kriegel, J. Sander, X. Xu.. A density-based algorithm for discovering clusters in large spatial databases. In Proc. 1996 Int. Conf. Knowledge Discovery and Data Mining (KDD'96), 226-231, Portland, OR, Aug, 1996

[5] M. Ankerst, M. Breunig, H. P. Kriegel, J. Sander. OPTICS: Ordering points to identity the clustering

structure. In Proc. 1999 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'99), 49-60, Philadelphia, PA, June 1999.

[6] R. Agrawal, J. Gehrke, D. Gunopulos, P. Raghavan. Automatic sub-space clustering of high dimensional data for data mining applications. In Proc. 1998 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'98), 94-105, Seattle, WA, June 1998.

[7] C. Q. Tang, H. P. Lu, Z. Huang and F. Zhang. Mathematical Morphology and Applications (in Chinese). Peking, China: Science Press, 1990