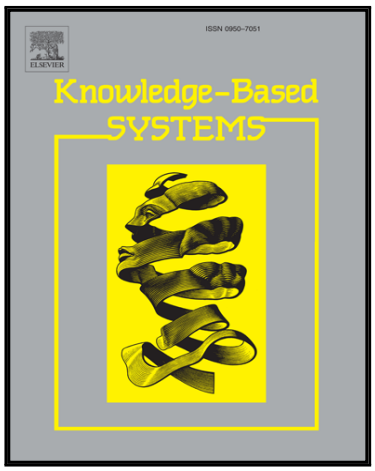

Density Core-based Clustering Algorithm with Dynamic Scanning Radius

Xie Jiang, Zhong-Yang Xiong, Yu-Fang Zhang, Yong Feng, Jie Ma

PII: S0950-7051(17)30557-9
DOI: 10.1016/j.knosys.2017.11.025
Reference: KNOSYS 4120

To appear in: *Knowledge-Based Systems*

Received date: 23 July 2017
Revised date: 14 November 2017
Accepted date: 22 November 2017

Please cite this article as: Xie Jiang, Zhong-Yang Xiong, Yu-Fang Zhang, Yong Feng, Jie Ma, Density Core-based Clustering Algorithm with Dynamic Scanning Radius, *Knowledge-Based Systems* (2017), doi: 10.1016/j.knosys.2017.11.025

Density Core-based Clustering Algorithm with Dynamic Scanning Radius

Jiang Xie, Zhong-Yang Xiong, Yu-Fang Zhang, Yong Feng, Jie Ma

(College of Computer Science, Chongqing University, Chongqing 404100, China)

Abstract

Clustering analysis has been widely used in many fields such as image segmentation, pattern recognition, data analysis, market researches and so on. However, the distribution patterns of clusters are natural and complex in many research areas. In other words, most real data sets are non-spherical or non-elliptical clusters. For example, face images and hand-writing digital images are distributed in manifolds and some biological data sets are distributed in hyper-rectangles. Therefore, it is a great challenge to detect clusters of arbitrary shapes in multi-density datasets. Most of previous clustering algorithms cannot be applied to complex patterns with large variations in density because they only find hyper-elliptical and hyper-spherical clusters through centroid-based clustering approaches or fixed global parameters. This paper presents DCNaN, a clustering algorithm based on the density core and the natural neighbor to recognize complex patterns with large variations in density. Density cores can roughly retain the shape of clusters and natural neighbors are introduced to find dynamic scanning radiuses rather than fixed global parameters. The results of our experiments show that compared to state-of-the-art clustering techniques, our algorithm achieves better clustering quality, accuracy and efficiency, especially in recognizing extremely complex patterns with large variations in density.

Keywords:
Clustering, Density core, Dynamic Scanning Radius, Natural Neighbor

1. Introduction

Nowadays, a large number of data continuously generate from application such as consumer clicks, telephone usage flows, multimedia data mining, computer network intrusion detection and sensor network data clustering. Clustering data has become one of the most important issues since a majority of unlabeled data come in the age of Big Data[1]. Along with the emergence of Big Data, there is an ever-increasing interest in clustering algorithms that can automatically understand, process and summarize the data [9]. Clustering analysis is the process of dividing a dataset into groups or clusters according to their similarities, therefore the objects in different clusters have few similarities while the objects in the same cluster have many similarities. There are many clustering algorithms such as density-based methods[3], partitioning methods[10], hierarchical methods[12][13], grid-based methods, fuzzy clustering methods[15], mean shift clustering methods[14] and combinations of these. Density-based methods and partitioning methods are the most popular two.

As the representative of density-based methods, Density-based Spatial Clustering of Applications with Noise(DBSCAN)[3] is widely used in clustering. It can detect clusters in any arbitrary shapes through a fixed scanning radius eps and a density threshold MinPts[16][18]. However, DBSCAN cannot be applied to high-dimensional data with large variations in density because of the fixed global parameter and the curse of dimensionality. Some clustering algorithms[17][19][20] are introduced to eliminate the limitations of DBSCAN.

Partitioning methods, such as K-Means[4], K-Medoids and K-Center[11], have been widely used to cluster data in Gaussian-like distribution for half a century. Restrictions in the selection of right initial clusters lead that K-means algorithm must run repeatedly to achieve better results. A data point is always assigned to the nearest center, therefore partitioning methods are neither able to detect non-spherical clusters nor robust to outliers and noises.

It is crucial to find clusters of arbitrary shapes in the application of clustering algorithms. For example, in the process of geographic information data, mountains, rivers and other terrains often present various irregular shapes that can be recognized by clustering algorithms. Moreover, in medicinal fields, clustering algorithms can effectively identify irregular spatial structures of biological proteins, which helps to cognize the composition and function of proteins. Therefore, in recent years, a lot of clustering algorithms have been presented for the discovery of arbitrary shapes. Clustering by Fast Search and Find of Density Peaks (DPeak)[8] is based on the assumption that whatever the shape of a cluster is, centers are surrounded by neighbors with lower local densities and these neighbors are at relatively large distance from any points with higher local densities. DPeak first identify cluster centers, then assign the remaining points to their appropriate clusters and detect outliers as well. However, DPeak also has some drawbacks. Firstly, the selection of the cutoff distance dc has great influence on the clustering results; Secondly, as a centroid-based method, DPeak may lead up to shape loss, false distance and false peak when applied to complex patterns [2]. In the elimination of the high dimensional curse, DPC-KNN-PCA[23] combines PCA, DPeak and k-nearest neighbor to avoid

Email address: xiejiang@cqu.edu.cn (Jiang Xie, Zhong-Yang Xiong, Yu-Fang Zhang, Yong Feng, Jie Ma)

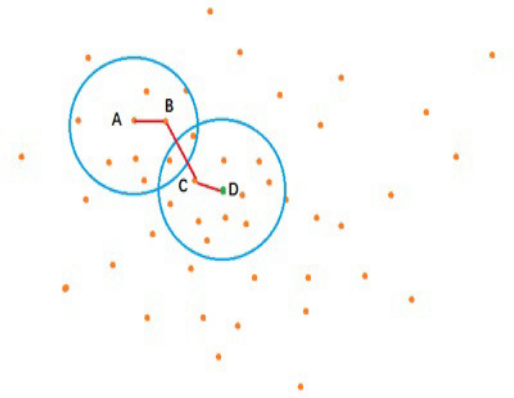

Figure 1: an example of getting a convergent point. Point A is a starting, B is the center of A, C is the center of B, and the green point D is the convergent point of A, i.e., $cp(A) = D$.

the shortcomings of DPeak. However, this center-based algorithm still cannot detect clusters containing mainifold distributions. Density-adaptive Affinity Propagation Clustering algorithm(DAAP)[24] extending form AP[25] algorithm is proposed to recognize complex manifold structures. But high time complexity is the limitation of DAAP. Recently, Density-core-based Clustering (Dcore)[2] has been proposed. It is based on the intuitive assumption that each cluster has a density core excluding outliers, borders and edges. Density core is a shrunken region consisting of a set of loosely connected points with relatively high densities in a cluster. Dcore can abandon over-reliance on a centroid by replacing the centroid with a density-core. Although Dcore can detect complex patterns, it selects non-trivial scanning radiuses and parameters for better results, which is similar to choosing an appropriate dc in DPeak. Furthermore, in the use of fixed global scanning radius, Dcore is still unable to cluster data with extremely large variations in density.

In order to solve problems of density-core based methods, we introduce a novel density core-based clustering algorithm with dynamic scanning radius(DCNaN) in this paper. Our new algorithm defines that the scanning radius of point ρ_i is based on the mean distance between point ρ_i and its natural neighbors, so the deficiency of Dcore, namely the use of the fixed scanning radius, can be avoided. In order to detect background noises and outliers, we also introduce a novel technique based on the level partition of scanning radiuses. Firstly, DCNaN uses different radiuses to get the convergent point of each point ρ_i and find the local density peaks whose convergent points are themselves. Secondly, we divide local density peaks into different groups called density cores by dynamic scanning radiuses. Each density core represents a predicted cluster. Lastly, each non-density peak point is assigned to a cluster containing its convergent point. Without setting parameters, DCNaN can be applied to complex patterns with extremely large variations in density and it is robust to outliers and noises.

The rest of this paper is organized as follows. Firstly, after a review of related works in Section 2, the clustering algorithm DCNaN is presented in Section 3. Secondly, the time and space complexity is discussed in Section 4; the processes and results of experiments on synthetic and real datasets are described in section 5. Finally, a summary of this paper and expectation of future researches are presented in Section 6.

2. Related works

2.1. Density core-based algorithm and its analysis

Dcore[2], a hybrid decentralized approach by means of k-center[11], NNC[22] and mean shift[14] at different levels,was published on *Information Science*. The assumption of Dcore is that each cluster is considered to have a density core instead of a centroid and the borders and edges are distributed in a hierarchical structure outside of the core. Dcore uses mean shift and k-center to find convergent points. For example, in Fig.1, we choose point A and shift it from a low-density area to a local density center via mean shift. The local density center is a point that is nearest to the average in each iteration. The path from A to D is a shifting track, which is denoted by $cp(A) = D$. All shifting tracks to the convergent points form shifting streams that can retain the information of the cluster shape. A point is assigned to its cluster according to its center point and convergent point instead of the direct distance between the point and its cluster. Local density peak is a point that its convergent point is itself.

Different density cores represent different predicted clusters. The ith density core is labeled as the predicted cluster π_i and $cl(\rho_i) = \pi_i$ indicates that point ρ_i is assigned to cluster π_i. Dcore assigns the none-convergent point ρ_j to the predicted cluster containing its convergent point $\mathrm{cp}(\rho_j)$.

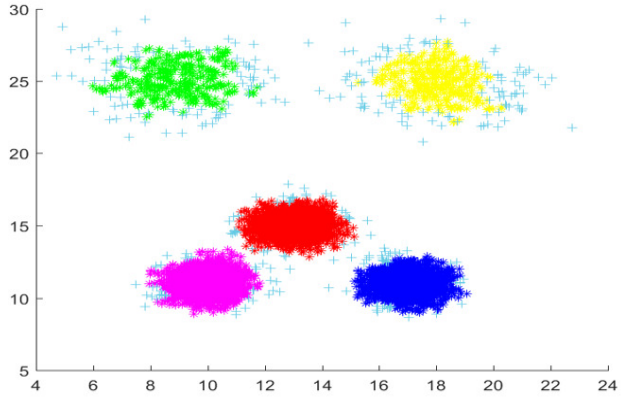

Figure 2: Five clusters with large variations in density. The densities of two clusters on the top are sparse, while the densities of other three clusters are dense.

Dcore is a hybrid method that utilizes centroid-based, non-centroid-based, hierarchical and partitional methods. It decentralizes each density peak into a loose density core, which can avoid over-reliance on centroids. The results of Dcore in different datasets indicate that it can perform well in many complex patterns. But here are still some apparent limitations:

(i) Due to the use of the fixed global scanning radius, it is difficult to find the most suitable radius in the search of the convergent point. Furthermore, density core is a group of local density peaks connected with each other based on the fixed scanning radius r. As shown in Fig.2, whether the value of radius r is big or small, the cluster results of datasets are affected, especially for datasets with extremely large variations in density.

(ii) Dcore uses the false local density peak to detect outliers and noises and provides three strategies to deal with the false local density peak in different situations. However, it is difficult to decide what kind of situation it is.

2.2. Natural Neighbor

Natural neighbor[6][7] is a new concept that comes from objective reality. The idea is based on that the number of one's true friends should be the number of people taking him/her as a friend and considered by him/her as friends at the same time. If everyone has at least one friend, it will form a relatively stable structure called Natural Stable Structure(NSS) in the society. For data points, if point ρ_j considers ρ_i as a neighbor and ρ_i considers ρ_j as a neighbor at the same time, ρ_i is one of the natural neighbors of point ρ_j. Natural Stable Structure for data points can be formulated as follows:

$$(\forall\rho_i)(\exists\rho_j)(k \in N) \wedge (\rho_i \neq \rho_j) \longrightarrow (\rho_i \in NN_k(\rho_j)) \wedge (\rho_j \in NN_k(\rho_i)) \tag{1}$$

Where $NN_k(\rho)$ is the kth-neighborhood of ρ. The searching round k increases from 1 to λ. λ is named Natural Neighbor Eigenvalue(NaNE)[26]. The value size of λ indicates the complexity of data object distribution. If the distribution of data objects is regular or the size of a dataset is small, the value of λ is small. If the distribution of data objects is irregular or the size of a dataset is big, the value of λ is big. Nevertheless, no matter how big the value of λ is, it is still far smaller than the value of dataset size.

In order to find the natural neighbor of each point, we continuously expand the neighbor searching range and then compute each point's neighbors that are also considered as other points' neighbors until every point is considered as a neighbor or the number of points that are not considered as neighbors no longer changes. The natural neighbor of point ρ_i is defined as follows, and details of this method are presented in algorithm 1.

$$\rho_j \in NaN(\rho_i) \Leftrightarrow \rho_i \in NN_\lambda(\rho_j) \wedge \rho_j \in NN_\lambda(\rho_i) \tag{2}$$

In algorithm 1, k is the searching range and $Nb(\rho_i)$ indicates the times that point ρ_i is considered as the neighbor of other points. $NN_k(\rho_i)$ is the k-th neighbor of ρ_i. $RNN_k(\rho_j)$ is the k-th reverse neighbor of ρ_j. If every point has at least one point considering it as a neighbor, the dataset will form a relatively stable structure. However, it is unable to find the natural neighbor of noise points in a dataset because there are no points considering them as neighbors. Therefore, the stopping criterion of iteration is that the number of points with no neighbor (i.e. $Nb(\rho_i) = 0$) no longer change.

3. DCNaN algorithm

According to algorithm 1, there is no need to set any parameters in the process of natural neighbor searching. Based on the density core and the natural neighbor, we propose a density core-based algorithm with dynamic scanning radius(DCNaN). Density cores can roughly retain the shape of a pattern and natural neighbors can find dynamic scanning radiuses. We also introduce a method based on the level partition of scanning radiuses to detect outliers and noises. Without parameter setting, DCNaN can recognize extremely complex patterns with large variations in density and it is robust to outliers or noises.

Algorithm 1 Natural Neighbor Search algorithm(NaN-Search)

Input: D (the dataset)
Output: λ, NaN, Nb

```
Initialize: k=1, Nb(ρ_i) = 0, NN_k(ρ_i) = Ø, RNN_k(ρ_i)=Ø, NaN(ρ_i) = Ø;
create a KD-tree T from dataset D;
while true do
    for each point ρ_i in D do
        find the k-th neighbor ρ_j of ρ_i using T
        Nb(ρ_j) = Nb(ρ_j) + 1;
        NN_k(ρ_i) = NN_{k-1}(ρ_i) ⋃ {ρ_j};
        RNN_k(ρ_j) = RNN_{k-1}(ρ_j) ⋃ {ρ_i};
    end for
    Find the number of point that Nb(ρ_i)==0, denoted by Num;
    if the Num does not change any more then
        Break;
    end if
    k = k + 1;
end while
λ = k
for each point ρ_i in D do
    NaN(ρ_i) = RNN_λ(ρ_i) ⋂ NN_λ(ρ_i);
end for
return λ, NaN, Nb
```

3.1. Dynamic scanning radius

Definition 1. *(The sum of distances between ρ_i and $NaN(i)$). The sum of distances between point ρ_i and its natural neighbors is defined as:*

$$d_{sum}(i) = \sum_{j \in NaN(i)} dist(i,j) \tag{3}$$

Definition 2. *(Dynamic scanning radius). The dynamic scanning radius of point ρ_i is the mean distance between point ρ_i and its natural neighbors. According to definition 1, the dynamic scanning radius of point ρ_i is defined as:*

$$r(i) = d_{sum}(i)/b(i) \tag{4}$$

$b(i)$ is the number of point ρ_i's natural neighbors. Each point ρ_i has a different number of natural neighbors, and the distance between ρ_i and its NaN are different. For example, compared with a data point in a dense region, a data point in a sparse region has fewer neighbors and longer distances from its neighbors. So the dynamic scanning radius of point ρ_i in a sparse region is longer than that of point ρ_j in a dense region. Unlike definitions in[21], there is no need to set parameters because the dynamic scanning radius is based on the theory of natural neighbor.

3.2. The level partition of scanning radiuses

Definition 3. *(Scanning Radius Variation). The Scanning Radius Variation of point ρ_i with respect to ρ_j, denoted by RVar (ρ_i, ρ_j), is defined that how denser or sparser ρ_i than ρ_j. It is computed as follows:*

$$RVar(\rho_i, \rho_j) = \frac{|r(\rho_i) - r(\rho_j)|}{r(\rho_i)} \tag{5}$$

Dynamic scanning radiuses are sorted in ascending order. Firstly, we make a list of sorted scanning radiuses r (denoted by $rList$) and assume the size of the list is n; secondly, we get $RVarList$ (denoted by $RVarList$) of size n-1 by computing $RVar(\rho_i, \rho_i)$ for each sequence-adjacent $r(i)$ and $r(i+1)$ in rList. The plot of $RVarList$ is shown in Fig.3. It is important to note that each element in $RVarList$ is corresponding to two data points in a dataset.

In $RVarList$ plot, there are sharp waves and relatively smooth lines. Sharp waves indicate that the corresponding points are outliers. So there is a partition method that we split $RVarList$ plot into two parts by a straight line, and the part higher than the straight line can be treated as outliers or noises. $RVarList$ plot is unnecessary according to this theory. Then we can get a threshold value β and move points whose $RVar$ value are bigger than value β out of $RVarList$. Based on statistical characteristics, the threshold value is defined as:

$$\beta = EX(RVarList) + \omega \times SD(RVarList) \tag{6}$$

EX is a mathematical expectation, SD is a standard deviation and ω is a tuning coefficient. As shown in Fig.3, sharp waves fluctuate slightly around $RVarList's\ EX$, and the width proportion of sharp waves is far less than that of smooth lines.

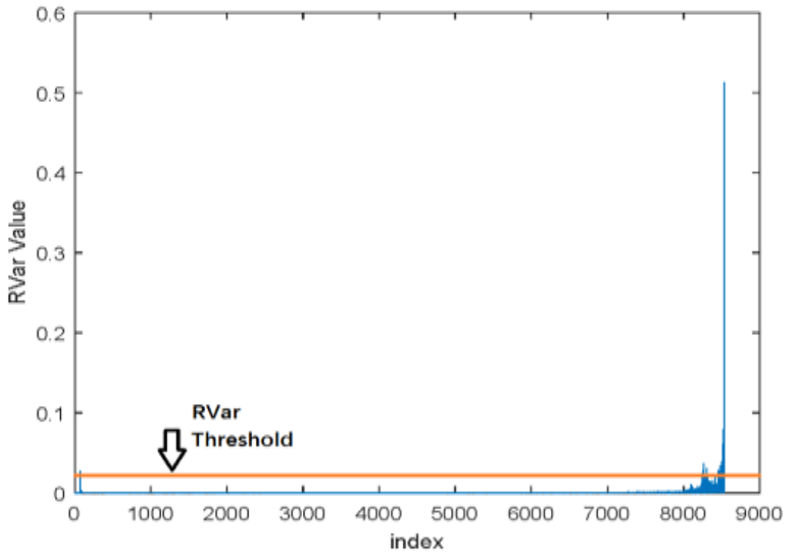

Figure 3: In RVarList plot, the smooth lines symbolize density level and the sharp waves indicate sharp changes.

Therefore, we add a positive adjustment in EX to detect outliers and noises. ω can be chosen from the range (0,3]. According to a large number of experiments, $\omega = 2.5$ is an ideal value[27] for most datasets. We determine the value of ω to be 2.5 in the aim of avoiding parameter setting.

Definition 4. *(Natural Outlier). If $Nb(\rho_i)$ remains zero till the end of algorithm1, the point that remains $Nb(\rho_i)$=0 is called Natural Outlier.*

Definition 5. *(Outlier). If RVar value of point ρ_i is above threshold β or $Nb(\rho_i)$ is equal to zero, this point is an outlier, which is denoted by $CL(\rho_i) = -1$.*

Algorithm 2 Find the convergent point(CP-Search)

Input: D (the dataset), NaN
Output: Center point vector CPV; Convergent point set CP; Dynamic scanning radius r

```
create a KD-tree T from dataset D;
For each point ρ_i, calculate the dynamic scanning radius r(i);
for each point ρ_i in D using T do
    Find the neighbors Nei(ρ_i) = {ρ_i | d_{i,j} < r(i) };
    CPV(ρ_i) = the point nearest to the center within Nei(ρ_i) ⋃ {ρ_i};
end for
for each point ρ_i do
    CP(ρ_i)=ρ_i
    while (CP(ρ_i) ≠ CPV(CP(ρ_i)) && not loop found) do
        CP(ρ_i)=CPV(CP(ρ_i))
    end while
end for
return CPV, CP, r
```

3.3. Density core

Definition 6. *(Density of point ρ_i with its $r(i)$-neighborhood) The density of point ρ_i with its $r(i)$-neighborhood is defined as:*

$$\rho_{i,r} = \sum_{k=1:N} l(d_{i,k} \leqq r(i)) \quad where \quad l(x) = \begin{cases} 0 & x = false \\ 1 & x = true \end{cases} \tag{7}$$

where $\rho_{i,r}$ represents the density of point ρ_i with its $r(i)$-neighborhood. N is the number of points and $d_{i,k}$is the distance between ρ_i and ρ_k. The dynamic scanning radius $r(i)$ is changing with the density of point ρ_i. As shown in Fig.4 and Algorithm2, mean shift and k-center algorithms are used to find the convergent points, which is denoted by $CP(\rho_i)$=ρ_j.

Algorithm 3 Find local density peak set(LPS-find)

Input: CP; CL
Output: Local density peak set (LPS)

```
Initialize the local density peak set LPS=Null;
for each point ρ_i do
    if (ρ_i == CP(ρ_i) && CL(ρ_i) ≠ -1) then
        LPS=LPS ⋃ {ρ_i};
    end if
end for
return LPS
```

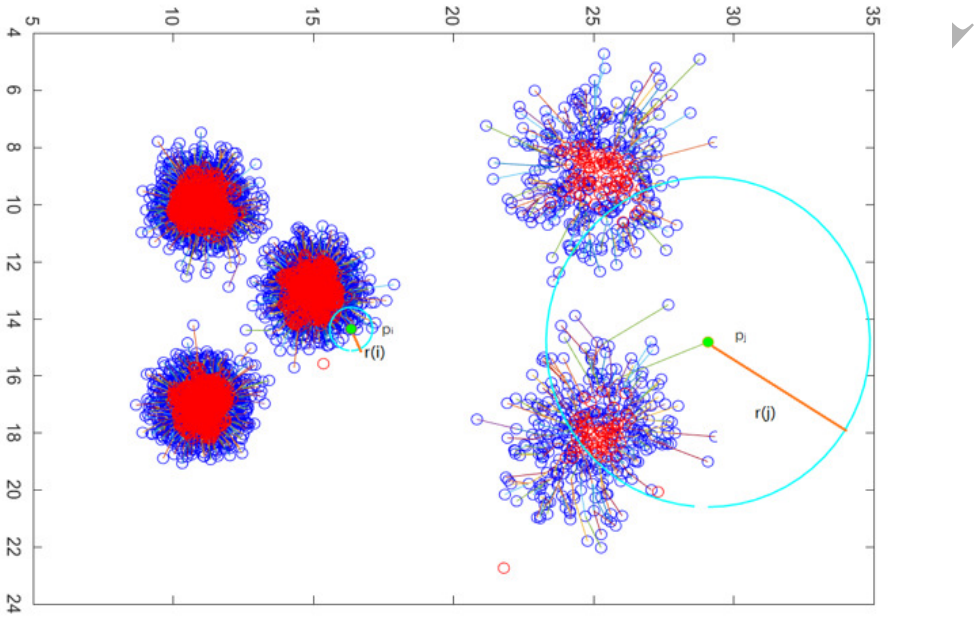

Figure 4: Each line shows the shift direction of a point toward its convergent point. The red regions represent density cores, and each point in the red regions is a local density peak. $r(i)$ and $r(j)$ represent the different scanning radiuses for point ρ_i and ρ_j in different densities.

Definition 7. *(Local density peak) If $cp(\rho_i) = \rho_i$ and ρ_i is neither an outlier nor a noise, point ρ_i is a local density peak.*

Local denstiy peak set(LPS) contains all density peaks. Algorithm 3 presents the process of finding a local density peak set.

Definition 8. *(Density Core) The definition of density core is as follows:*

$$CORE = \{\rho | \exists \rho_i \in (LPS)\ s.t.\ d(\rho, \rho_i) < r\ \&\&\ \forall \rho_j \in CORE'\ s.t.\ d(\rho, \rho_j) > r\} \tag{8}$$

The definition of density core in formula (8) means that a core that is far from other cores is a set consisting of connected local density peaks. As show in Fig.4, the circle represents a different point, the red regions represent density cores, and each point in the red regions is a local density peak. Each line shows the shift direction of a point toward its convergent point. The red points are local density peaks, while the blue points are non-density peaks. The greater the density of point ρ_i is, the smaller the radius is. Due to the use of dynamic scanning radiuses, we can clearly see that the scanning radius $r(i)$ of the green point ρ_i is smaller than the scanning radius $r(j)$ of the green point ρ_j. Algorithm 4 presents the process of finding density cores which can roughly retain the pattern shape. Each density core represents a predicted cluster.

3.4. Density Core-based Clustering Algorithm with Dynamic Scanning Radius(DCNaN)

We introduce a novel clustering algorithm, namely Density Core-based Clustering Algorithm with Dynamic Scanning Radius(DCNaN). The basic steps are that: firstly, we find natural neighbors and compute the dynamic scanning radius of each point; secondly, we compute the scanning radius variation list $RVarList$ and threshold β to detect outliers and noises; thirdly, we get a set of local density peaks; fourthly, we find density cores representing different predicted clusters; lastly, each point ρ_i that does not belong to LPS is marked as the $ClusterID$ of its convergent point. As shown in Algorithm 5, details of DCNaN are described below.

Algorithm 4 Find the Density Cores(Core-find)

Input: Local density peak set LPS; Dynamic scanning radius r; CL
Output: Cluster Label CL;

for each point ρ_i in LPS **do**
  if $CL(\rho_i)$==0 **then**
    $ClusterID$=max(CL)+1;
    $NeiCP$= $\{\rho_k \mid d_{i,k} < r(i)\}$;
    $CL(\rho_i)$=$ClusterID$;
    while ($\exists$ $\rho_j \in NeiCP$) **do**
      if $CL(\rho_j)$==0 **then**
        $CL(\rho_j)$=$ClusterID$;
        $NeiCP$=$NeiCP \bigcup \{ \rho_n \mid d_{j,n} < r(j)\}$
      end if
    end while
  end if
end for
return CL

Algorithm 5 DCNaN

Input: D (the dataset);
Output: Cluster Label CL;

Initialize the CL=0;
[λ,NaN,Nb]=NaN-Search(D);
[CPV,CP,r]=CP-Search (D, NaN);
Compute the scanning radius variation list
Compute the threshold β based on $\beta = EX(RVarList) + 2.5 \times SD(RVarList)$;
for each point ρ_i in $RVarList$ **do**
  if $RVar(\rho_i, \rho_j) > \beta$ **then**
    $CL(\rho_j)$=-1;
  end if
  if $Nb(\rho_i)$==0 **then**
    $CL(\rho_i)$=-1;
  end if
end for
[LPS]=LPS-find (CP, CL);
[CL]=Core-find (LPS, r, CL);
for each point $\rho_i \notin LPS$ **do**
  if $CL(\rho_i) \neq -1$ **then**
    Assign $CL(\rho_i)$=$CL(CP(\rho_i))$
  end if
end for
return CL

4. Complexity analysis

The time complexity of DCNaN depends on following parts: (a) in the use of KD-tree, the time for searching natural neighbors is $O(n * \log n)$, and the number of datasets is n. (b) $O(n * \log n)$ is needed in the process of finding convergent points and dynamic scanning radiuses. (c) There are three steps to detect outliers: getting the radius variation, computing the threshold, and determining whether the point is an outlier or not. The time complexity of these three steps is $O(n)$, $O(1)$ and $O(n)$. (d) The complexity of the fourth step namely finding local density peaks is $O(n)$. (e) In this part, we have to find density cores, and its time complexity is $O(n_{lps}^2)$. (f) In the last part, a point that does not belong to LPS is assigned to its cluster and its time complexity is $O(n_1)$, n_1=$n - n_{lps}$.

n_{lps} is the size of local density peak set and ($n_{lps} \ll n$), and the above analysis for DCNaN reveals that the main cost of computation is searching natural neighbors and finding convergent points. Therefore, the total time complexity of DCNaN is $O(n * \log n)$.

5. Experiment analysis

5.1. Evaluation methods

Cluster validity indexes have been used to evaluate the fitness of partitions produced by clustering algorithms[39]. Generally speaking, there are two type of clustering validity indexes: external criteria and internal criteria. In recent years, a number of novel external and internal criteria have been proposed. For example, the Partition Coefficient and Exponential Separation (PCAES) index[39] is proposed to evaluate fuzzy clustering, and the Conn_index[38] based on inter and intra cluster connectives of prototypes is proposed for Prototype-Based Clustering. The real datasets used in our experiments have class labels, so we firstly use external criteria to evaluate clustering results. In the real world, it is false to assume that the cluster matches the class. Therefore, we use the internal index (DBI) to further validate the performance of DCNaN in the real world. In conclusion, we evaluate clustering results from two different perspectives: external criteria and internal criteria.

As the most popular external evaluation criterion, Accuracy[28] is our first choice to evaluate the performance of clustering algorithms. The formula of Accuracy is as follows:

$$AC(\rho, t) = \frac{\sum_{i=1}^{n} \delta(t_i, map(\rho_i))}{n} \quad where \quad \delta(a, b) = \begin{cases} 1, & if \quad a == b \\ 0, & otherswise \end{cases} \tag{9}$$

Where $map(\rho_i)$ is a mapping function that matches the predicted cluster and its real cluster label. $AC \in [0, 1]$, in other words, the bigger the AC value is, the better the clustering is. The upper bound of the AC value is 1.

Normalized Mutual Information[29], the second evaluation criterion, is defined as:

$$NMI(A, B) = 2 * I(A, B)/(H(A) + H(B)) \tag{10}$$

Where $I(A, B)$ is the mutual information between A and B, H is the entropy for random variable, and NMI is in the range of [0,1].

The third evaluation criterion is F-Measure, and the definition of F-Measure is as follows:

$$F = (2 * P * R)/(P + R) \tag{11}$$

Where P represents Precision and R represents Recall. F-Measure also ranges from 0 to 1.

The last evaluation criterion is the internal criterion Davies-Bouldin index[38]. The formula of Davies-Bouldin index is as follows:

$$DBI = \frac{1}{k} \sum_{i=1}^{k} \max_{i \neq j} \left(\frac{\overline{C_i} + \overline{C_j}}{\|w_i - w_j\|_2} \right) \tag{12}$$

$\overline{C_i}$ represents the average distance between each point and the centroid in the ith cluster.$\|w_i - w_j\|_2$ represents the Euclidean distance between the centroid of the i-th cluster and that of the j-th cluster. The smaller the DBI is, the smaller the intra cluster distances are and the bigger the distances between clusters are. In other words, the smaller the DBI value is, the better the clustering is.

Table 1: the characteristics of UCI datasets

Real data	Number of instances	Number of attributes	Number of clusters	Source
Iris	150	4	3	[30]
Wine	178	13	3	[30]
Breast	699	10	2	[30]
BankNote	1372	4	2	[30]
Control	600	60	6	[30]
Pageblock	5473	10	5	[30]
Inonsphere	7474	34	2	[30]

Table 2: the characteristics of synthetic datasets

Real data	Number of instances	Number of attributes	Number of clusters	Source
DS1	6700	2	5	DCNaN
DS2	4599	2	3	DCNaN
DS3	788	2	5	[34]
DS4	1431	2	4	[32]
DS5	8000	2	6	[33]
DS6	8537	2	7	[22]

5.2. Data preprocessing

In our experiments, real-world datasets come from the UCI machine learning repository[30] and the Olivetti face dataset[31], while four synthetic datasets come from researches published in [22][32][33][34]. Table 1 describes the characteristics of UCI datasets and Table 2 displays that of the six synthetic datasets.

Data preprocessing and reduction have become essential techniques in current knowledge discovery scenarios. These methods aim at reducing the complexity inherent to real-world datasets which can be easily processed by current data mining solutions[35]. In data preprocessing phase of DCNaN, data normalization is achieved by min-max normalization which is defined as follows:

$$X_{norm} = (X - X_{min})/(X_{max} - X_{min}) \tag{13}$$

Where X_{norm} is the value of point ρ_i. X_{max} is the maximum value and X_{min} is the minimum value of each corresponding attribute. Before the data is normalized, we replace the missing attribute value with the mean value of the attribute set.

In the data preprocessing of the Olivetti face dataset[31], first of all, we get the gray-level value from each image to construct a 112×92 matrix $S_{112\times92}$. Secondly, we transform S into a feature vector s_i with $112 \times 92 = 10304$ dimensions. Thirdly, we combine the first 100 image feature vectors into a 100×10304 matrix $D_{100\times10304}$. At last, we perform PCA (Principal Component Analysis)[36]on the matrix $D_{100\times10304}$ by preserving its 99.95% accumulation variance to reduce the dimension of matrix $D_{100\times10304}$ and get the result matrix $D_{100\times98}$.

5.3. Experiments on synthetic datasets

We demonstrate the novel clustering algorithm DCNaN has better performance by comparing it with K-means[11], DBSCAN[3], DPeak[8] and Dcore[2]. This section firstly shows the experiment results of the clustering algorithms in six complex synthetic datasets which are embedded in two-dimensional space. The six synthetic datasets are illustrated in Fig.5 and the clustering results of five algorithms are showed in Fig:6-11.

Table 3: the parameter setting of each clustering algorithm in synthetic datasets

Synthetic data	K-means	DBSCAN	DP	Dcore	DCNaN
DS1	N=5	Eps=1 Minpts=5	dc=6%	r1=0.5 r2=0.45 R=0.5 T1=50 Tn=15	-
DS2	N=3	Eps=0.05 Minpts=6	dc=6%	r1=0.08 r2=0.07 R=0.08 T1=30 Tn=10	-
DS3	N=5	Eps=2 Minpts=4	dc=6%	r1=5 r2=4 R=6.5 T1=30 Tn=8	-
DS4	N=4	Eps=0.25 Minpts=4	dc=6%	r1=0.35 r2=0.3 R=0.5 T1=10 Tn=4	-
DS5	N=6	Eps=6 Minpts=5	dc=6%	r1=15 r2=14 R=15 T1=40 Tn=10	-
DS6	N=7	Eps=2 Minpts=10	dc=6%	r1=1 r2=0.9 R=2 T1=30 Tn=10	-

Table 3 displays the parameter setting of each clustering algorithm in the six synthetic datasets. In K-means, the initial cluster number K is set as the desired cluster number and the initial cluster centers are randomly generated. DBSCAN needs two parameters: Eps and Minpts. The cutoff distance dc of DPeak is set as 6% and the density is computed by the exponential kernel suggested by DPeak. The results of Dcore are affected by the selection of parameters r1, r2, T1 and Tn. In order to achieve better results, we try to use different parameter settings. The symbol - in Table 3 means that there is no need to set parameters in DCNaN.

Fig.6 shows that only DCNaN can find correct clusters and detect noises at the same time in the DS1 dataset. Because the chosen Eps is too large, DBSCAN aggregates DS1 into three classes. In Fig.6 (c) and (d), the clustering results of DPeak and that of Dcore are similar. Due to improper choices of parameters, DPeak finds four clusters while Dcore treats noise points as a cluster. DS1 is a dataset with large variations in density, so the experiment results in Fig.6 reveal that algorithms using global fixed parameters cannot be applied to multi-density environments.

The clustering displayed in Fig.7 demonstrates the results that whether those algorithms can process simple manifold dataset or not. The results of DBSCAN and that of DPeak are similar, but none of them finds the right clusters, neither does K-means. Dcore and DCNaN both have found three clusters, while Dcore mistakenly assigns some yellow points into the maroon clusters. In a word, without the density-core structure, K-means, DBSCAN and DPeak cannot roughly retain the pattern shape so that they cannot process manifold datasets.

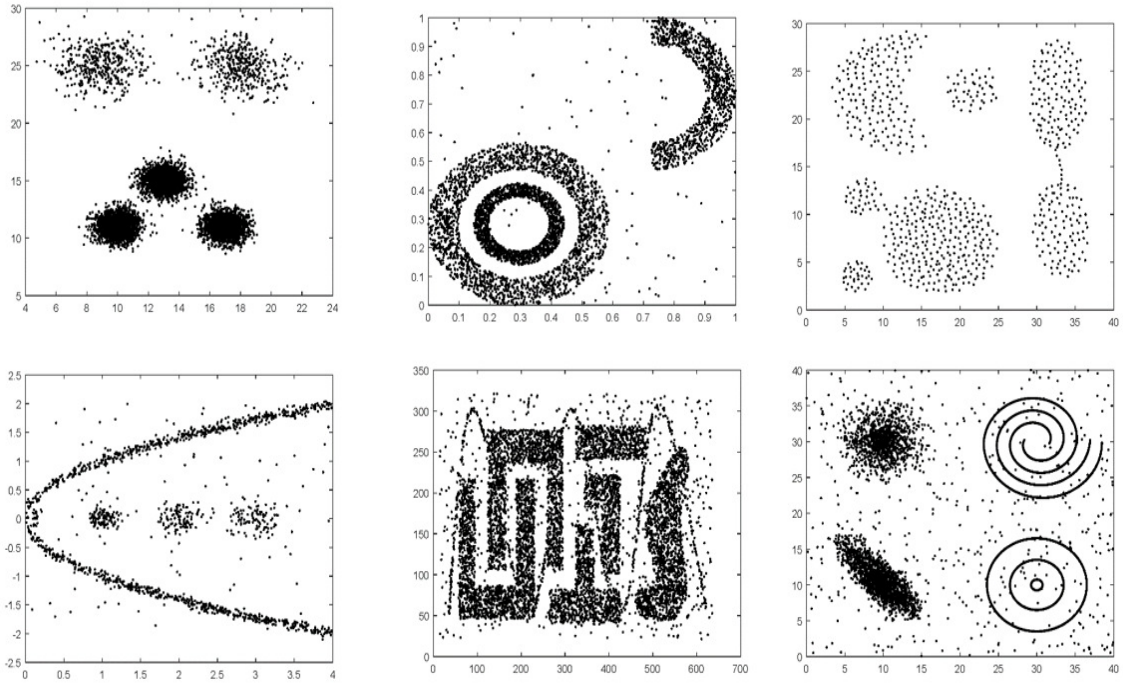

Figure 5: Six original synthetic datasets: from left to right and then from top to bottom, they are dataset 1, dataset 2, dataset 3, dataset 4, dataset 5 and dataset 6.

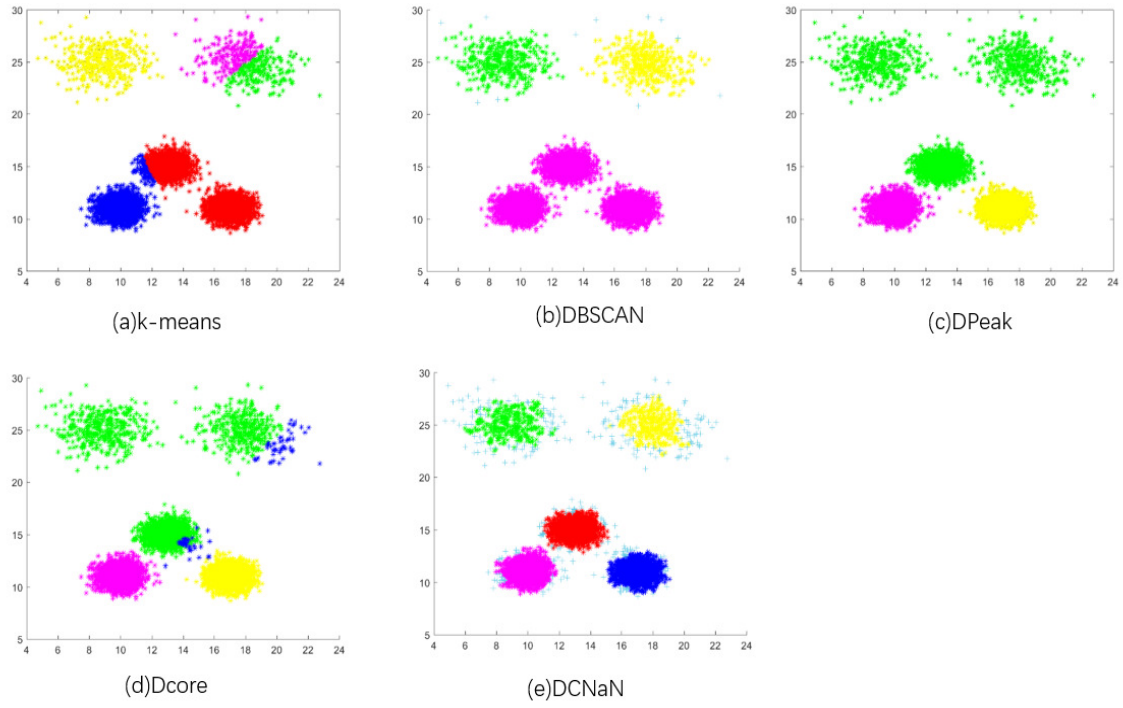

Figure 6: the results of 5 clustering algorithms in DS1:(a) K-means, (b) DBSCAN, (c) DPeak, (d) Dcore and (e)DCNaN

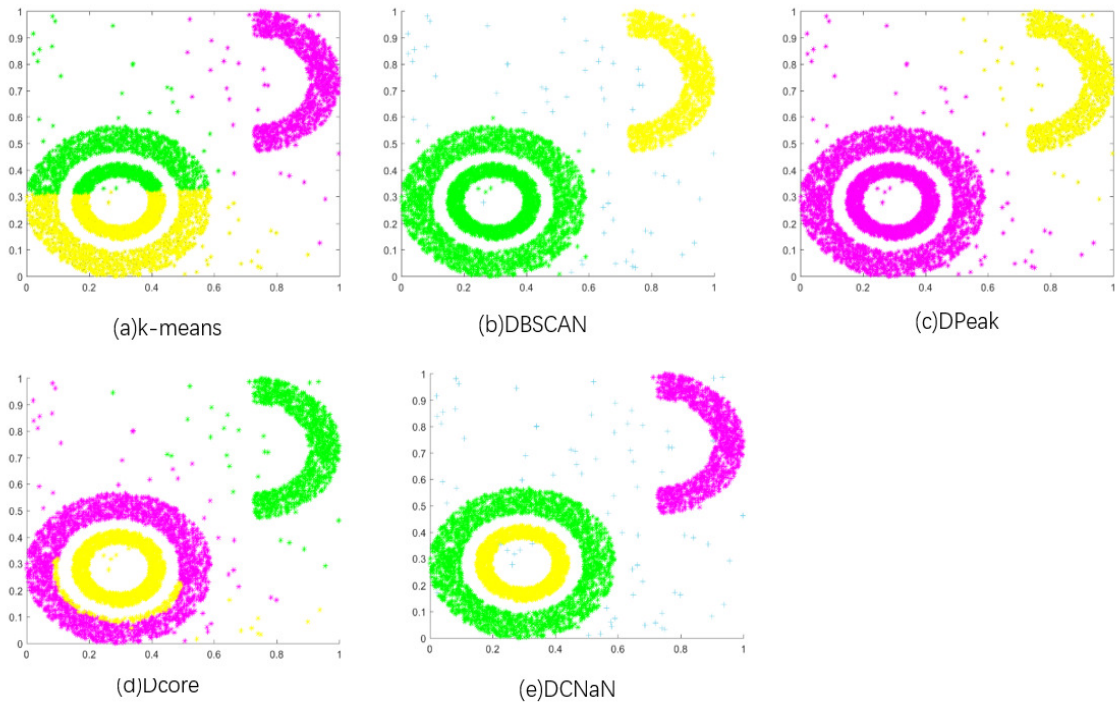

Figure 7: the results of 5 clustering algorithms in DS2:(a) K-means, (b) DBSCAN, (c) DPeak, (d) Dcore and (e)DCNaN

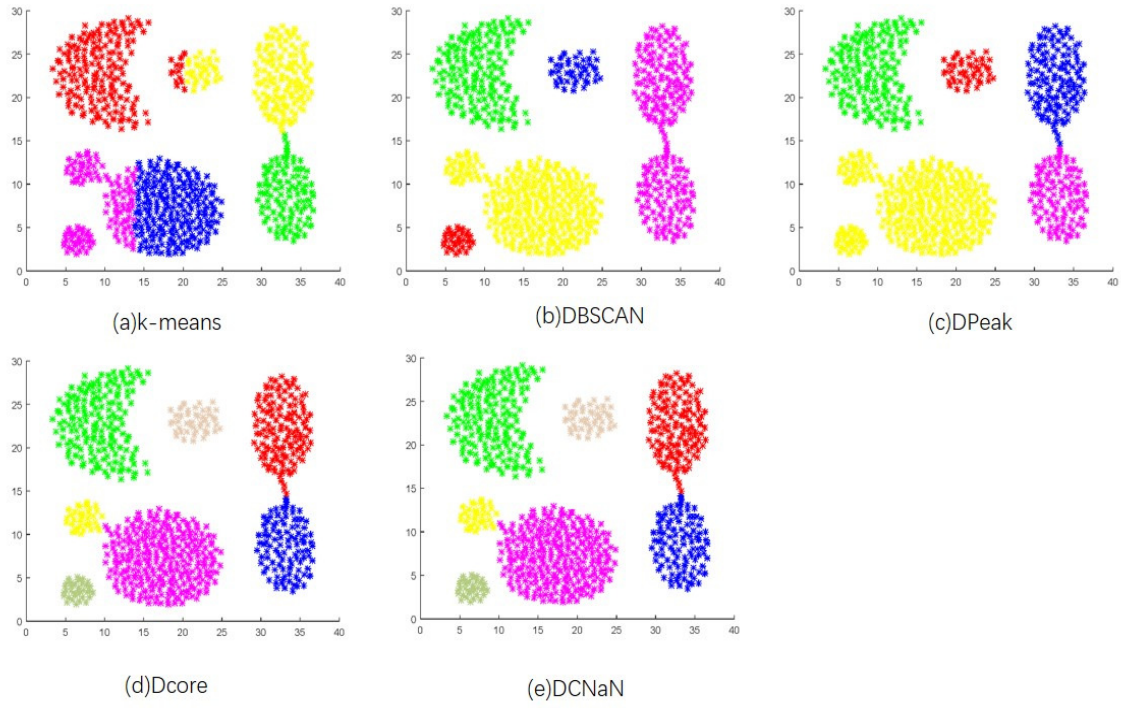

Figure 8: the results of 5 clustering algorithms in DS3:(a) K-means, (b) DBSCAN, (c) DPeak, (d) Dcore and (e)DCNaN

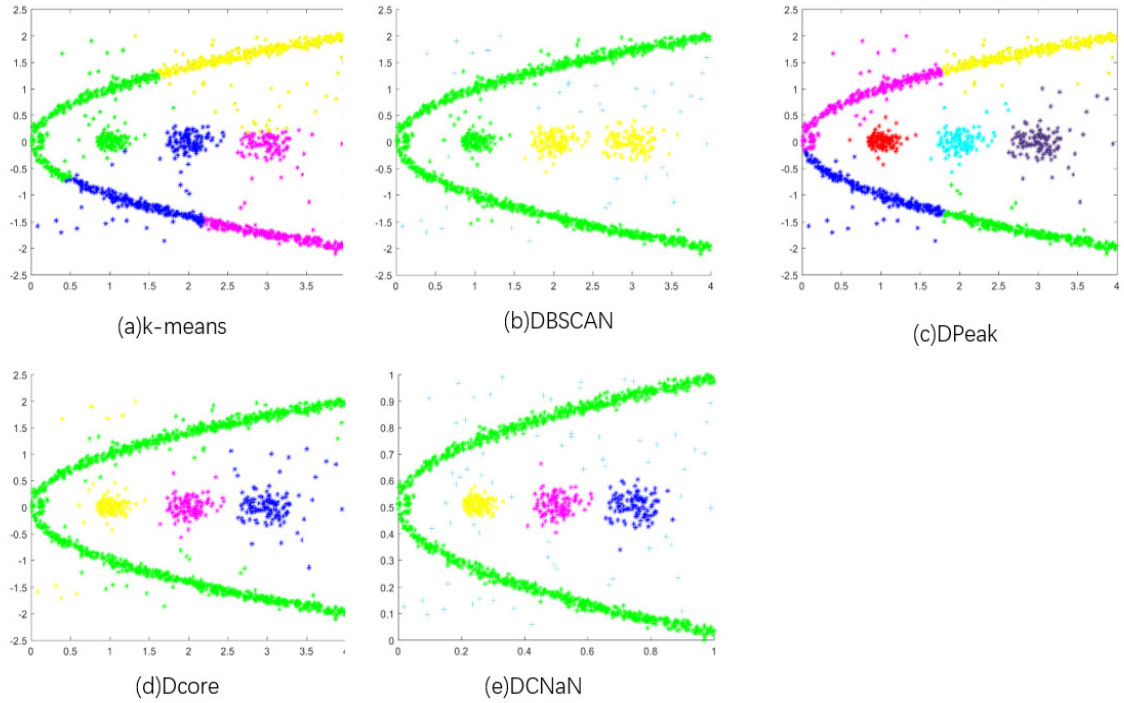

Figure 9: the results of 5 clustering algorithms in DS4:(a) K-means, (b) DBSCAN, (c) DPeak, (d) Dcore and (e)DCNaN

The clustering shown in Fig.8 reveals that Dcore and DCNaN have recognized the clusters in DS3 dataset while K-means, DBSCAN and DPeak have not. In fact, Dcore works better in a single density environment than it does in a multi-density environment.

The clustering shown in Fig.9 tells that both DCNaN and Dcore are good at dealing with manifold structure datasets. K-Means and DPeak can find the correct clusters in a spherical, however, they are unable to discover the clusters in a manifold. In Fig.9 (b), DBSCAN detects partial noises but incorrectly divides DS4 into two clusters.

The clustering shown in Fig.10 demonstrates that only DCNaN and Dcore can detect rectangle clusters. Because the radius R is too large, Dcore assigns partial blue points into wrong clusters. In single density dataset DS5, DCNaN and DBSCAN are robust to noises. It is clear that K-Means, DPeak and Dcore are unable to detect noises. Since clustering centers are manually selected according to the decision graph, the clustering performance of DPeak is unstable.

Experiment results displayed in Fig.11 is to evaluate the clustering performance on more complex patterns. Dataset DS6 is composed of noises and seven clusters that differ in size, density and shape. Meanwhile, the fact that three circles and two spirals tangled with each other makes cluster detection more challenging. As shown in Fig.11 (e), DCNaN correctly finds the noises represented by the symbol + and seven main clusters. Dcore detect two simple shapes and three circles, but fails to find the two spirals. DPeak and DBSCAN both find the rightmost two simple shapes but neither recognize circles and spirals. The reason for the failure of K-Means, DBSCAN and DPeak lies in their over-reliance on centroids, while the cause of Dcore's failure lies in the global fixed radius. Therefore, DCNaN applies dynamic scanning radiuses instead of manual setting to find correct density cores that can be easily distinguished from other cores and roughly retain the shape of different patterns.

Form Fig.6 to Fig.11, we can see that DCNaN performs better than other algorithms. Furthermore, DCNaN can automatically detect noises and outliers through the level partitioning of scanning radiuses.

5.4. Experiments on real datasets in UCI

In this section, we test the performance of DCNaN by using several benchmarking real datasets in UCI. Table 1 illustrates the characteristics of UCI datasets. The performance of K-means, DBSCAN, DPeak, Dcore and DCNaN is benchmarked in terms of AC, NMI,F-Measure and DBI. Table 4 displays the parameter setting of each clustering algorithm in seven real datasets of UCI. Table 5 compares the scores of AC, F-Measure and NMI, and Table 6 compares that of DBI. Table 7 illustrates the consuming time of these algorithms.

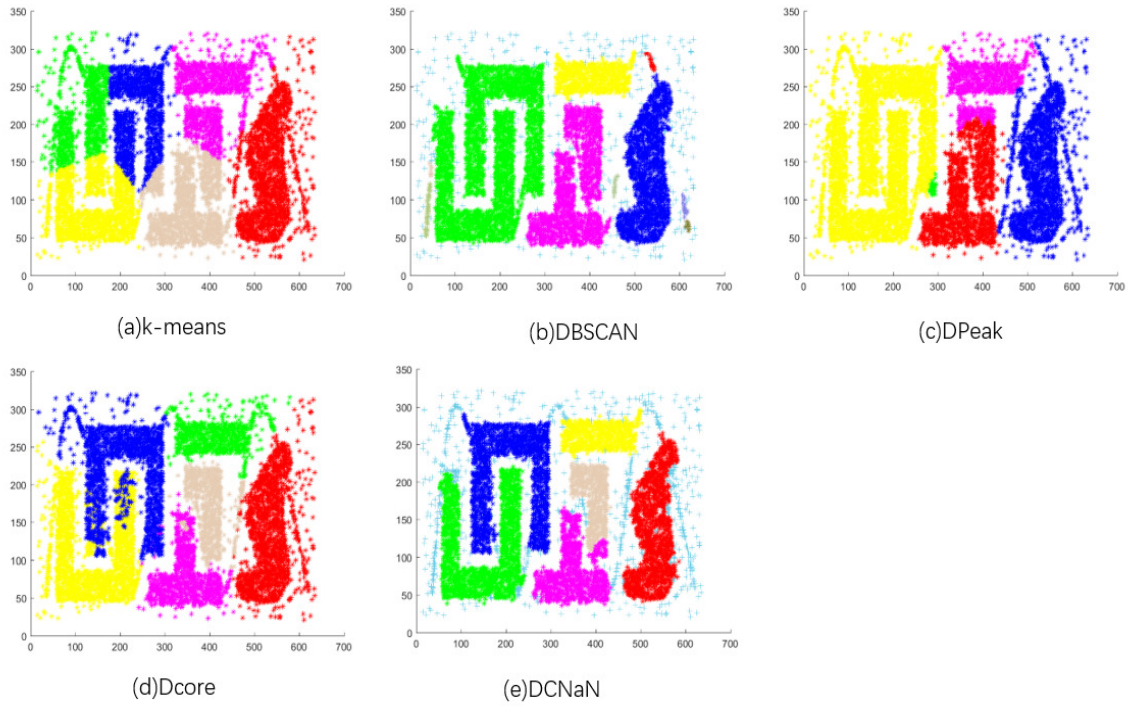

Figure 10: the results of 5 clustering algorithms in DS5:(a) K-means, (b) DBSCAN, (c) DPeak, (d) Dcore and (e)DCNaN

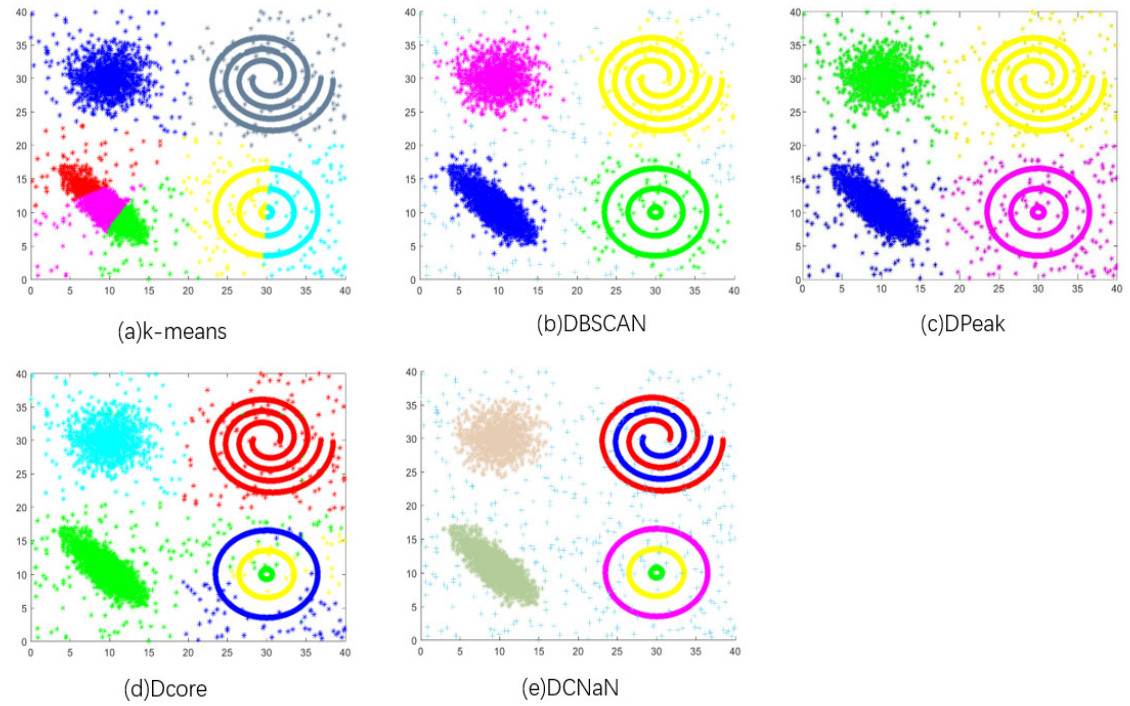

Figure 11: the results of 5 clustering algorithms in DS6:(a) K-means, (b) DBSCAN, (c) DPeak, (d) Dcore and (e)DCNaN

Table 4: the parameter setting of each clustering algorithm in UCI datasets

Real data	K-means	DBSCAN	DP	Dcore	DCNaN
Iris	N=3	Eps=1 Minpts=4	dc=6%	r1=0.3 r2=0.2 R=0.15 T1=5 Tn=0	-
Wine	N=3	Eps=0.4 Minpts=3	dc=6%	r1=1 r2=0.8 R=0.4 T1=10 Tn=2	-
Breast	N=2	Eps=0.5 Minpts=4	dc=6%	r1=0.4 r2=0.3 R=0.15 T1=2 Tn=0	-
BankNote	N=2	Eps=0.5 Minpts=4	dc=6%	r1=0.6 r2=0.5 R=0.6 T1=10 Tn=1	-
Control	N=6	Eps=1 Minpts=4	dc=6%	r1=1 r2=0.9 R=0.2 T1=10 Tn=2	-
Pageblock	N=5	Eps=1 Minpts=4	dc=6%	r1=0.2 r2=0.15 R=0.2 T1=10 Tn=2	-
Ionosphere	N=2	Eps=1 Minpts=4	dc=6%	r1=1.2 r2=1.1 R=1.2 T1=10 Tn=2	-

In Table5, the benchmark results in terms of AC, NMI and F-Measure in UCI datasets demonstrate that DCNaN performs better than K-Means, DBSCAN, DPeak and DCore do except in the Iris dataset. DCNaN gets the second best result that is 0.032 less than the result of Dcore in Iris. Furthermore, AC, NMI and F-Measure of DCNaN are almost consistent except in the Pageblock dataset and the Ionosphere dataset. In the BankNote dataset, K-Means and Dcore get the best results, just as DCNaN does. By repeatedly setting parameters, Dcore achieves better results, especially in the Iris dataset, the Wine dataset and the Breast dataset. But in more complex datasets, the performance of Dcore is not as good as that of DCNaN. Furthermore, many parameters in Dcore have to be set manually. The scores of K-Means are relatively high in simple datasets, but they cannot keep high in datasets with complex patterns and high dimension. The performances of DPeak and DBSCAN are relatively stable, but the scores of AC, NMI and F-Measure are generally low.

As shown in Table 6, DCNaN gets the second good scores in the Iris dataset and the best scores in other data sets. Therefore, it is obvious that the clustering results of DCNaN are better than that of other algorithms in terms of separation and compactness.

According to Table 7, we can clearly see that the running time of DCNaN is similar to that of other clustering algorithms.

Table 5: comparison of AC, NMI and F-Measure benchmarks in UCI datasets

Real data		K-means	DBSCAN	DP	Dcore	DCNaN
Iris	AC	0.873	0.667	0.883	**0.940**	0.908
	NMI	0.706	0.734	0.723	0.817	0.801
	F-Measure	0.873	0.778	0.893	0.940	0.912
Wine	AC	0.932	0.725	0.708	0.876	**0.942**
	NMI	0.801	0.477	0.419	0.616	0.821
	NMI	0.931	0.714	0.719	0.834	0.938
Breast	AC	0.941	0.923	0.655	0.921	**0.943**
	NMI	0.696	0.589	0.716	0.632	0.672
	F-Measure	0.948	0.845	0.792	0.951	0.933
BankNote	AC	**1.000**	0.555	0.731	**1.000**	**1.000**
	NMI	1.000	0.672	0.332	1.000	1.000
	F-Measure	1.000	0.714	0.751	1.000	1.000
Control	AC	0.431	0.350	0.500	0.500	**0.803**
	NMI	0.421	0.338	0.749	0.675	0.692
	F-Measure	0.413	0.388	0.665	0.588	0.681
Pageblock	AC	0.678	0.858	0.879	0.924	**0.927**
	NMI	0.124	0.002	0.101	0.248	0.257
	F-Measure	0.623	0.898	0.882	0.852	0.758
Ionosphere	AC	0.641	0.769	0.652	0.641	**0.784**
	NMI	0.012	0.165	0.073	0.052	0.197
	F-Measure	0.561	0.534	0.622	0.605	0.594

Table 6: comparison of DBI benchmark in UCI datasets

Real data		K-means	DBSCAN	DP	Dcore	DCNaN
Iris	DBI	0.6664	0.8664	0.4632	**0.4133**	0.4556
Wine	DBI	0.5643	0.7032	0.7234	0.6143	**0.5225**
Breast	DBI	0.8891	0.8956	1.4239	0.9135	**0.8832**
BankNote	DBI	**1.9378**	2.8321	3.1456	**1.9378**	**1.9378**
Control	DBI	1.1415	1.2312	0.9723	0.9875	**0.7321**
Pageblock	DBI	0.5271	0.5132	0.4983	0.4721	**0.4562**
Inonsphere	DBI	1.5114	1.4191	1.4872	1.3985	**1.0173**

From the above analysis, we can draw a conclusion that DCNaN provides an overall good performance in clustering compared to other clustering algorithms.

Table 7: the consuming time of five clustering algorithms

Dataset	K-means	DBSCAN	DPeak	Dcore	DCNaN
Iris	0.018	0.020	0.118	0.231	0.254
Wine	0.023	0.016	0.389	0.252	0.250
Breast	0.053	0.298	0.567	0.712	0.717
BankNote	0.041	0.075	4.030	1.451	1.504
Control	0.038	0.026	1.125	0.451	0.487
Pageblock	0.156	0.453	33.123	10.136	11.843
Inonsphere	0.057	0.024	0.234	0.312	0.371

5.5. Experiments on real datasets in the Olivetti face dataset

In order to further demonstrate the effectiveness of DCNaN, we also do experiments on the Olivetti face dataset[31]. As a widespread benchmark for machine learning algorithms, this dataset contains 400 facial images of 40 individuals, and each column shows 10 similar facial expressions. After data preprocessing of the first 100 images, we get a result matrix $D_{100\times98}$ and each line in the matrix represents a face image.

Table 8 displays the parameter setting of each clustering algorithm in the Olivetti face dataset. Table 9 and Table 10 compare the benchmarks of AC, NMI,F-Measure and DBI. Fig.12, Fig.13 and Fig.14 demonstrate the clustering results of DPeak, Dcore and DCNaN.

Figure 12: clustering results of DPeak in the Olivetti Face Database (Faces of the same color belong to the same cluster. Faces marked with red borders are incorrectly clustered images.) DPeak detects 9 clusters.

Figure 13: clustering results of Dcore in the Olivetti Face Database (Faces of the same color belong to the same cluster. Faces marked with red borders are incorrectly clustered images.) Dcore detects 11 clusters.

Figure 14: clustering results of DCNaN in the Olivetti Face Database (Faces of the same color belong to the same cluster. Faces marked with red borders are incorrectly clustered images.) DCNaN detects 10 clusters.

Table 8: the parameter setting of each clustering algorithm in the Olivetti face dataset

Real dataset	DPeak	Dcore	DCNaN
Olivetti face dataset	2%	r1=0.91 r2=0.81 R=0.91 T1=1 Tn=0	-

Table 9: comparison of AC, NMI and F-Measure benchmarks in the Olivetti face dataset

Real dataset		DPeak	Dcore	DCNaN
Olivetti face dataset	AC	0.660	0.700	0.790
	NMI	0.780	0.814	0.864
	F-Measure	0.665	0.696	0.788

Table 10: comparison of DBI benchmark in the Olivetti face dataset

Real dataset		DPeak	Dcore	DCNaN
Olivetti face dataset	DBI	1.9721	1.6780	1.5780

According to the results of experiments on the Olivetti face dataset shown in Table 9 and Table 10, we can see that DCNaN performs best in the scores of the AC, NMI, F-Measure and DBI. As for Dcore, the results are almost as same as what reported in the original paper[2]. The best accuracy, namely 0.7, is obtained by Dcore when r1=0.91, r2=0.81, R=0.91, T1=1 and Tn=0. It is important to note that the results of DPeak are better than what reported in the original paper[8]. All three algorithms, DPeak, Dcore and DCNaN, find density peaks, however, DPeak directly assigns each point to its nearest cluster center while Dcore and DCNaN assign each point to its convergent point. Therefore, all the three algorithms achieve almost the same acceptable performance in the Olivetti face dataset.

6. Conclusion

This paper proposes a new clustering algorithm named DCNaN whose core idea is to find density cores through dynamic scanning radiuses. Firstly, we automatically get the dynamic scanning radius for each point in a dataset according to the new neighbor concept. Secondly, we find local density peaks among their neighbors in the range of dynamic scanning radiuses and then get density cores that are constructed from above local density peaks. At last, we assign remaining points to the clusters of their convergent points. The density-core based structure can roughly retain the shape of extremely complex patterns, so DCNaN can deal with datasets of complex rectangle and manifold. The level partition of scanning radiuses can automatically detect outliers and noises, so DCNaN is robust to outliers and noises. The experiment results of synthetic and real datasets demonstrate that DCNaN can recognize extremely complex patterns with large variations in density. Most of all, unlike other algorithms, DCNaN does not need to manually set any parameters while Dcore has to manually set five parameters. Furthermore, DCNaN can be applied to many scenes, such as video coding[37], image segmentation, network intrusion detection and ordinal regression[5].

Although DCNaN performs well in various experiments, it has limitations. The scanning radius is chosen automatically, therefore, if DCNaN is applied to dense datasets, the number of local density peaks will be so large that time and space complexity increase.

Acknowledgments

We are obliged to DongDong Cheng for giving advice on our manuscript and providing the matlab code of natural neighbor search algorithm. We also thank anonymous reviewers for their help in improving the manuscript. This work is supported by the project of frontier and application foundation research program of CQ CSTC (no.cstc2017jcyjAX0340), and social undertakings and livelihood security science and technology innovation funds of CQ CSTC(no.cstc2017shmsA20013).

References

[1] C.C. Aggarwal, J. Han, J. Wang, P.S. Yu, A framework for projected clustering of high dimensional data streams, in: Proceedings of the Thirtieth International Conference on Very Large Data Bases-Volume 30, VLDB Endowment, 2004, pp. 852863

[2] Yewang Chen, Shengyu Tang, Lida Zhou, Cheng Wang, Jixiang Du, Tian Wang, Songwen Pei, Decentralized Clustering by Finding Loose and Distributed Density Cores, Information Sciences 000 (2016) 117

[3] Ester M, Kriegel H, Sander J, Xu X. A density-based algorithm for discovering clusters in large spatial databases with noise. In Proc. the 2nd International Conference on Knowledge Discovery and Data Mining, Aug. 1996, pp.226-231.

[4] J.A. Hartigan, M.A. Wong, Algorithm AS 136: A K-Means clustering algorithm, Appl. Stat. 28 (1) (1979) 100, doi:10.2307/2346830. URL:http://www.jstor.org/stable/10.2307/2346830?origin=crossref.

[5] B. Gu, V.S. Sheng, K.Y. Tay, W. Romano, S. Li, Incremental support vector learning for ordinal regression, Neural Netw. Learn. Syst. IEEE Trans. 26 (7) (2015) 14031416.

[6] Qingsheng Zhu, Ji Feng, Jinlong Huang, Natural neighbor: A self-adaptive neighborhood method without parameter K, Pattern Recognition Letters 80 (2016) 3036.

[7] Dongdong Cheng, Qingsheng Zhu, Jinlong Huang, Lijun Yang, Quanwang Wu, Natural neighbor-based clustering algorithm with local representatives, Knowledge-Based Systems 123 (2017) 238253.

[8] A. Rodriguez, A. Laio, Clustering by fast search and find of density peaks, Science 344 (6191) (2014) 14921496.

[9] J. Xie, H. Gao, W. Xie, X. Liu, P.W. Grant, Robust clustering by detecting density peaks and assigning points based on fuzzy weighted k-nearest neighbors, Inf. Sci. 354 (2016) 1940.

[10] D. Feldman, M. Schmidt, C. Sohler, Turning big data into tiny data: Constant-size coresets for k-means, pca and projective clustering, in: Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, in: SODA '13, SIAM, 2013, pp. 14341453. URL http://dl.acm.org/citation.cfm?id=2627817.2627920

[11] P.J. Rousseeuw, Finding Groups in Data: An Introduction to Cluster Analysis, John Wiley & Sons, 2009.

[12] B. King, Step-wise clustering procedures, J. Am. Stat. Assoc. 62 (317) (1967)86101

[13] W.W. Moss, H. Ja, Numerical taxonomy, Annu. Rev. Entomol. 18 (1973)227258.

[14] Y. Cheng, Mean shift, mode seeking, and clustering, TPAMI 17 (8) (1995) 790799

[15] K. Zhou, S. Yang, Exploring the uniform effect of fcm clustering: a data distribution perspective, Knowl. Based Syst. 96 (2016) 7683.

[16] J. Han, M. Kamber, J. Pei, Data Mining: Concepts and Techniques, The Morgan Kaufmann Series in Data Management Systems, 3rd ed., Morgan Kaufmann, 2011

[17] Q.-B. Liu, S. Deng, C.-H. Lu, B. Wang, Y.-F. Zhou, Relative density based k-nearest neighbors clustering algorithm, in: Proceedings of the International Conference on Machine Learning and Cybernetics, 2003, vol. 1, 2003, doi:10.1109/ICMLC.2003.1264457. 13317.

[18] R. Xu, I. Wunsch D., Survey of clustering algorithms, Neur. Netw. IEEE Trans. 16 (3) (2005) 645678.

[19] P. Van Kerm, et al., Adaptive kernel density estimation, Stata J. 3 (2) (2003) 148156.

[20] G.R. Terrell, D.W. Scott, Variable kernel density estimation, Annals Stat. (1992) 12361265

[21] G. Wang, Q. Song, Automatic clustering via outward statistical testing on density metrics, IEEE Trans. Knowl. Data Eng. 28 (8) (2016) 19711985.

[22] Gerhard X. Ritter, Jos-A. Nieves-Vzquez, Gonzalo Urcid, A simple statistics-based nearest neighbor cluster detection algorithm, Pattern Recognition 48 (2015) 918932

[23] M.J. Du, S.F. Ding, H.J. Jia, Study on density peaks clustering based on k-nearest neighbors and principal component analysis, Knowl. Based Syst. 99 (2016)135145.

[24] H. Jia, S. Ding, L. Meng, S. Fan, A density-adaptive affinity propagation clustering algorithm based on spectral dimension reduction, Neural Comput. Appl. 25(2014) 15571567.78.

[25] B.J. Frey, D. Dueck, Clustering by passing messages between data points, Science 315 (5814) (2007) 972976.

[26] Lijun Yang, Qingsheng Zhu, Jinlong Huang, Dongdong Cheng, Adaptive edited natural neighbor algorithm, Neurocomputing 230 (2017) 427433.

[27] Zhongyang Xiong, Ruotian Chen, Yufang Zhang, Xuan Zhang, Multi-density DBSCAN Algorithm Based on Density Levels Partitioning, Journal of Information & Computational Science 9: 10 (2012) 27392749

[28] M. Wu, B. Schlkopf, A local learning approach for clustering, in: Advances in Neural Information Processing Systems, 2006, pp. 15291536.

[29] X.V. Nguyen, J. Epps, J. Bailey, Information theoretic measures for clusterings comparison: is a correction for chance necessary? in: Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June 14-18, 2009, 2009, pp. 10731080.

[30] K. Bache, M. Lichman, UCI machine learning repository, 2013. URL http://archive.ics.uci.edu/ml

[31] F.S. Samaria, A.C. Harter, Parameterisation of a stochastic model for human face identification, in: Applications of Computer Vision, 1994., Proceedings of the Second IEEE Workshop on, IEEE, 1994, pp. 138142.

[32] J. Ha, S. Seok, J.S. Lee, Robust outlier detection using the instability factor, Knowl. Based Syst. 63 (2014) 1523.

[33] G. Karypis, E. H. Han, V. Kumar, CHAMELEON: A hierarchical clustering algorithm using dynamic modeling, Computer, 32(8), 1999, 68-75

[34] A. Gionis, H. Mannila, P. Tsaparas, Clustering aggregation, ACM Trans. Knowl. Discov. Data 1 (1) (2007) Article 4, doi:10.1145/1217299.1217303.

[35] Sergio Ramrez-Gallego, Bartosz Krawczyk, Salvador Garca, Micha Wozniak,Francisco Herrera, A survey on data preprocessing for data stream mining: Current status and future directions, Neurocomputing 239 (2017) 3957.

[36] Weiling Cai, A dimension reduction algorithm preserving both global and local clustering structure, Knowledge-Based Systems 118 (2017) 191203.

[37] Z. Pan, Y. Zhang, S. Kwong, Efficient motion and disparity estimation optimization for low complexity multi view video coding, Broadcast. IEEE Trans.61 (2) (2015) 166176.

[38] Tasdemir, K. and Mernyi, E.A Validity Index for Prototype-Based Clustering of Data Sets with Complex Structures,IEEE Trans. Systems, Man and Cybernetics,Part B,2011.

[39] Kuo-Lung Wu, Miin-Shen Yang, A cluster validity index for fuzzy clustering, Pattern Recognition Letters 26 (2005) 12751291