IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 4, 2019, accepted April 15, 2019, date of publication April 23, 2019, date of current version May 21, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2912332

Efficient Clustering Method Based on Density Peaks With Symmetric Neighborhood Relationship

CHUNRONG WU[1], JIA LEE[1,2], TEIJIRO ISOKAWA[3], JUN YAO[1], AND YUNNI XIA[1,2], (Senior Member, IEEE)

[1]College of Computer Science, Chongqing University, Chongqing 400044, China
[2]Chongqing Key Laboratory of Software Theory and Technology, Chongqing 400044, China
[3]Graduate School of Engineering, University of Hyogo, Himeji 671-2280, Japan

Corresponding author: Jia Lee (lijia@cqu.edu.cn)

This work was supported in part by the Natural Science Foundation of Chongqing under Grant cstc2016jcyjA1315, in part by the Fundamental Research Funds for the Central Universities under Grant 2019CDXYJSJ0022, and in part by the International Exchange Program of National Institute of Information and Communications (NICT).

ABSTRACT The density peaks clustering (DPC) is a clustering method proposed by Rodriguez and Laio (Science, 2014), which sets up a decision graph to identify the cluster centers of data points. Because the improper selection of its parameter cut-off distance will lead to the wrong selection of initial cluster centers with no corrective actions in the subsequent assignment process, DPC may not identify cluster centers with different densities accurately. Especially, all cluster centers are settled as soon as they are detected, after which the DPC simply assigns each point to the same cluster as its nearest neighbor of higher density. This tends to cause the erroneous assignments of data and thus degrade the efficiency of clustering. In this paper, we propose a robust clustering method which establishes a symmetric neighborhood graph over all data points, based on the k-nearest neighbors and reverse k-nearest neighbors of each point. In order to distinguish the density peaks from all data points, local densities of each point are calculated using the reverse k-nearest neighbors. After that, initial centers for clusters are estimated over the peaks and similar clusters are aggregated on the symmetric neighborhood graph, which ends up with every point being successfully assigned to a cluster. To testify the efficiency of the new clustering method, numerical experiments and comparison works have been done on a variety of artificial and real data sets for clustering.

INDEX TERMS Clustering, symmetric neighborhood, reverse k-nearest neighbors, density peaks clustering.

I. INTRODUCTION

Clustering is an indispensable and fundamental method for data mining. Up to now, various algorithms have been proposed which include partitioning methods [1]–[3], density-based clustering [4]–[7], spectral clustering [8], [9], ensemble clustering [10], [11], and hierarchical clustering [12]–[14]. This paper focuses on a well-known density-based clustering method, called Density Peaks Clustering (DPC), which was proposed by Rodriguez and Laio [15]. The DPC algorithm measures the local density of a data point by the number of points in a radius, and estimates a point to be a cluster center via its local density along with the distance from points of higher density (see equation 2). After the DPC locates cluster centers through a decision graph, each point will be assigned to a cluster which its nearest neighbor of higher density belongs to. Especially, the DPC also defines cluster core and cluster halo.

Though the DPC sounds simple and effective, the measurement of local densities heavily depends on the cut-off distance that is difficult to seek in advance of the clustering. In addition, the DPC usually requires each cluster center to be selected manually on the decision graph. Moreover, the assignment of a point to the same cluster as its nearest neighbor of higher density may ignore the actual distribution of data points, thereby degrading the accuracy of clustering.

In order to avoid the difficulty of choosing the parameter d_c, recently several researchers use k-nearest neighbors

The associate editor coordinating the review of this manuscript and approving it for publication was Muhammad Asif.

(kNN [25]), mutual k-nearest neighbors (MkNN [16], [17]), and natural neighbors (NN [22]) to estimate the local densities of each point [19]–[24]. Likewise, statistical test based clustering (STClu [19]) uses statistical test method to automatically detect cluster centers, and adaptive density peak clustering based on k-nearest neighbors (ADPC-kNN [23]) identifies some local core points with a certain density and merges clusters if they are density reachable. Unfortunately, STClu still suffers from the erroneous assignment of data due to the same assignment principle as DPC, and ADPC-kNN performs badly on data sets with different densities. For the sake of avoiding erroneous assignments, fuzzy weighted k-nearest neighbors density peak clustering (FkNN-DPC [20]) assigns non-outliers based on kNN starting from each cluster center, and then allocates the rest points, including outliers, using fuzzy weighted kNN. Natural neighbor-based clustering algorithm with density peaks (NaNDP [22]) expands each cluster from its center by searching natural neighbors of points in the cluster. Both FkNN-DPC and NaNDP, however, rely on the manual selection of initial cluster centers from decision graph on some data sets.

In this paper, we propose a new clustering method, called Density Peaks Clustering using Symmetric Neighborhood Relationship (DPC-SNR). Our new method establishes a symmetric neighborhood graph over all data points, which is achieved using the k-nearest neighbors and reverse k-nearest neighbors of each point. Especially, for the sake of distinguishing the peaks from other points, local densities of each point are calculated using the reverse k-nearest neighbors, which enables more efficient identification of initial centers for clusters. Combined with the local densities, all points are sorted based on the distances from points of higher density. Starting from the peak point, the DPC-SNR assigns each point to a proper cluster through a breadth first search on symmetric neighborhood graph. Finally, the DPC-SNR merges every tiny cluster into a major cluster based on their mutual connectivity.

This paper is organized as follows. Section II outlines the DPC together with several variation of the algorithm, and introduces the concept of symmetric neighborhood relationship. Section III describes our new clustering algorithm. Section IV performs experiments on a number of synthetic and real data sets, and analyzes the efficiency of our clustering method. This paper finishes with conclusions in Section V.

II. RELATED WORKS

In this section, we will review the process of DPC and introduce these studies which arouse interests in DPC.

A. DENSITY PEAKS CLUSTERING

DPC [15] thinks local density of a cluster center is higher than other points in the same group and has a relatively large distance from any points with a higher local density. DPC uses the local density ρ_i of a point and its distance δ_i from points with higher density to construct a decision graph. If ρ_i and δ_i of a point are higher, the point may be a cluster center. The local density ρ_i is defined as follows:

$$\rho_i = \sum_j \chi(dist(i,j) - d_c) \tag{1}$$

where d_c is a cutoff distance and $\chi(a) = 1$ when $a < 0$, $\chi(a) = 0$ when $a \leq 0$. $dist(i,j)$ denotes Euclidean distance between point i and point j. The formula of the distance δ_i from points of higher density shows as follows:

$$\delta_i = min_{j:\rho_j > \rho_i}(dist(i,j)) \tag{2}$$

For the point with highest density, they conventionally take $\delta_i = max_j(dist(i,j))$. Rodriguez and Laio also use another way to present ρ_i, which is defined as a Gaussian kernel function:

$$\rho_i = \sum_j exp(-\frac{dist^2(i,j)}{d_c^2}) \tag{3}$$

where the point j is eligible when $dist(i,j)$ is less than d_c. d_c is the only influence parameter in two formulas above. From the public code [40], we can know that the value of d_c comes from one of the distances between two points. The algorithm firstly sorts the values of distances, then sets p percent of the data set as the position, and finally chooses the value of the distances at this position as d_c. Therefore, parameter p is considered rather than d_c for simplicity in our paper.

In addition, we find there is another issue in DPC. DPC performs badly in manifold data sets with different densities. Fig. 1 presents that DPC cannot find cluster centers in manifold data with low density, and two cluster centers are located in manifold data with high density. There is no influence to select cluster centers about the different values of parameter p. In this case, DPC is not able to find the correct clusters.

There are some new formulas to get ρ_i. In order to avoid the influence of d_c, many studies introduce kNN to modify the formula of ρ_i. STClu [19] defines ρ_i as follows:

$$\rho_i = \frac{k}{\sum_{j \in kNN(i)} dist(i,j)} \tag{4}$$

where k is an input parameter, $kNN(i)$ the number of kNN of point i. The formula of local density proposed by FkNN-DPC [20] is:

$$\rho_i = \sum_{j \in kNN(i)} exp(-dist(i,j)) \tag{5}$$

Density peaks clustering based on k-nearest neighbors and principal component analysis (DPC-kNN-PCA) [21] changes the formula of local density, which shows as follows:

$$\rho_i = exp(-\frac{1}{k} \sum_{j \in kNN(i)} dist^2(i,j)) \tag{6}$$

where k is computed as a percent of the number of data sets. Though the computing formula of ρ_i in ADPC-kNN [23] also uses kNN, the parameter of d_c is used in the formula. Moreover, ADPC-kNN uses a new method to get the value

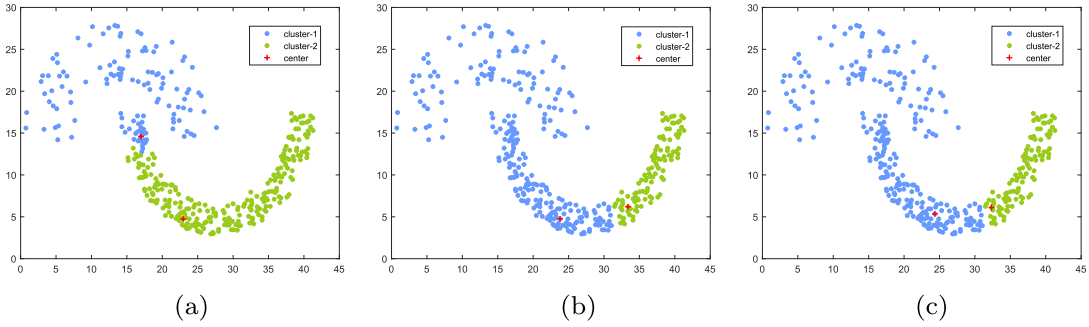


FIGURE 1. DPC on manifold data sets of different densities. (a) p=0.8. (b) p=6.0. (c) p=10.0.

of d_c. The local density ρ_i is identified by the following formula:

$$\rho_i = \sum_{j \in kNN(i)} exp(-\frac{dist^2(i,j)}{d_c^2}) \tag{7}$$

where $d_c = \mu + \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(d_i - \mu)^2}$, μ is the mean value of d_i of all points, and $d_i = max_{j \in kNN(i)}(dist(i,j))$.

NaNDP [22] uses natural neighbors to compute the local density, and constructs maximum neighborhood graph to assign non-core points. The formula of ρ_i shows as follows:

$$\rho_i = \frac{Max_{nb}}{\sum_{j \in N(i,Max_{nb})} dist(i,j)} \tag{8}$$

where Max_{nb} is the maximum number of natural neighbors, $N(i, Max_{nb})$ the Max_{nb} nearest neighbors of point i.

Comparative density peaks clustering (CDP) [24] introduces the mutual k-nearest neighbors to get the local density ρ_i, which also redefines δ_i. The computational process of δ_i is a little complex, thus there is no description in detail here, which shows in the paper. The computational formula of ρ_i is defined as:

$$\rho_i = \sum_{j \in MkNN(i)} exp(-\frac{dist^2(i,j)}{d_c^2}) \tag{9}$$

where $MkNN(i)$ is a set of points associated with point i in the MkNN.

Compared the formula of ρ_i in DPC with formulas in related studies above, we can see that the region of influence is diminishing. However, we think that the differences between core points and non-core points should be increased. Therefore, we choose reverse kNN as the region of influence, which means the value degree from others. Cluster centers should be surrounded by dense points so that value degree of cluster centers will be higher. If the number of *kNN* about a point is pre-computed, the complexity of computing the local density will reduce. The cluster centers are thought to have higher values of ρ_i and δ_i. Therefore there are two possible methods for selecting centers. One is to use a rectangular box in a decision graph based on matlab to select manually these points. The other is to compute a new quantity $\gamma_i = \rho_i \delta_i$ for each point i, then sort γ_i in descending order and finally choose first m values of γ_i.

B. SYMMETRIC NEIGHBORHOOD RELATIONSHIP

kNN is one of the simplest methods in data mining classification technology, which arouses the interest of many researchers [25], [26]. This approach also has been applied to clustering [27], [28]. The distance between two points is generally achieved by calculating the Euclidean distance. There are also many studies about other nearest neighbors based on kNN, including natural neighbors, mutual k-nearest neighbors and shared nearest neighbors [29]. The main idea of mutual k-nearest neighbors and nature neighbors are using symmetric neighborhood relationship.

kNN and reverse kNN are symmetric neighborhood relationship [30]. We always assume that there are n data points with m dimension. After the distance between points is computed, we should sort these distances in ascending order to find the first k nearest distances. All points contained in the k nearest distance corresponds to kNN, and we also get reverse kNN during the process. More specifics about kNN and reverse kNN are discussed in Section III.

III. CLUSTERING ALGORITHM BASED ON SYMMETRIC NEIGHBORHOOD RELATIONSHIP

There are two respects to improve DPC by respectively using reverse kNN and symmetric neighborhood relationship for computing the local density of a point and changing the assignment method. This section will present the details of the proposed clustering algorithm and analyze its complexity.

This paper presents a new clustering algorithm by using symmetric neighborhood relationship. Here is the basic idea: firstly, find the symmetric neighborhood of each point; then calculate local density and distance of each point using reverse kNN, and cluster from highest value of density and distance by undertaking a breadth first search of the symmetric neighborhood; finally, identify big clusters and small clusters, and combine small clusters into big ones. The details of DPC-SNR algorithm are shown as Algorithm 1, and the clustering process on a simple data set is presented in Fig. 2, where red circles and rectangles to represent outliers and cluster centers respectively. Furthermore, different shapes mean different clusters.

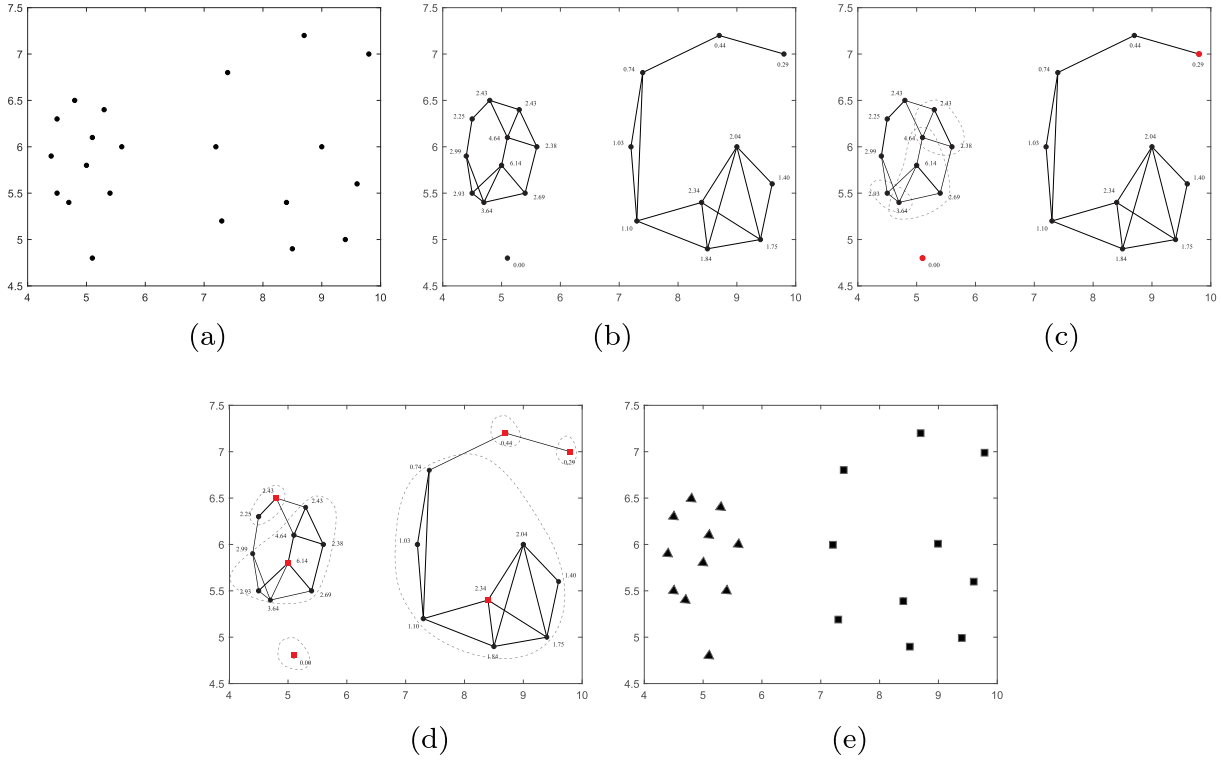


FIGURE 2. The clustering process of DPC-SNR. (a) Data set. (b) Symmetric neighborhood. (c) Clustering. (d) Initial clusters. (e) Final clusters.

A. DENSITY PEAKS CLUSTERING IN SYMMETRIC NEIGHBORHOOD

Let D be a database, i and j be some objects in D, and k be a positive integer. We use $dist(i, j)$ to denote the Euclidean distance between object i and j.

The k-distance of i, denoted as $k_{dist}(i)$, is the distance $dist(i, o)$ between point i and point o in D, which shows as:

- *For at least k objects* $o' \in D$ *it holds that* $dist(i, o') \leq dist(i, o)$, *and*
- *For at most (k-1) objects* $o' \in D$ *it holds that* $dist(i, o') < dist(i, o)$

If a point j meets $dist(i, j) \leq k_{dist}(i)$, then call j as one of the kNN of i. A set of points J which contains finite points j forms the kNN of i, denoted as $kNN(i)$. The definition of $kNN(i)$ is:

$$kNN(i) = \{J \in D | dist(i, J) \leq k_{dist}(i)\} \tag{10}$$

The point i is regarded as the reverse kNN of j, and a set of points I which contains finite points i composes the reverse k-nearest neighborhood, denoted as $RkNN(j)$. $RkNN(i)$ can be defined as:

$$RkNN(i) = \{j | j \in D, i \in kNN(j)\} \tag{11}$$

The results of intersection of the k-nearest neighborhood and the reverse k-nearest neighborhood are used to estimate the density distribution around i, and the neighborhood space is called as symmetric neighborhood of i, denoted as $SN_k(i)$. $SN_k(i)$ means that two people are true friends only when they agree with each other, which shows as follows:

$$SN_k(i) = \{o | o \in D, o \in (kNN(i) \cap RkNN(i))\} \tag{12}$$

In general, searching kNN of point i will return at least k results, while the results of $RkNN$ will be zero, one or many. We consider the influence of a point from other points, therefore we use the reverse kNN of a point to calculate the local density instead of other nearest neighbors, which makes us identify cluster centers more easily. The new local density can be defined as:

$$\rho_i = \sum_{j \in RkNN(i)} exp(-dist^2(i, j)) \tag{13}$$

where $RkNN(i)$ is the reverse kNN of point i.

This definition can guarantee that the local density ρ_i of point i is affected by the distribution information of its reverse kNN, while the original definition in [15] is calculated using the cutoff distance d_c. It is hard to ensure the value of cutoff distance d_c, which will affect the local density of points and selection of cluster centers. Furthermore, the determination of parameter k is easier than that of cutoff distance d_c.

B. EXTENDING CLUSTER FROM PEAKS ON SYMMETRIC NEIGHBORHOOD GRAPH

The graph constructed by linking the symmetric neighborhood of each point is called as symmetric neighborhood graph(SNG). Outliers are regarded as points with less than two neighbors in the symmetric neighborhood. Though there

Algorithm 1 DPC-SNR Algorithm

Input: The number of nearest neighbors, k; The using data set, D;
Output: The set of clusters, $C \leftarrow c_1, c_2, \ldots, c_m$;

Initializing: $dist[i,j] \leftarrow 0$, $\rho_i \leftarrow 0$, $Outlier[i] \leftarrow 0$, $cl[i] \leftarrow -1$;
use the formula of Euclidean distance to calculate the distance between point i and point j, then get $dist[i,j]$;
use a common method to get kNN and $RkNN$;
for each $a \in D$ **do**
  $SN_k[a] \leftarrow kNN[a] \cap RkNN[a]$;
end for
Calculate ρ_i and δ_i for point i respectively using (13) and (2);
get the product γ_i of ρ_i and δ_i, and sort in descending order;
for each $a \in D$ **do**
  if $SN_k[a] \leq 1$ **then**
    $Outlier[a] \leftarrow 1$
  end if
end for
$C \leftarrow$ Assignment method($Outlier$, SN_k, $dist$, γ);

will be a boundary between two clusters with different density, we also need to add some rules to reinforce the boundary. We assume a point x is an extending point, and a point y is one of points in the symmetric neighborhood of x, which has not been extended. If y meets all the following rules, y will be extended. Rules include y is not a outlier, y is not visited and the distance between x and y is less than the mean distance between y and points from symmetric neighborhood of y. The third rule means if y is a little far away from x, then y will not be extended temporarily.

It can be seen from SNG that if there are close connections between two initial clusters, then the two initial clusters may belong to a same cluster. A point with larger values of local density and distance is more likely to be a cluster center. Therefore, we choose to cluster from a point with the largest product value of local density and distance by undertaking a breadth first search of SNG. After traversing the whole SNG, there will be some clusters. We can obtain main clusters from the process of clustering, therefore, the cluster with less than k points is a small cluster, including outliers. Then these clusters are automatically assigned to large clusters and small clusters. We combine small clusters into large clusters whose number of connected edges are largest in SNG. If there still are small unassigned clusters, we assign the small clusters to clusters which most of points in their reverse kNN belong to. The assignment method is described in Algorithm 2. In the algorithm, $visit[i]$ is a flag, which means whether i is visited.

C. THE COMPLEXITY ANALYSES OF DPC-SNR

Suppose that there are N points in the data set and let num denote the number of clusters. The space complexity of

Algorithm 2 Assignment Method

Input: The tag parameters of points whether they are outliers, $Outlier$; The symmetric neighborhood of all points, SN_k; The distance between two points, $dist$; The product of density and distance of all points, γ;
Output: The final clusters, la_C;

Initializing: $Q \leftarrow \emptyset$, $visit[i] \leftarrow 0$; $nclust \leftarrow 0$;
for $i = 1$ *to* n **do**
  $center \leftarrow \gamma[i]$;
  if $visit[center] = 1$ **then**
    continue;
  end if
  $visit[center] \leftarrow 1$;
  $nclust \leftarrow nclust + 1$;
  $cl[center] \leftarrow nclust$;
  Add $center$ into Q;
  while $Q \neq \emptyset$ **do**
    Assign the first value of Q to $first$
    $tmp \leftarrow SN_k[first]$;
    Delete $first$ from Q;
    for each $a \in tmp$ **do**
      calculate the average distance ave of point a in its symmetric neighborhood;
      if $visit[a] = 0$ **and** $outlier[a] = 0$ **and** $dist[first, a] \leq ave$ **then**
        Add a into Q;
        $cl[a] \leftarrow i$;
        $visit[a] \leftarrow 1$;
      end if
    end for
  end while
end for
according to the number of non-repeating values of cl, get C;
according to the value of k, get small clusters sm_C and large clusters la_C;
for each $a \in sm\_C$ **do**
  count the number of edges connected to all large clusters la_C;
  combine small cluster a into the large cluster with the largest number of edges;
  delete small cluster a from sm_C;
end for
if there are small clusters unassigned, assign the small clusters to clusters which most of points in their reverse kNN belong to;
return la_C

DPC-SNR relies on the three aspects: the matrix storing the distance between two points ($O(N^2)$), two attributes ρ_i and δ_i ($O(2N)$) and three neighborhood kNN, reverse kNN and SN ($O(KN + N + N)$). Spaces required by these points do not exceed $O(N^2)$, thus the space complexity of DPC-SNR is the same with DPC in [15].

The time complexity of DPC-SNR relies on the following aspects: (a) using a common method to get kNN and reverse kNN ($O(N^2)$); (b) computing the distance between two points ($O(N^2)$); (c) calculating the symmetric neighborhood of each point ($O(N)$); (d) calculating the local density ρ_i with reverse kNN ($O(LN)$), L is the number of points in reverse kNN of point i, and L is not greater than N; (e) clustering from the largest value of γ_i on SNG($O(CN^2)$), and C is the number of symmetric neighborhood of a point; (f) combining small clusters into large clusters($O(M^3)$), and M is far less than N. Therefore, the overall time complexity of DPC-SNR is $O(N^2)$ which is the same with DPC.

IV. EXPERIMENTS

In the section, experiments were conducted on synthetic and real data sets to test the performance of DPC-SNR. The performance of DPC-SNR was compared with these clustering algorithms including affinity propagation (AP) in [18], DBSCAN in [4], DPC in [15], ADPC-kNN in [23], and DPC-kNN-PCA in [21]. The codes of DPC and ADPC-kNN were provided by their authors, and the code of DPC shows in [40]. We do not consider finding the cluster halo when we do experiments on DPC. DBSCAN, DPC-kNN-PCA and AP were implemented with MATLAB R2017b.

Moreover, we use these parameters mentioned in their papers to conduct these experiments, including percent p in DPC, iteration invariant number *convits* and maximum number of iterations *maxits* in AP, the number of nearest neighbors k in ADPC-kNN, the number of nearest neighbors k and a percentage of the number of points m in DPC-kNN-PCA and the distance *Eps* and the number of points in the current distance *MinPts* in DBSCAN. Especially, we set that damping coefficient *lam* is 0.9 in AP throughout the experiments. We implemented the algorithms on each data set for a number of times and listed the best result of each method out.

A. ASSESSMENT OF CLUSTERING PERFORMANCE

We use clustering accuracy(Acc) index which is often used in [21], [22] and Normalized Mutual Information(NMI) [41] index to evaluate the clustering performance on these experiments. And the formula of Acc is as follows:

$$Acc = \frac{1}{N}\sum_{i=1}^{N}\delta(r_i, map(s_i)) \tag{14}$$

where r_i is the real cluster label, s_i the serial number obtained by clustering. If $a = b$, $\delta(a, b) = 1$; otherwise, $\delta(a, b) = 0$. The larger value of Acc means the better clustering performance of the algorithm.

The formula of NMI shows as follows:

$$NMI(X, Y) = \frac{MI(X, Y)}{\sqrt{H(X) * H(Y)}} \tag{15}$$

where $MI(X, Y)$ is the mutual information between two random variables X and Y, $H(Z)$ the entropy of random variables Z. When the value of NMI is bigger, clustering performance is better.

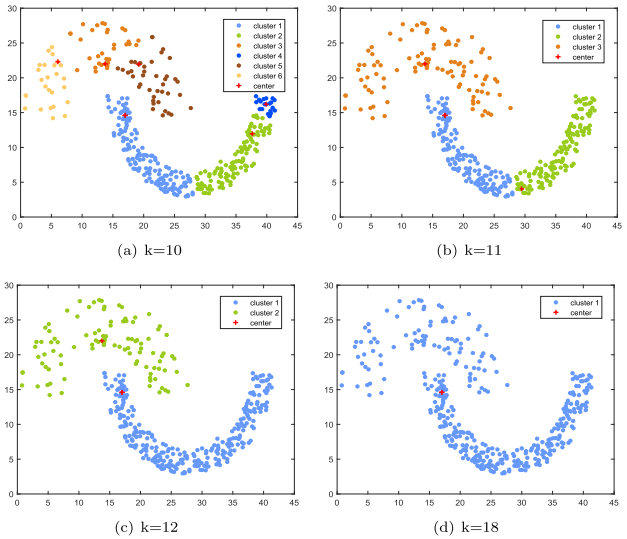


FIGURE 3. The clustering results about 10NN, 11NN, 12NN, and 18NN on data set 1.

B. ANALYSIS OF DIFFERENT VALUES ABOUT THE PARAMETER

We will discuss different values of k in different data sets, which will be presented later. As shown in Fig. 3, we can see that there are six centers in the data when the value of k is 10. When the value of k is less than 10, the number of clusters is more than six. However, when the value of k is suitable, there will be the right number of clusters. When the value of k is large, two clusters will be merged into a cluster. Therefore, the best value of k is [12], [17]. We also do experiments on other five artificial data sets, and we find there is also a range when we get good results on these data sets. However, there is only a best value on data set 2. If we choose a value of k that exceeds the range, the result will be bad or all clusters will be a cluster.

We also do experiments on real data sets. Fig. 4 shows that there is only a best value of k with the number of real clusters. However, there is stability in Iris and Banknote_A on the number of clusters, Acc and NMI with the increase of k value. There is a fluctuation in Breast_C before best values of Acc and NMI, and then the number of clusters is 1. We can also see that there are best values of Acc and NMI, however, the number of clusters is wrong. We do experiments on other eight data sets, and find that change rules of the eight data sets are similar with four data sets previously mentioned. We also use nature neighbors to get k in order to reduce the parameter. However, we find that nature neighbors cannot find the best value of k in many data sets.

C. CLUSTERING ON ARTIFICIAL DATA SETS

We choose six artificial data sets to demonstrate the efficiency of DPC-SNR, which are illustrated in Fig. 5. Data set 1,

TABLE 1. Comparison of two benchmarks for six clustering algorithms on artificial data sets.

Algorithm	Par	Val	Cl	Acc	NMI	Algorithm	Par	Val	Cl	Acc	NMI
Data set 1						Data set 2					
DPC-SNR	k	15	2	**1.00**	**1.00**	DPC-SNR	k	8	6	**0.96**	**0.95**
DPC	p	0.8	2	0.92	0.65	DPC	p	3.5	6	0.67	0.77
DPC-kNN-PCA	k/m	10/355.7	2	0.86	0.51	DPC-kNN-PCA	k/m	6/359.1	6	0.78	0.80
ADPC-kNN	k	38	2	0.43	0.19	ADPC-kNN	k	30	5	0.67	0.79
DBSCAN	$Eps/MinPts$	3.2/7	2	0.81	0.26	DBSCAN	$Eps/MinPts$	2.2/8	4	0.83	0.83
AP	$convits/maxits$	5/28	2	0.31	0.26	AP	$convits/maxits$	5/32	6	0.50	0.63
Data set 3						Data set 4					
DPC-SNR	k	12	3	**1.00**	**1.00**	DPC-SNR	k	7	3	**1.00**	**1.00**
DPC	p	9.6	3	0.43	0.11	DPC	p	2.5	3	**1.00**	**1.00**
DPC-kNN-PCA	k/m	15/239.2	3	0.58	0.39	DPC-kNN-PCA	k/m	10/124.8	3	0.36	0.00
ADPC-kNN	k	24	3	0.59	0.33	ADPC-kNN	k	15	3	**1.00**	**1.00**
DBSCAN	$Eps/MinPts$	0.06/4	3	**1.00**	**1.00**	DBSCAN	$Eps/MinPts$	3.6/7	3	**1.00**	**1.00**
AP	$convits/maxits$	5/32	3	0.36	0.20	AP	$convits/maxits$	5/37	2	0.36	0.00
Data set 5						Data set 6					
DPC-SNR	k	13	5	**1.00**	**1.00**	DPC-SNR	k	6	4	**1.00**	**1.00**
DPC	p	3.5	5	**1.00**	**1.00**	DPC	p	1.6	4	0.30	0.46
DPC-kNN-PCA	k/m	30/153.2	5	**1.00**	**1.00**	DPC-kNN-PCA	k/m	16/600	4	0.38	0.21
ADPC-kNN	k	28	5	0.85	0.92	ADPC-kNN	k	27	2	0.31	0.02
DBSCAN	$Eps/MinPts$	2.8/6	3	0.65	0.79	DBSCAN	$Eps/MinPts$	0.07/5	4	**1.00**	**1.00**
AP	$convits/maxits$	5/30	5	0.96	0.91	AP	$convits/maxits$	5/200	4	0.25	0.00

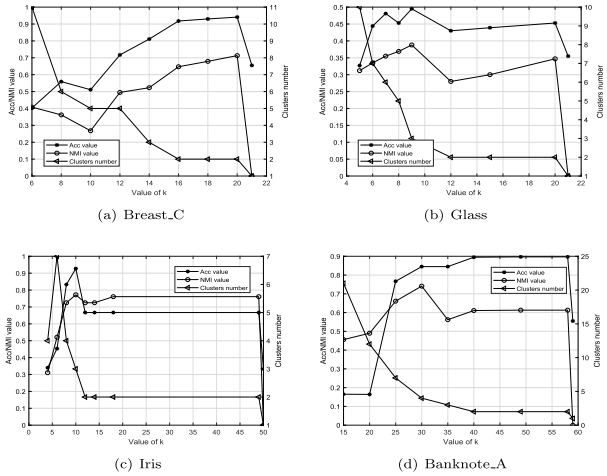


FIGURE 4. Different values of k, clusters number, Acc and NMI on data sets Breast_C, Glass, Iris and Banknote_A.

taken from [42], consists of two manifold data with different densities and contains 373 points. Data set 2, taken from [43], is composed of three spherical data and three irregular data and has a total of 399 points. Data set 3 consists of three ring data with different densities and contains 299 points. Data set 4, from [9], a total of 312 points, is composed of three manifold data. Data set 5 from [44], consists of five spherical data with different densities and contains 383 points. Data set 6, has a total of 1000 points and includes four ring data with the same density. And information about some data sets also shows in [45]. The clustering results of these algorithms are shown in Fig. 6–11. The comparison of these algorithms on Acc and NMI scores are shown in Table. 1, and the running time is shown in Table. 2.

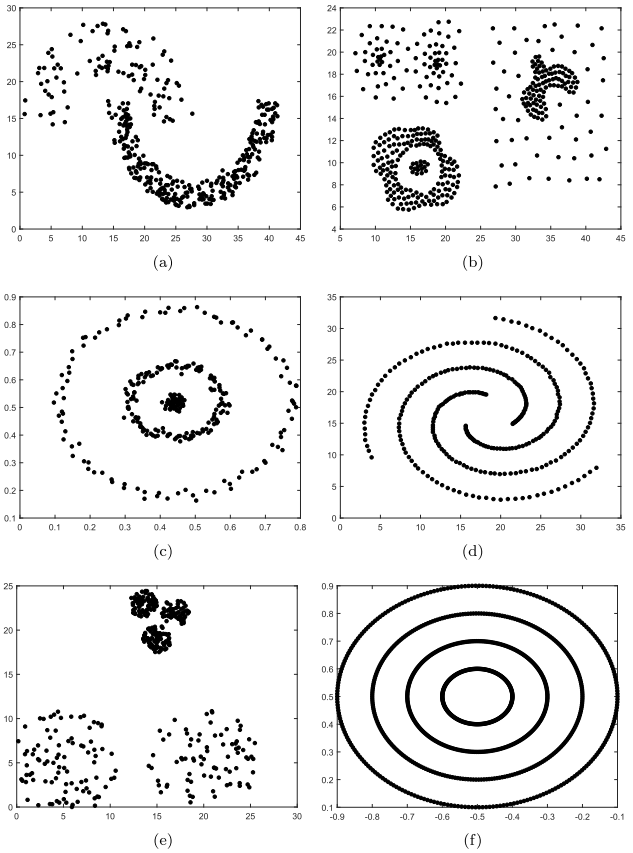


FIGURE 5. Six original synthetic data sets. (a) Data set 1. (b) Data set 2. (c) Data set 3. (d) Data set 4. (e) Data set 5. (f) Data set 6.

Fig. 6 shows ADPC-kNN, AP, DBSCAN, DPC-kNN-PCA and DPC algorithms cannot find correct clusters and they all perform badly on manifold data with different densities. However, DPC-SNR can identify correct clusters, the

TABLE 2. The running time(s) of the algorithms on artificial data sets.

Dataset	DPC-SNR	DPC	DPC-kNN-PCA	ADPC-kNN	DBSCAN	AP
Data set 1	0.39	0.22	0.40	0.18	0.27	0.50
Data set 2	0.49	0.30	0.41	0.25	0.30	0.56
Data set 3	0.38	0.23	0.34	0.23	0.26	0.60
Data set 4	0.35	0.19	0.34	0.21	0.24	0.38
Data set 5	0.44	0.25	0.40	0.24	0.29	0.56
Data set 6	1.47	1.30	1.34	0.71	1.02	5.39

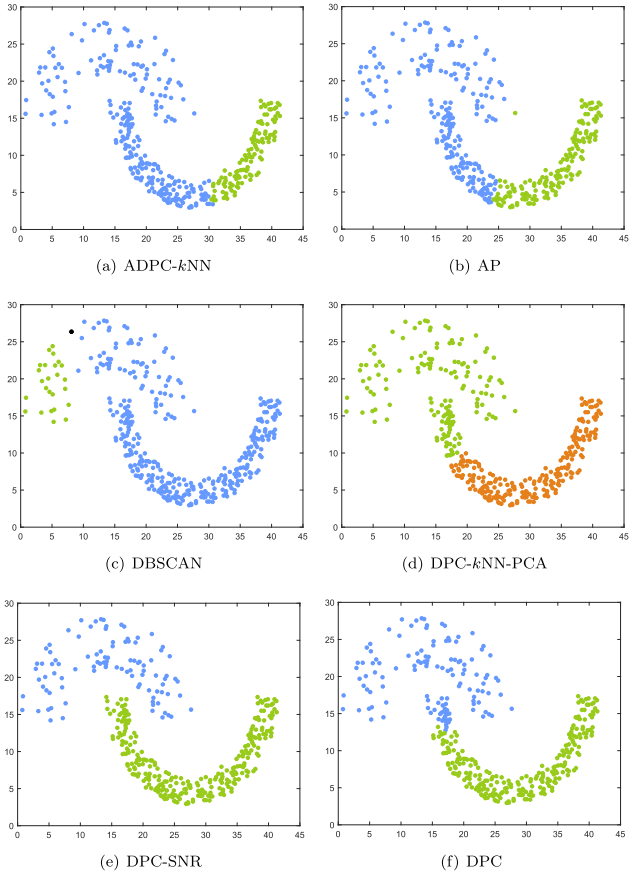


FIGURE 6. The clustering results of ADPC-kNN, AP, DBSCAN, DPC-kNN-PCA, DPC-SNR, and DPC algorithms on data set 1.

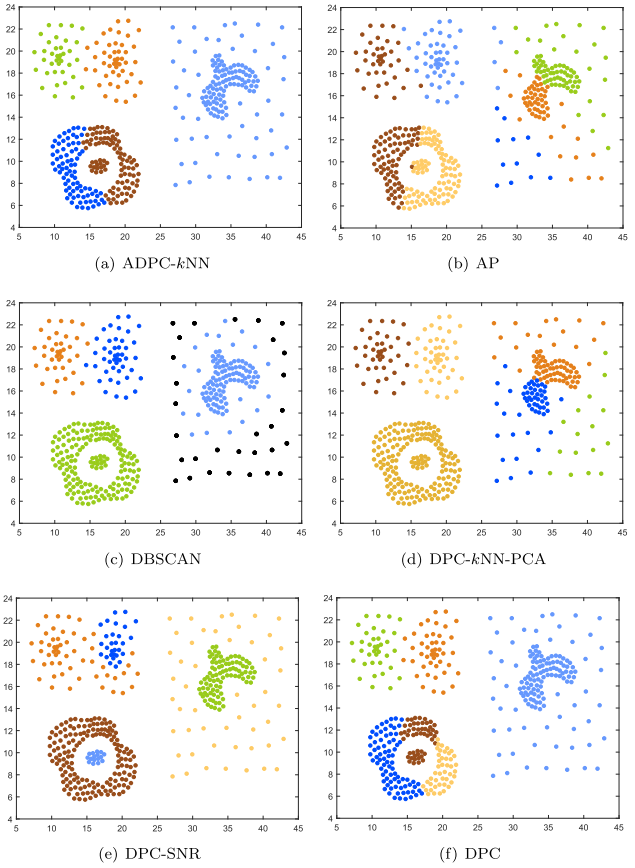


FIGURE 7. The clustering results of ADPC-kNN, AP, DBSCAN, DPC-kNN-PCA, DPC-SNR, and DPC algorithms on data set 2.

benchmarks data of which are all 1.00 in Table. 1. Data set 4 is also a kind of manifold data set with almost the same density. As shown in Fig. 9, ADPC-kNN, DBSCAN, DPC-SNR and DPC can perform well and the benchmarks data of four algorithms are all 1.00, while DPC-kNN-PCA still cannot get the right results. DPC-kNN-PCA uses nearby principle to assign other non-core points, so that there are many wrong assignments. Therefore, we can see that DPC-kNN-PCA may be not suitable for manifold data sets.

Fig. 7 shows the clustering results of six algorithms on data set 2. DPC-SNR can find all clusters out correctly and assign almost all points to their corresponding clusters. However, the two clusters on the right side are incorrectly clustered into one by ADPC-kNN, DBSCAN and DPC, which are also treated as parts by DPC-kNN-PCA and AP. Similarly, DBSCAN and DPC-kNN-PCA mistakenly identify the two clusters on the bottom right side as one, which are divided into many parts by other algorithms. Fortunately, all algorithms can correctly identify two spherical clusters on the top right side.

Fig. 8 shows DPC-SNR and DBSCAN can find all clusters out correctly and assign all points to their corresponding clusters with benchmarks data of 1.0 in Table. 1. There are similar clustering results between ADPC-kNN and DPC-kNN-PCA, which show three and one clusters respectively in the outer and inner ring. DPC and AP show the opposite effect on the outer two rings. Fig. 11 shows the clustering results of six algorithms on data set 6. DPC-SNR and DBSCAN also can get true clusters and the highest benchmarks data in Table. 1.

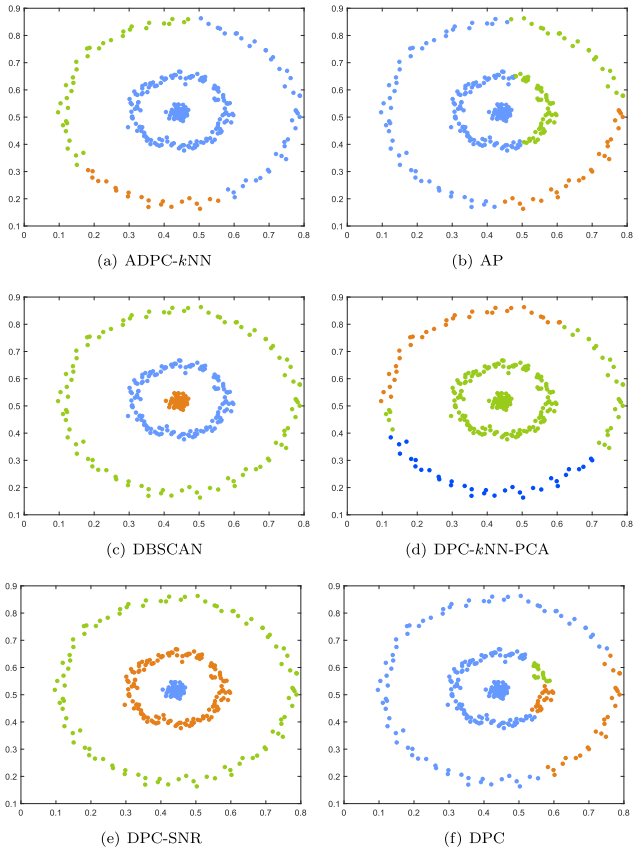


FIGURE 8. The clustering results of ADPC-*k*NN, AP, DBSCAN, DPC-*k*NN-PCA, DPC-SNR, and DPC algorithms on data set 3.

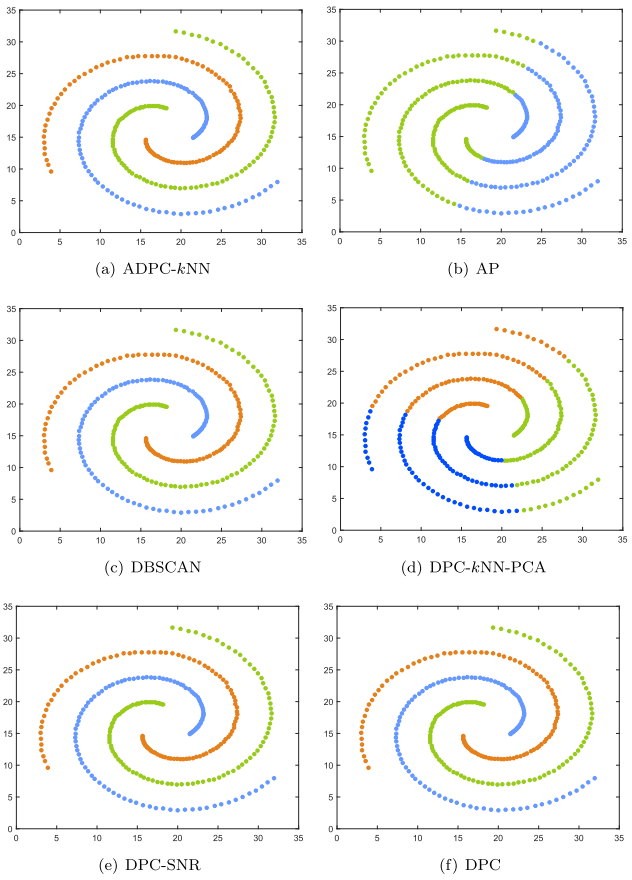


FIGURE 9. The clustering results of ADPC-*k*NN, AP, DBSCAN, DPC-*k*NN-PCA, DPC-SNR, and DPC algorithms on data set 4.

Clustering results of ADPC-*k*NN, AP, DPC-*k*NN-PCA and DPC are similar, which present some fan-shaped structures. However, ADPC-*k*NN cannot find out the number of clusters correctly.

Fig. 10 shows the clustering results of six algorithms on data set 5. DPC, DPC-*k*NN-PCA and DPC-SNR perform well on the data set, and the benchmarks data of these algorithms are all 1.00 in Table. 1. ADPC-*k*NN can identify clusters with low density, however ADPC-*k*NN only can find two clusters when there are three high density spherical data. Though DBSCAN can also find clusters with low density, DBSCAN performs worse on high density spherical data and considers three clusters as one. Fortunately, AP can identify correctly these clusters with some incorrect assignment points.

From the above results and analysis, we can see that DBSCAN, DPC and ADPC-*k*NN algorithms have a certain capacity to cluster manifold data with high density. However, as shown in the above results, they can hardly cluster the manifold data correctly with different densities. DPC-*k*NN-PCA performs worse on all manifold data. DPC-*k*NN-PCA and DPC can correctly identify spherical data with different densities, while ADPC-*k*NN and DBSCAN cannot. ADPC-*k*NN, DPC-*k*NN-PCA and DPC perform badly on a ring data, while DBSCAN can correctly identify the data. AP can identify spherical data, however AP perform badly on data sets of other shapes. Therefore, from the results of artificial data sets, we can see that DPC-SNR can get the right number of final clusters with the influence of a parameter k. The application scope of DPC-SNR is wider than other clustering algorithms, though the running time of DPC-SNR is higher in Table. 2. DPC-SNR can get satisfactory clustering results on complex manifold, ring and spherical data sets. In order to demonstrate the efficiency of DPC-SNR, we also do experiments on real data sets as the following section.

D. CLUSTERING ON REAL DATA SETS

In order to further demonstrate the effectiveness of our algorithm, we compare our algorithm with the five algorithms mentioned above on several benchmark real data sets from UCI [46]. These data sets are often used in clustering or classification and the detailed information are shown in Table. 3, which are preprocessed to better serve the clustering algorithm. The performance shown in Table. 4 is bench-marked in terms of Acc and NMI, and the running time of these algorithms is shown in Table. 5.

Moreover, we use these parameters mentioned in their papers to conduct these experiments, including percent p in DPC, iteration invariant number *convits* and maximum

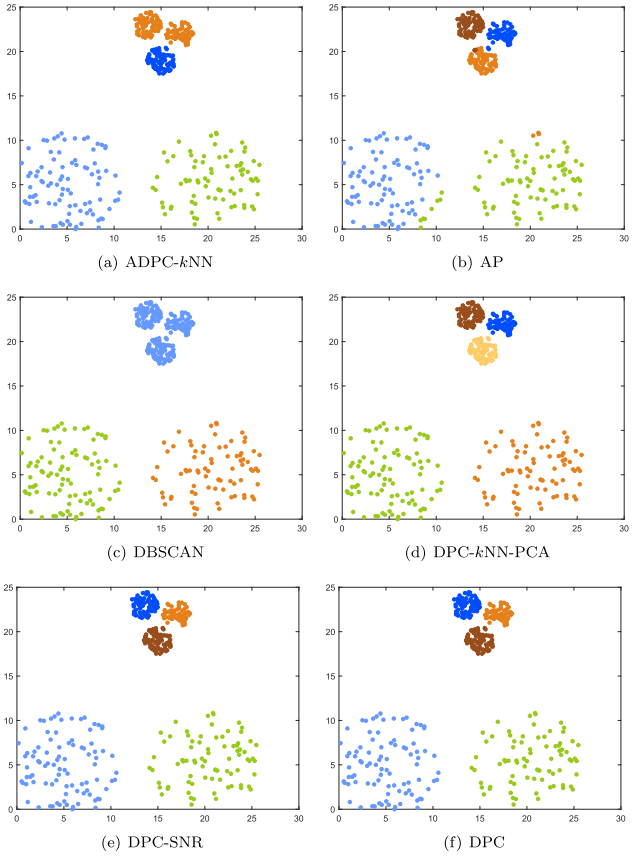


FIGURE 10. The clustering results of ADPC-*k*NN, AP, DBSCAN, DPC-*k*NN-PCA, DPC-SNR, and DPC algorithms on data set 5.

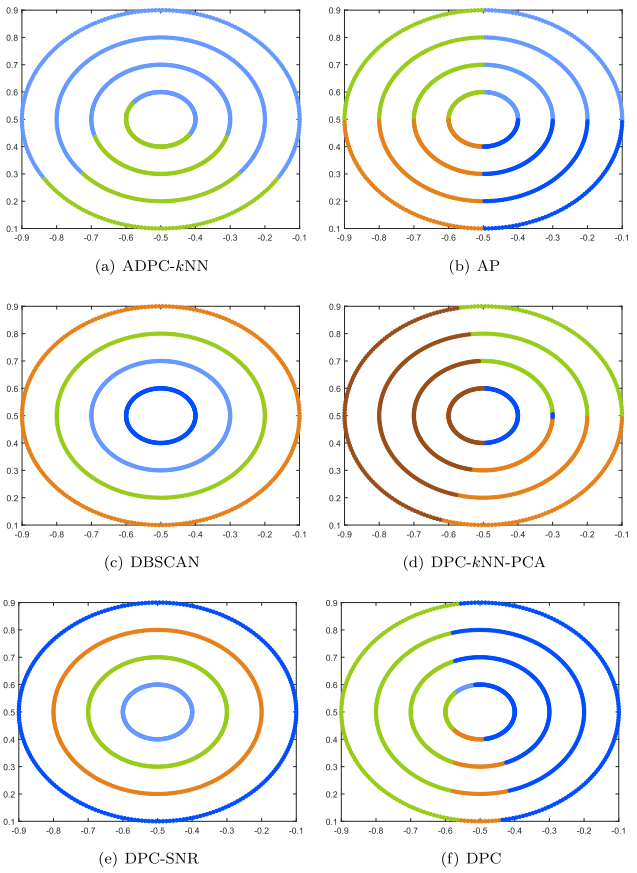


FIGURE 11. The clustering results of ADPC-*k*NN, AP, DBSCAN, DPC-*k*NN-PCA, DPC-SNR, and DPC algorithms on data set 6.

TABLE 3. Data characteristics of real data sets.

Dataset	Instances	Attributes	Clusters
Breast_C	699	9	2
Iris	150	4	3
Glass	214	9	6
Spect_H	267	22	2
Abalone	4177	8	3
Banknote_A	1372	4	2
Wisconsin_PBC	198	33	2
Chess	3196	36	2
Haberman_S	306	3	2
Spectf_H	267	22	2
Hayes_R	160	5	3
Ionosphere	351	34	2

number of iterations *maxits* in AP, the number of nearest neighbors k in ADPC-kNN, the number of nearest neighbors k and a percentage of the number of points m in DPC-kNN-PCA and the distance *Eps* and the number of points in the current distance *MinPts* in DBSCAN. Especially, we set that damping coefficient *lam* is 0.9 in AP throughout the experiments.

As shown in Table. 4, DPC-SNR outperforms other five algorithms on Breast_C, Spect_H, Banknote_A and Chess data sets in terms of benchmark Acc and NMI. We also can see that DPC-SNR outperforms DPC and DBSCAN on most of data sets, and has the same results as DPC on Abalone data set. DBSCAN can get higher values in terms of benchmark NMI on Haberman_S, Spectf_H and Hayes_R data sets. DPC-SNR can get higher values of benchmark Acc on Iris, Haberman_S, Spectf_H and Hayes_R data sets. ADPC-kNN gets the best results on Wisconsin_PBC and performs better on Iris and Spect_H in benchmark NMI. While ADPC-kNN performs worse on other data sets compared with DPC-SNR and cannot find the number of real clusters, neither can DBSCAN. The number of clusters of DPC and DPC-kNN-PCA is determined artificially, thus there is no problem of error recognition about the number of clusters. DPC-kNN-PCA gets better results on Glass data sets in benchmark NMI, while DPC-kNN-PCA performs worse on other data sets. AP algorithm mostly produces bad clustering results, since it is hard to control the number of clustering. However, AP sometimes can get the number of real clusters and high values of Acc and NMI. The bold data in each line is the best result. Here, we use $\gamma_i = \rho_i \delta_i$ to select n initial cluster centers. The running time of DPC-SNR and DPC-kNN-PCA in Table. 5 is more than other algorithms except AP, which is affected by the data dimension. Because the two algorithms will take a lot of time to find kNN and compute the distance between two

TABLE 4. Comparison of two benchmarks for six clustering algorithms on real data sets.

Algorithm	Par	Val	Cl	Acc	NMI	Algorithm	Par	Val	Cl	Acc	NMI
Breast_C						Iris					
DPC-SNR	k	15	2	**0.94**	**0.71**	DPC-SNR	k	10	3	**0.93**	0.77
DPC	p	2.5	2	0.66	0.11	DPC	p	3.5	3	0.83	0.72
DPC-kNN-PCA	k/m	15/139.8	2	0.47	0.05	DPC-kNN-PCA	k/m	7/30	3	0.90	0.76
ADPC-kNN	k	17	2	0.44	0.01	ADPC-kNN	k	17	3	0.91	**0.81**
DBSCAN	$Eps/MinPts$	25/8	2	0.52	0.00	DBSCAN	$Eps/MinPts$	0.7/10	3	0.65	0.72
AP	$convits/maxits$	5/32	20	0.65	0.53	AP	$convits/maxits$	5/27	3	0.90	0.80
Glass						Spect_H					
DPC-SNR	k	7	6	**0.48**	0.36	DPC-SNR	k	19	2	**0.76**	**0.06**
DPC	p	2.5	6	0.40	0.30	DPC	p	5.5	2	0.61	0.00
DPC-kNN-PCA	k/m	10/85.6	6	0.44	**0.40**	DPC-kNN-PCA	k/m	12/53.4	2	0.45	0.03
ADPC-kNN	k	6	6	0.37	0.38	ADPC-kNN	k	10	2	0.69	**0.06**
DBSCAN	$Eps/MinPts$	0.6/3	5	0.39	0.39	DBSCAN	$Eps/MinPts$	1.8/5	2	0.55	0.03
AP	$convits/maxits$	5/28	6	**0.48**	0.35	AP	$convits/maxits$	5/19	2	0.56	**0.07**
Abalone						Banknote_A					
DPC-SNR	k	634	3	**0.50**	**0.13**	DPC-SNR	k	45	2	**0.90**	**0.61**
DPC	p	4.6	3	**0.50**	**0.13**	DPC	p	3.4	2	0.74	0.34
DPC-kNN-PCA	k/m	7/1253.1	3	0.44	0.07	DPC-kNN-PCA	k/m	70/686	2	0.66	0.15
ADPC-kNN	k	121	2	0.49	0.15	ADPC-kNN	k	150	2	0.74	0.34
DBSCAN	$Eps/MinPts$	3.8/60	1	0.37	0.00	DBSCAN	$Eps/MinPts$	1.4/43	2	0.03	0.11
AP	$convits/maxits$	5/25	4	0.33	0.06	AP	$convits/maxits$	5/24	2	0.58	0.02
Wisconsin_PBC						Chess					
DPC-SNR	k	11	2	0.64	0.02	DPC-SNR	k	17	2	**0.51**	**0.03**
DPC	p	3.8	2	0.49	0.02	DPC	p	6.4	2	0.50	0.00
DPC-kNN-PCA	k/m	7/99	2	0.60	0.04	DPC-kNN-PCA	k/m	15/1917.6	2	0.48	0.00
ADPC-kNN	k	12	2	**0.78**	**0.11**	ADPC-kNN	k	28	2	0.50	0.00
DBSCAN	$Eps/MinPts$	100/3	2	0.61	0.02	DBSCAN	$Eps/MinPts$	1.7/11	2	0.49	0.01
AP	$convits/maxits$	25/65	12	0.15	0.06	AP	$convits/maxits$	5/17	4	0.35	0.02
Haberman_S						Spectf_H					
DPC-SNR	k	10	2	**0.64**	0.01	DPC-SNR	k	9	2	**0.72**	0.00
DPC	p	7.8	2	0.54	0.00	DPC	p	3.5	2	0.66	0.00
DPC-kNN-PCA	k/m	9/183.6	2	0.55	0.00	DPC-kNN-PCA	k/m	10/53.4	2	0.51	0.00
ADPC-kNN	k	11	3	0.46	0.01	ADPC-kNN	k	3	2	0.22	0.02
DBSCAN	$Eps/MinPts$	2.3/4	2	0.53	**0.04**	DBSCAN	$Eps/MinPts$	20/5	2	0.25	**0.04**
AP	$convits/maxits$	5/17	19	0.08	0.07	AP	$convits/maxits$	5/23	23	0.18	0.06
Hayes_R						Ionosphere					
DPC-SNR	k	6	3	**0.45**	0.02	DPC-SNR	k	12	2	0.54	**0.11**
DPC	p	2.1	3	0.36	0.01	DPC	p	1.2	2	0.49	0.07
DPC-kNN-PCA	k/m	7/105.6	3	0.35	0.01	DPC-kNN-PCA	k/m	24/70.2	2	**0.65**	0.04
ADPC-kNN	k	25	3	0.39	0.02	ADPC-kNN	k	8	2	0.54	0.07
DBSCAN	$Eps/MinPts$	2.4/4	3	0.07	**0.12**	DBSCAN	$Eps/MinPts$	0.3/4	2	0.07	0.10
AP	$convits/maxits$	5/33	3	0.36	0.02	AP	$convits/maxits$	5/50	26	0.11	0.29

TABLE 5. The running time(s) of the algorithms on real data sets.

Dataset	DPC-SNR	DPC	DPC-kNN-PCA	ADPC-kNN	DBSCAN	AP
Breast_C	1.01	0.64	0.81	0.46	0.42	0.86
Iris	0.32	0.13	0.26	0.20	0.17	0.22
Glass	0.39	0.19	0.33	0.30	0.22	0.34
Spect_H	0.45	0.16	0.32	0.31	0.24	0.53
Abalone	58.05	50.69	48.92	5.45	17.87	232.43
Banknote_A	3.16	2.98	2.80	0.40	1.60	7.93
Wisconsin_PBC	0.32	0.13	0.30	0.22	0.19	0.27
Chess	33.25	27.31	30.60	2.60	9.57	61.68
Haberman_S	0.45	0.19	0.35	0.21	0.25	0.49
Spectf_H	0.43	0.17	0.36	0.31	0.23	0.57
Hayes_R	0.24	0.19	0.26	0.20	0.16	0.19
Ionosphere	0.58	0.24	0.42	0.50	0.28	0.53

points twice. However, ADPC-kNN computes the distance once and saves k distances about kNN instead of k points, which benefits the process of obtaining the local density of each point.

V. CONCLUSIONS

Clustering has found tremendous applications in many fields, such as business intelligence [31]–[34], pattern recognition [35] and cloud computing [36]–[39]. This paper

proposed a new clustering algorithm that is robust to outliers. In particular, our algorithm employs reverse k-nearest neighbors to estimate the local densities of each data point, and clusters each point staring from the peaks among all points on the symmetric neighborhood graph. After that, tiny clusters including outliers are merged into larger clusters based on their mutual connectivity on the graph. Experiments on various artificial and real data sets demonstrated that the DPC-SNR can successfully identify cluster centers over these data regardless of their distributions and dimensionality. Thus, the DPC-SNR can outperform the original DPC, DPC-kNN-PCA, AP, ADPC-kNN and DBSCAN. Finally, as the efficiency of our method tends to depend on the selection of the parameter k, how to evaluate an optimal value for k is left for our future study.

ACKNOWLEDGMENT

The authors are grateful to the reviewers for the valuable suggestions and helpful comments.

REFERENCES

[1] A. Likas, N. Vlassis, and J. J. Verbeek, "The global k-means clustering algorithm," *Pattern Recognit.*, vol. 36, no. 2, pp. 451–461, Feb. 2003.

[2] I. Khan and Z. Luo, "Nonnegative matrix factorization based consensus for clusterings with a variable number of clusters," *IEEE Access*, vol. 6, pp. 73158–73169, 2018.

[3] R. T. Ng and J. Han, "CLARANS: A method for clustering objects for spatial data mining," *IEEE Trans. Knowl. Data Eng.*, vol. 14, no. 5, pp. 1003–1016, Sep./Oct. 2002.

[4] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. KDD*, vol. 96, Aug. 1996, pp. 226–231.

[5] A. Sharma and A. Sharma, "KNN-DBSCAN: Using k-nearest neighbor information for parameter-free density based clustering," in *Proc. Int. Conf. Intell. Comput., Instrum. Control Technol. (ICICICT)*, Jul. 2017, pp. 787–792.

[6] I. Khan, J. Z. Huang, and K. Ivanov, "Incremental density-based ensemble clustering over evolving data streams," *Neurocomputing*, vol. 191, pp. 34–43, May 2016.

[7] Y. Lv *et al.*, "An efficient and scalable density-based clustering algorithm for datasets with complex structures," *Neurocomputing*, vol. 171, pp. 9–22, Jan. 2016.

[8] P. Yang, Q. Zhu, and B. Huang, "Spectral clustering with density sensitive similarity function," *Knowl.-Based Syst.*, vol. 24, no. 5, pp. 621–628, 2011.

[9] H. Chang and D.-Y. Yeung, "Robust path-based spectral clustering," *Pattern Recognit.*, vol. 41, no. 1, pp. 191–203, 2008.

[10] I. Khan, J. Z. Huang, N. T. Tung, and G. Williams, "Ensemble clustering of high dimensional data with fastmap projection," in *Proc. Pacific–Asia Conf. Knowl. Discovery Data Mining*. Cham, Switzerland: Springer, May 2014, pp. 483–493.

[11] H. Liu, M. Shao, S. Li, and Y. Fu, "Infinite ensemble clustering," *Data Mining Knowl. Discovery*, vol. 32, no. 2, pp. 385–416, Mar. 2018.

[12] S. Theodoridis and K. Koutroumbas, "Clustering algorithms II: Hierarchical algorithms," *Pattern Recognit.*, 2009, pp. 653–700.

[13] G. Karypis, E.-H. Han, and V. Kumar, "Chameleon: Hierarchical clustering using dynamic modeling," *Computer*, vol. 32, no. 8, pp. 68–75, Aug. 1999.

[14] D. Cheng, Q. Zhu, J. Huang, Q. Wu, and L. Yang, "A novel cluster validity index based on local cores," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 4, pp. 985–999, Apr. 2018.

[15] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, no. 6191, pp. 1492–1496, Jun. 2014.

[16] M. A. Abbas and A. A. Shoukry, "CMUNE: A clustering using mutual nearest neighbors algorithm," in *Proc. 11th Int. Conf. Inf. Sci., Signal Process. Appl. (ISSPA)*, Jul. 2012, pp. 1192–1197.

[17] Z. Hu and R. Bhatnagar, "Clustering algorithm based on mutual K-nearest neighbor relationships," *Stat. Anal. Data Mining*, vol. 5, no. 2, pp. 100–113, Apr. 2012.

[18] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, Feb. 2007.

[19] G. Wang and Q. Song, "Automatic clustering via outward statistical testing on density metrics," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 8, pp. 1971–1985, Aug. 2016.

[20] J. Xie, H. Gao, W. Xie, X. Liu, and P. W. Grant, "Robust clustering by detecting density peaks and assigning points based on fuzzy weighted K-nearest neighbors," *Inf. Sci.*, vol. 354, pp. 19–40, Aug. 2016.

[21] M. Du, S. Ding, and H. Jia, "Study on density peaks clustering based on K-nearest neighbors and principal component analysis," *Knowl.-Based Syst.*, vol. 99, pp. 135–145, May 2016.

[22] D. Cheng, Q. Zhu, J. Huang, and L. Yang, "Natural neighbor-based clustering algorithm with density peeks," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2016, pp. 92–98.

[23] L. Yaohui, M. Zhengming, and Y. Fang, "Adaptive density peak clustering based on K-nearest neighbors with aggregating strategy," *Knowl.-Based Syst.*, vol. 133, pp. 208–220, Oct. 2017.

[24] Z. Li and Y. Tang, "Comparative density peaks clustering," *Expert Syst. Appl.*, vol. 95, pp. 236–247, Apr. 2018.

[25] J. M. Keller, M. R. Gray, and J. A. Givens, "A fuzzy k-nearest neighbor algorithm," *IEEE Trans. Syst., Man Cybern.*, vol. SMC-15, no. 4, pp. 580–585, Jul./Aug. 1985.

[26] W. Wu, J. Liu, H. Rong, H. Wang, and M. Xian, "Efficient k-nearest neighbor classification over semantically secure hybrid encrypted cloud database," *IEEE Access*, vol. 6, pp. 41771–41784, 2018.

[27] J. Huang, Q. Zhu, L. Yang, D. Cheng, and Q. Wu, "QCC: A novel clustering algorithm based on quasi-cluster centers," *Mach. Learn.*, vol. 106, no. 3, pp. 337–357, Mar. 2017.

[28] Y. Qin, Z. L. Yu, C.-D. Wang, Z. Gu, and Y. Li, "A novel clustering method based on hybrid K-nearest-neighbor graph," *Pattern Recognit.*, vol. 74, pp. 1–14, Feb. 2018.

[29] L. Ertöz, M. Steinbach, and V. Kumar, "Finding clusters of different sizes, shapes, and densities in noisy, high dimensional data," in *Proc. SIAM Int. Conf. Data Mining*, May 2003, pp. 47–58.

[30] W. Jin, A. K. H. Tung, J. Han, and W. Wang, "Ranking outliers using symmetric neighborhood relationship," in *Proc. Pacific–Asia Conf. Knowl. Discovery Data Mining*, vol. 6. Berlin, Germany: Springer, Apr. 2006, pp. 577–593.

[31] I. Khan, J. Z. Huang, M. A. Masud, and Q. Jiang, "Segmentation of factories on electricity consumption behaviors using load profile data," *IEEE Access*, vol. 4, pp. 8394–8406, 2016.

[32] X. Fu, K. Yue, L. Liu, Y. Feng, and L. Liu, "Reputation measurement for Online services based on dominance relationships," *IEEE Trans. Services Comput.*, to be published.

[33] G. Zou, Q. Lu, Y. Chen, R. Huang, Y. Xu, and Y. Xiang, "QoS-aware dynamic composition of Web services using numerical temporal planning," *IEEE Trans. Services Comput.*, vol. 7, no. 1, pp. 18–31, Jan./Mar. 2014.

[34] I. Khan, J. Z. Huang, Z. Luo, and M. A. Masud, "CPLP: An algorithm for tracking the changes of power consumption patterns in load profile data over time," *Inf. Sci.*, vol. 429, pp. 332–348, Mar. 2018.

[35] H. He and Y. Tan, "Automatic pattern recognition of ECG signals using entropy-based adaptive dimensionality reduction and clustering," *Appl. Soft Comput.*, vol. 55, pp. 238–252, Jun. 2017.

[36] L. Zhang, S. Wang, R. K. Wong, F. Yang, and R. N. Chang, "Cognitively adjusting imprecise user preferences for service selection," *IEEE Trans. Netw. Service Manage.*, vol. 14, no. 3, pp. 717–729, Sep. 2017.

[37] W. Gong, L. Qi, and Y. Xu, "Privacy-aware multidimensional mobile service quality prediction and recommendation in distributed fog environment," *Wireless Commun. Mobile Comput.*, vol. 2018, Apr. 2018, Art. no. 3075849.

[38] Y. Xu, L. Qi, W. Dou, and J. Yu, "Privacy-preserving and scalable service recommendation based on simhash in a distributed cloud environment," *Complexity*, vol. 2017, Dec. 2017, Art. no. 3437854.

[39] W. Li, K. Liao, Q. He, and Y. Xia, "Performance-aware cost-effective resource provisioning for future grid IoT-cloud system," *J. Energy Eng.*, 2019. doi: 10.1061/(ASCE)EY.1943-7897.0000611.

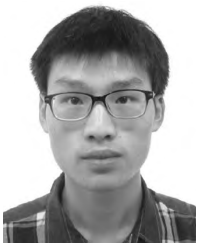
[40] A. Laio. *Clustering by Fast Search-and-Find of Density Peaks*. Accessed: Mar. 20, 2019. [Online]. Available: https://people.sissa.it/~laio/Research/Res_clustering.php

[41] A. Strehl and J. Ghosh, "Cluster ensembles—A knowledge reuse framework for combining multiple partitions," *J. Mach. Learn. Res.*, vol. 3, pp. 583–617, Dec. 2002.
[42] A. K. Jain and M. H. Law, "Data clustering: A user's dilemma," in *Proc. Int. Conf. Pattern Recognit. Mach. Intell.*. Berlin, Germany: Springer, Dec. 2005, pp. 1–10.
[43] C. T. Zahn, "Graph-theoretical methods for detecting and describing gestalt clusters," *IEEE Trans. Comput.*, vol. C-20, no. 1, pp. 68–86, Jan. 1971.
[44] C. Cassisi, A. Ferro, R. Giugno, G. Pigola, and A. Pulvirenti, "Enhancing density-based clustering: Parameter reduction and outlier detection," *Inf. Syst.*, vol. 38, no. 3, pp. 317–330, 2013.
[45] P. Fränti and S. Sieranoja, "*K*-means properties on six clustering benchmark datasets," *Appl. Intell.*, vol. 48, no. 12, pp. 4743–4759, Dec. 2018. Accessed: Mar. 20, 2019. [Online]. Available: http://cs.uef.fi/sipu/datasets/
[46] D. Dua and K. T. Efi. *UCI Machine Learning Repository*. Accessed: Mar. 20, 2019. [Online]. Available: http://archive.ics.uci.edu/ml

CHUNRONG WU received the B.S. degree in software engineering from Xinjiang University, Ürümqi, China, in 2016, and the M.S. degree in software engineering from Zhejiang University, Hangzhou, China, in 2018. She is currently pursuing the Ph.D. degree with Chongqing University, Chongqing, China. Her research interests include data mining and intelligent computing.

JIA LEE received the B.E., M.E., and Ph.D. degrees from Hiroshima University, Hiroshima, Japan, in 1996, 1998, and 2001, respectively. He is currently a Professor with the College of Computer Science, Chongqing University, China. His research interests include cellular automata, swarm intelligence, and asynchronous circuits.

TEIJIRO ISOKAWA received the B.E. degree in electronic engineering, the M.E. degree in electronic engineering, and the D.E. degree from the Himeji Institute of Technology, Japan, in 1996, 1999, and 2004, respectively. He is currently an Associate Professor with the Division of Computer Engineering, Graduate School of Engineering, University of Hyogo, Japan. His research interests include nanocomputing, hypercomplex-valued neural networks, and cognitive models in visual systems.



JUN YAO received the B.S. degree from the University of Shanghai for Science and Technology, Shanghai, China, in 2017. He is currently pursuing the master's degree with Chongqing University, Chongqing, China. His research interests include various aspects of theoretical computer science, especially in algorithms, complexity theory, and multi-agent systems.

YUNNI XIA (SM'14) received the B.S. degree in computer science from Chongqing University, China, in 2003, and the Ph.D. degree in computer science from Peking University, China, in 2008. Since 2008, he has been an Associate Professor with the School of Computer Science, Chongqing University. He has authored or coauthored over 50 research publications. His research interests include Petri nets, software quality, performance evaluation, and cloud computing system dependability.