Information Sciences 571 (2021) 375–400

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

ROBP a robust border-peeling clustering using Cauchy kernel

Mingjing Du *, Ru Wang, Ru Ji, Xia Wang, Yongquan Dong

School of Computer Science and Technology, Jiangsu Normal University, Xuzhou 221116, China

ARTICLE INFO

Article history:
Received 23 December 2019
Received in revised form 4 April 2021
Accepted 29 April 2021
Available online 4 May 2021

Keywords:
Density-based clustering
Border-peeling clustering
Cauchy kernel

ABSTRACT

Recently a novel density-based clustering algorithm, namely, border-peeling (BP) clustering algorithm, is proposed to group data by iteratively identifying border points and peeling off them until separable areas of data remain. The BP clustering is able to correctly recognize the true structure of clusters and automatically detect the outliers on several test cases. However, there are some drawbacks in BP, and these may hinder its widespread application. The BP clustering might yield bad results on datasets with non-uniformly-distributed clusters. Especially, the BP clustering tends to over-partition the data with complex shape. To overcome these defects, a robust border-peeling clustering algorithm (named as ROBP) is proposed in this paper. Our method improves the BP clustering algorithm from two aspects: density influence (i.e. density estimation) and linkage criterion (i.e. association strategy). In density estimation, we use Cauchy kernel with longer tails instead of Gaussian kernel in the local scaling function, and further propose a kernel density estimator, i.e., the density estimator based on Cauchy kernel. It can calculate quickly and accurately the density influence value of each point. In association strategy, we design a linkage criterion based on the shared neighborhood information. The linkage criterion can create some links between peeled border points and their neighboring peeled border points, in order to avoid over-segmentation of the clusters. We integrate the proposed linkage criterion and the uni-directional association strategy, and further propose a bi-directional association strategy. In experiments, we compare ROBP with 7 representative density-based clustering (or hierarchical clustering) algorithms, including BP, DBSCAN, HDBSCAN, density peak (DP) clustering, DPC-KNN, DPC-DBFN and McDPC, on 8 synthetic datasets and 11 real-world datasets. Results show that the proposed algorithm outperforms 7 competitors in most cases. Moreover, we compare the robustness of ROBP and BP, and evaluate their running time. Experimental results indicate that ROBP is much more robust and reliable, as well as it is competitive to BP in computational efficiency.

1. Introduction

Clustering is one of the most important and useful techniques in data mining and knowledge discovery, and can be used to automatically discover potentially, interesting, and novel patterns from unlabeled data. As an unsupervised learning tool, clustering aims to classify objects into several groups, or clusters, in such a way that objects in the same cluster are as similar as possible, and objects in different clusters are as distinct as possible. Various clustering algorithms have a broad range of applications, ranging from natural language processing [14,16,17] to image segmentation [18,48].

* Corresponding author.
E-mail address: dumj@jsnu.edu.cn (M. Du).

https://doi.org/10.1016/j.ins.2021.04.089
0020-0255/

From a procedural point of view, these clustering algorithms can be roughly categorized into two groups: partitional clustering algorithms and connectivity-based clustering algorithms. Partitional clustering algorithms such as *K*-means and its variations, aim to discover the clusters that may exist in the data set by optimizing a specific objective function and iteratively improving the quality of the partitions. Some special forms of model-based algorithms can be simplified to partitional algorithms. For example, the Expectation-Maximization (EM) algorithm for Gaussian mixture model (GMM) can be proved to be equivalent to *K*-means algorithm under certain conditions. Due to their simplicity and ease of implementation, partitional clustering algorithms are often desirable in a wide variety of scenarios. However, their objective functions (e.g., sum-of-squared errors in *K*-means) typically result in clusters of convex (or spherical) shape.

In contrast, connectivity-based clustering algorithms can discover intrinsic groups from the data set based on their connectivity (e.g., pairwise distances between the objects, or adjacencies among the nodes). Density-based clustering [4], as an important connectivity-based approach, is capable of discovering complex shaped clusters, where clusters are assumed to be connected high-density areas separated by low-density areas. Based on this assumption, most of density-based clustering algorithms try to find one or many high-density points (or density peaks) at first, and then classify these points as separate clusters by performing a region query (or allocation) strategy.

A new density-based clustering algorithm, Border-Peeling (BP) clustering [3], is proposed recently. Different from traditional density-based clustering algorithms, the algorithm peels off the border points, which ensures that the cores of nearby clusters are clearly separated before the data points are classified as separate clusters. BP clustering can automatically discover complex shaped clusters in the setting where the number of clusters is not provided. Although BP clustering method exhibits powerful modeling capabilities, it has some shortcomings.

The local scaling function f is defined as an exponential function form (or so-called Gaussian kernel). Although definition of the function f is very simple, the computation of the exponential function is not so simple. Additionally, the density influence value and function f based on the Gaussian kernel are preferable to coping with the data that is of uniformly-distributed clusters. As a result, the BP clustering might yield bad results on datasets with non-uniformly-distributed clusters.

The association strategy focuses solely on connections between the peeled border points and the remaining core points rather than the peeled border points' connections between one another. Therefore, the BP clustering algorithm has a poor capability of tackling some elongated clusters.

The BP clustering claims to be insensitive to its parameters setting. However, the input parameter k (i.e., the number of neighbors) is difficult to determine, especially, when BP clustering deals with complex shaped clusters.

Aiming to address the above challenges, we propose a RObust Border-Peeling (ROBP) clustering algorithm. In order to avoid over-partitioning, we improve BP clustering algorithm in two aspects: density influence (i.e. density estimation) and linkage criterion (i.e. association strategy). In density estimation, inspired by t-SNE [42], we design a density influence, which is not only more efficient, but also can significantly reduce the negative impacts of the data with non-uniform distributions. In addition, we abandon the linkage criterion of BP. Instead, we use bi-directional association strategy to compute clusters iteratively. In summary, the main contributions of our work include the following:

Based on the Cauchy distribution, we design a Cauchy kernel with longer tails. Compared to the Gaussian kernel, it is more efficient, and less sensitive to the data with non-uniform distributions.

We use it to redesign a local scaling function. Additionally, the density influence is redefined based on Cauchy kernel. This criterion is instrumental in improving the performance on complex datasets that are of non-uniform distributions.

We use the concept of shared nearest neighbors to design a linkage criterion, which is able to link peeled border points.

We further propose a bi-directional association strategy based on the proposed linkage criterion and the original association strategy, which can avoid over-partitioning.

The remainder of this paper is organized as follows. In Section 2, we introduce the related work. In Section 3, we describe the principle of the BP clustering and reveal some problems. We make a detailed description of ROBP in Section 4. Section 5 presents our experimental results on synthetic data sets and real data sets. Finally, concluding remarks and directions of future works are presented in Section 6.

2. Related work

Discovering natural clusters with complex shape is still an active field in cluster analysis research. Traditional partitional clustering algorithms such as K-means can only detect clusters with convex distribution. One option is to use multiple prototypes to model each class of data. A notable technique in multiple prototypes representation literature is K-Multiple-Means method (KMM) [31]. Another line of research focuses on spectral clustering, such as JSESR[32], GDBNSC-Ncut[40], NSSRD[39]. Furthermore, in order to deal with high dimensional data with non-convex distribution, many efficient dimensionality reduction algorithms, such as non-negative matrix factorization (NMF) or subspace learning, attract a lot of attention in the community of data scientists. These methods, such as SODNMF [30] and LDSSL [38], map the data into a low dimensional space for a better partition.

On the other side, there are a lot of connectivity-based clustering algorithms that are able to discover natural clusters with complex shape. These clustering methods can be categorized into hierarchical methods and density-based methods. This section introduces the most closely related works concerning the proposed algorithm.

The hierarchical algorithm based on single-linkage is used to find nonlinear pattern structures [41,46]. Due to so-called "chaining effect" [21], which may merge clusters that, in practice, "should be detected" by the algorithm as independent clusters, the single linkage criterion is prone to merge multiple different clusters into one cluster.

To avoid the chaining effect, a density-based clustering algorithm is proposed. The method first tries to remove noise objects (causing the chaining-effect), and then employs a single-linkage manner to cluster the remaining points.

DBSCAN [10] is one of the most important density-based techniques. Due to the use of index structures for density estimation, it claims to be scalable to large databases. Conceptually, DBSCAN makes the following assumption: clusters are regarded as connected, high-density regions separated by low-density regions. To define a density level, two parameters, including a distance threshold r (or so-called ε) and a density threshold k (or so-called *minPts*), are required to be given in advance. According to this criterion, DBSCAN is able to define three different types of points. A point is called core point if it has a dense neighborhood, i.e., the number of points in its neighborhood with radius r (or ε) has to exceed the threshold k (or *minPts*). A point is called border point if it belongs to a cluster, but its neighborhood is not dense. A point is called noise point, if it does not belong to any cluster. These definitions based on ε- neighborhood forms a transitive association between non-noise points (i.e., core points or border points), and further discover density-based clusters. However, in practical terms, it is difficult to tune its input parameters. There are several studies which try to improve DBSCAN. EWOA–DBSCAN [34] tries to use the earthworm optimization algorithm [44], as a metaheuristic algorithm, to estimate the exact values of the parameters of DBSCAN and achieves significant improvements. In order to be more suitable for large scale data, many researches put forward strategies to improve DBSCAN. Li [23] proposes a new version of DBSCAN. It is based on the nearest neighbor similarity and fast parallel nearest neighbor search, where the former reduces the time of the distance computation, and the later improves the process of searching neighbors. Lin et al. introduce an improved algorithm called F-DBSCAN [24], which takes no account of distant neighbors of an object and computes only the distances between the object and its nearby neighbors. Thus, the computational cost is obviously reduced.

Additionally, if clusters in different regions have very different local densities, it is difficult that DBSCAN yields good results. To address the challenge, OPTICS [2] and HDBSCAN [5], two hierarchical density-based clustering algorithms, are proposed successively. Both algorithms construct a minimum spanning tree (MST) of the data based on their respective definitions of the reachability distance. One of the main differences between OPTICS and HDBSCAN is the definition of the density estimation. OPTICS still follows DBSCAN's density estimation that is based on ε- neighborhood. In HDBSCAN, the density level is defined according to a k-nearest-neighbor (KNN), i.e., the fixed value threshold is the number of neighbors k (or *minPts*) rather than the radius r (or ε).

Chameleon [22] also uses the idea of KNN. Different from HDBSCAN, Chameleon is a hierarchical clustering algorithm. It first uses a KNN graph to generate several relatively small sub-clusters and then creates clusters by merging the sub-clusters with high similarity. Both Relative Interconnectivity (RI)[1] and Relative Closeness (RC)[2] of clusters determine the similarity of clusters.

Ertöz et al. [9] propose a clustering algorithm based on shared-nearest-neighbors (SNN) graph. The SNN similarity is defined as the number of shared neighbors of two points. A main difference between their approach and ours is that the SNN clustering uses the SNN similarity to redefine some definitions in DBSCAN, but our method utilizes the SNN similarity as a linkage criterion.

Recently, several works [11,13,19,33] using like-KNN graph are proposed. CMUNE algorithm [1] utilizes Mutual k-nearest-neighbor (MKNN) graph to calculate the density of each point.

Density peaks (DP) clustering [36], a density-based algorithm, is proposed by Rodriguez and Laio. Unlike traditional density-based clustering methods, the algorithm can be considered as a combination of the density-based clustering algorithm and the centroid-based clustering algorithm. It starts by determining the centroids of clusters according to two important quantities: ρ and δ. The second step is to determine which objects to merge in a cluster. There are several shortcomings in density peaks clustering algorithm. In order to overcome these difficulties, some improvements over DP, such as, DPC-KNN [8], FKNN-DPC [47], SNN-DPC [25], DPC-DBFN [27], McDPC [45] and so on [6], using various nearest-neighbor graph are proposed. However, unlike our method, the BP clustering and these above methods using KNN and its variations mostly adopt a density estimator based on Gaussian kernel.

3. Preparation

3.1. Notations

Let $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$denote a dataset of n data points, where for each $i,\ 1 \leqslant i \leqslant n, \boldsymbol{x}_i = [x_{i,1}, \ldots, x_{i,m}]$ with m attributes. We use $d(\boldsymbol{x}_i, \boldsymbol{x}_j)$ to represent Euclidean distance between two points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, as follows:

[1] RI can be viewed as the sum of the similarities among pairs of points belonging to different clusters.
[2] RC can be viewed as the average of the similarities among pairs of points belonging to different clusters.

Table 1
Symbols and notations in BP and ROBP.

Symbol	Description
n	The number of objects in the dataset
m	The number of attributes in each object
k	The number of nearest neighbors
t	The t^{th} iteration in the peeling process
$\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$	The whole data set.
$\boldsymbol{X}^{(t)}$	The set of remain unpeeled points at the t^{th} iteration
$\boldsymbol{X}_B^{(t)}$	The set of peeled points at the t^{th} iteration
$d(\cdot,\cdot)$	Euclidean distance
$N_k^{(t)}(\boldsymbol{x}_i)$	The k nearest neighbors set of $\boldsymbol{x}_i$ with respect to$d(\cdot,\cdot)$
$RN_k^{(t)}(\boldsymbol{x}_i)$	The reverse k nearest neighbors set of $\boldsymbol{x}_i$ with respect to$d(\cdot,\cdot)$
$b_i^{(t)}$	The density influence value of$\boldsymbol{x}_i$
l_i	The spatially-variant threshold value of$\boldsymbol{x}_i$

$$d(\boldsymbol{x}_i, \boldsymbol{x}_j) = \| \boldsymbol{x}_i - \boldsymbol{x}_j \|_2 \tag{1}$$

$\boldsymbol{X}^{(t)}$ and $\boldsymbol{X}_B^{(t)}$ denote a set of remain unpeeled points and the set of peeled points at t^{th} iteration, respectively. The set of unpeeled data points for the next iteration is given by

$$\boldsymbol{X}^{(t+1)} = \boldsymbol{X}^{(t)} \; \boldsymbol{X}_B^{(t)} \tag{2}$$

$RN_k^{(t)}(\boldsymbol{x}_i)$ denotes the set of the reverse k nearest neighbors (RKNN) of a point $\boldsymbol{x}_i$ in $\boldsymbol{X}^{(t)}$, as follows:

$$RN_k^{(t)}(\boldsymbol{x}_i) = \left\{ \boldsymbol{x}_j \middle| \boldsymbol{x}_i \in N_k^{(t)}(\boldsymbol{x}_j) \right\} \tag{3}$$

where $N_k^{(t)}(\boldsymbol{x}_j)$ denotes the set of k nearest neighbors of $\boldsymbol{x}_j$. For each point $\boldsymbol{x}_i \in \boldsymbol{X}^{(t)}$, $b_i^{(t)}$ and l_i denotes its density influence value and its spatially-variant threshold value, respectively.

For sake of clarity, the above-mentioned symbols and notations are listed in Table 1.

3.2. Border-peeling clustering

Different from traditional density-based clustering algorithms, BP finds core points of latent clusters by using a border-peeling strategy that iteratively peels off layers of points in order to ensure that contiguous regions of core points are clearly separated. Based on the idea, some important concepts are defined.

Definition 1 (*(Density influence value).*). *In the BP method, the local density level is called the density influence value. The density influence value $b_i^{(t)}$ of a point $\boldsymbol{x}_i$ is defined as the sum of the similarity between $\boldsymbol{x}_i$ and its reverse k nearest neighbors, as followed*

$$b_i^{(t)} = \sum_{\boldsymbol{x}_j \in RN_k^{(t)}(\boldsymbol{x}_i)} f(\boldsymbol{x}_i, \boldsymbol{x}_j) \tag{4}$$

where local scaling function $f(\cdot,\cdot)$ is a Gaussian kernel is given by

$$f(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(- \frac{\| \boldsymbol{x}_i - \boldsymbol{x}_j \|_2^2}{\sigma_j^2} \right) \tag{5}$$

where σ_j is set to the distance from the point $\boldsymbol{x}_j$ to its k^{th} nearest neighbor at iteration t, as followed

$$\sigma_j = \| \boldsymbol{x}_j - N_k^{(t)}(\boldsymbol{x}_j)[k] \|_2 \tag{6}$$

where $N_k^{(t)}(\boldsymbol{x}_j)[k]$ represents the k^{th} nearest neighbor of $\boldsymbol{x}_j$at iteration t.

Definition 2 (*(Peeled point).*). *At t^{th} iteration, a point $\boldsymbol{x}_i$ is called as a peeled point, if its density influence value is no more than the cut-off value of the iteration.*

Thus, the set of peeled points $\boldsymbol{X}_B^{(t)}$ at each iteration satisfies the following relationship:

$$\boldsymbol{X}_B^{(t)} = \left\{ \boldsymbol{x}_i \middle| b_i^{(t)} \leqslant \tau^{(t)} \right\} \tag{7}$$

where $\tau^{(t)}$ is set such that 90% of the remaining data points at each iteration have larger values of $b_i^{(t)}$.

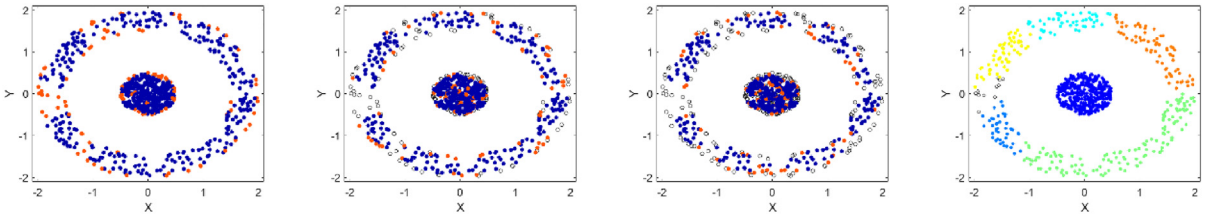

Fig. 1. Border-peeling technique on the Target dataset.

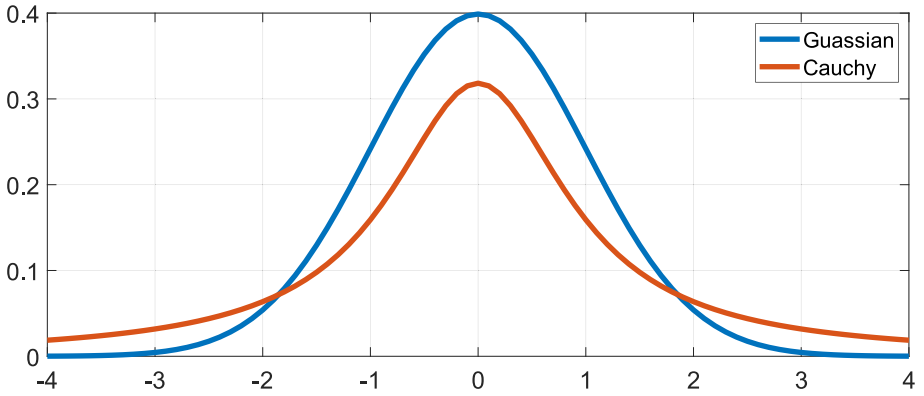

Fig. 2. Cauchy distribution and Gaussian distribution.

Definition 3 ((*Core point).*). *A point* $\boldsymbol{x}_i$ *is called as a core point, if it is one of the remaining points after the peeling process.*

In order to assign each peeled point to a unique cluster, during the peeling process, the method creates associations between the peeled points and remaining points at each iteration. Thus, the transitive association yields paths from each peeled border point to one of the core points.

To avoid the chaining effect, BP marks some points as noise points.

Definition 4 ((*Noise point).*). *A point* $\boldsymbol{x}_i$ *is called noise point, if there are no non-peeled points* (*i.e., remaining points at iteration t*) *within a distance of* l_i*from* $\boldsymbol{x}_i$ *at iteration t, as followed*:

$$\left\{\boldsymbol{x}_j \middle| \| \boldsymbol{x}_i - \boldsymbol{x}_j \|_2 \leqslant l_i \text{ and } \boldsymbol{x}_j \in \boldsymbol{X}^{(t+1)}\right\} = \varnothing \tag{8}$$

where l_i is a spatially-variant threshold. It is worth noting that these thresholds are not specified individually for each data point by the user. In first iteration, l_i is set to $mean(D_k) + std\ (D_k)$, where D_k is all of the pairwise distances in the k-neighborhood of each point. For the subsequent iterations, l_i is set to $\min\left\{\frac{3}{k}\sum_{\boldsymbol{x}_j \in NN_{B,k}^{(t)}} ||\boldsymbol{x}_i - \boldsymbol{x}_j||_2, mean(D_k) + std(D_k)\right\}$, where $NN_{B,k}^{(t)}$ denotes the set of points that were already peeled up to the current iteration and were not marked as outliers.

The number of iterations of the peeling process T can be set automatically. The termination condition of iterations of the peeling process is based on an important observation: peeled border points should reside in denser areas than the border points of the previous iteration. Hence, in each iteration t, the BP clustering tracks the set of values of border points that are about to be peeled: $\left\{b_i^{(t)} | \boldsymbol{x}_i \in \boldsymbol{X}_B^{(t)}\right\}$, and calculate the mean value of that set, denoted by $\bar{b}_p^{(t)}$. The termination condition of iterations of the peeling process is $\frac{\bar{b}_p^{(t)}}{\bar{b}_p^{(t-1)}} - \frac{\bar{b}_p^{(t-1)}}{\bar{b}_p^{(t-2)}} > 0.15$.

For simplicity, we call the association strategy that creates associations between the peeled points and remaining points as the uni-directional association strategy.

Definition 5 ((*Uni-directional association strategy).*). *If there are one or more non-peeled points within a distance of* l_i *from* $\boldsymbol{x}_i$*, then point* $\boldsymbol{x}_i$ *is associated with the closest non-peeled point to* $\boldsymbol{x}_j$.

The association strategy is essentially an allocation strategy. It forms a transitive association between the border points with one of the core points.

Algorithmically, BP iteratively identifies border points by adopting the border-peeling formulation and associates them to unpeeled data points in inner layers. To prevent the erroneous merging, BP sets a spatially-variant threshold l_i so that noise points are removed before computing associations. Once the border peeling process terminates, the remaining core points are then grouped into clusters using a simplified version of DBSCAN.

3.3. Analysis

Although the BP clustering yields high-performance results in many contexts, there still exist some obvious drawbacks.

3.3.1. Density influence and local scaling function

Note that local scaling function $f(\cdot,\cdot)$ using a Gaussian kernel. For sake of clarity, we give a definition of the Gaussian kernel. Given two points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, the Gaussian kernel is defined as:

$$\kappa_G(\boldsymbol{x}_i,\boldsymbol{x}_j) = \exp\left(-\frac{\| \boldsymbol{x}_i - \boldsymbol{x}_j \|_2^2}{\sigma^2}\right) \tag{9}$$

where σ is an adjustable parameter that is commonly given by the users.

It is noticeable that Gaussian kernel is a form of the exponential function. Most commonly, the exponential function is defined by the following Taylor series:

$$\exp(x) = \exp(a)\sum_{k=0}^{\infty}\frac{(x-a)^k}{k!} = \exp(a)\left[1 + (x-a) + \frac{(x-a)^2}{2} + \cdots\right] \tag{10}$$

There are various approximation techniques of the exponential function [37]. Although these approximation techniques can reduce the overall computation time, the exponential function still is not a simple, low-level function.

More importantly, the Gaussian kernel that is of the form of a bell-shaped curve is essentially the similarity as an exponential function of the squared distance between the points. The similarity based on Gaussian kernel decreases rapidly and is infinitely close to zero, as two points grow farther apart from each other. The rapid decay property of Gaussian kernel means that the points away from the center actually contribute very little to the result. When there are considerably different local densities in the same cluster, Gaussian kernel is prone to ignore the contributions of some faraway neighbors. As a result, a density estimator based on Gaussian kernel is more appropriate to handle data with relatively uniform distributions (or relatively uniform densities).

3.3.2. Association strategy

BP has difficulties in finding the optimal partition structures, especially for the datasets containing elongated clusters with relatively non-uniform densities. Specifically, Fig. 1 shows the peeling process and the clustering results of the BP technique on the Target dataset where a spherical cluster is surrounded by a ring-shaped cluster. Fig. 1(a)-(c) present three consecutive peeling iterations (the identified border points and the remaining points at the t^{th} iteration are colored in orange and in blue, and black bubbles (or circles) represent the peeled points in the previous iterations). Fig. 1 (d) shows the final clustering result obtained by BP (black bubbles (or circles) represent outliers).

This figure clearly shows that the BP clustering algorithm is not able to correctly find the optimal partition structures on the dataset. We suspect that its association strategy has a greater contribution to the problem. If two nearby peeled border points are linked to two different regions, then the following peeled border points in the next iteration may also be assigned to different regions. As the peeling process continues, the gaps generated by the inappropriate peeling are gradually widened, and these gaps can separate points belonging to the same cluster into different regions. In this case, BP is not able to merge these different regions into one cluster. This is because the BP clustering algorithm does not take the links between the peeled border points into account. When the nearby peeled border points are assigned to different regions, the subsequent peeling and association of the remaining points will be misled, resulting in a large partitioning error. Generally, this association strategy is still prone to yield over-partitioning.

4. ROBP algorithm

This section presents a detailed description of our proposed clustering algorithm. To describe the algorithm, we start with some concepts and definitions. Then, we propose the improved ROBP clustering method. Finally, we present the computational complexity analysis of ROBP.

4.1. Definitions for ROBP

Now, we give definitions on the fundamental concepts.

Table 2
Parameter settings.

Method	Parameters
BOBP	$k \in (2:1:25)$
BP	$k \in (2:1:25)$
DBSCAN	$\varepsilon \in (0.1:0.25:5.1)andk \in (2:1:25)$
HDBSCAN	$k \in (2:1:25)$and$N_c \in \{5,\ 10,\ 15,\ 20\}$
DP	$d_c \in (1\%:1\%:10\%)$and$\alpha \in (1:0.5:5)$
DPC-KNN	$k \in (2:1:25)$and$\alpha \in (1:0.5:5)$
DPC-DBFN	$k \in (2:1:25)$and$\alpha \in (1:0.5:5)$
McDPC	$d_c \in (1\%:1\%:10\%)$, $\gamma \in \{0.1, 0.2, 0.3\}$ and$\theta, \lambda \in \{1, 2, 3\}$

Table 3
Synthetic datasets.

Data Sets	Class#	Dimension#	Number#
Diffdensity	4	2	863
Jain	2	2	373
Target	2	2	758
Twospirals	2	2	378
Data5	4	2	1336
DS5	7	2	972
T8	8	2	7677
T7	9	2	9208

4.1.1. Cauchy kernel

Inspired by t-SNE [42], we use Cauchy kernel, i.e., a kernel based on the Cauchy distribution, as follows:

$$\kappa_C(\boldsymbol{x}_i, \boldsymbol{x}_j) = \left(\frac{\| \boldsymbol{x}_i - \boldsymbol{x}_j \|_2^2}{\sigma^2} + 1 \right)^{-1} \tag{11}$$

From a computationally convenient view, it is much faster to evaluate the similarity between points under a Cauchy distribution than under a Gaussian distribution, because Eq. (11) does not involve an exponential form. From an algorithmically robust view, the Cauchy kernel is much more robust against the datasets with non-uniformly-distributed clusters than the Gaussian kernel, since a Cauchy distribution has longer tails than a Gaussian distribution, as seen in Fig. 2.

We redefine the local scaling function and the density influence based on Cauchy kernel.

Definition 6. (*Local scaling function based on Cauchy kernel*). *According to Cauchy kernel, we present a local scaling function.*

Given two points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, the local scaling function based on Cauchy kernel is defined as

$$f_C(\boldsymbol{x}_i, \boldsymbol{x}_j) = \left(\frac{\| \boldsymbol{x}_i - \boldsymbol{x}_j \|_2^2}{\sigma_j^2} + 1 \right)^{-1} \tag{12}$$

where the choice of σ_j follows Eq. (6).

Next, we use the proposed local scaling function $f_C(\cdot,\cdot)$ to redefine the density influence of each point in $\boldsymbol{X}^{(t)}$.

Definition 7. (*Density influence based on Cauchy kernel*). *For any a point $\boldsymbol{x}_i$ in $\boldsymbol{X}^{(t)}$, the density influence based on Cauchy kernel $\hat{b}_i^{(t)}$ is defined as the sum of the proposed local scaling values f_C of its reverse k nearest neighbors. The density influence based on Cauchy kernel is approximately the same as Eq. (4), as followed*:

$$\hat{b}_i^{(t)} = \sum_{\boldsymbol{x}_j \in RN_k^{(t)}(\boldsymbol{x}_i)} f_C(\boldsymbol{x}_i, \boldsymbol{x}_j) \tag{13}$$

4.1.2. Shared nearest neighbors

Although the proposed density influence can improve the performance of the BP clustering method to a certain extent, it cannot correctly discover the optimal partition structures on complex datasets with clusters that are of various densities. The major reason for this behavior is that BP does not take the associations between the peeled border points into account. To cope with the problem, a simple, intuitive linkage criterion is that two closest peeled border points are linked together, in terms the Euclidean distance between them. For convenience, the above linkage criterion is called linkage criterion *A*. However, linkage criterion *A* based on the simple idea causes the chaining effect, and further leads to the undesirable connectivity

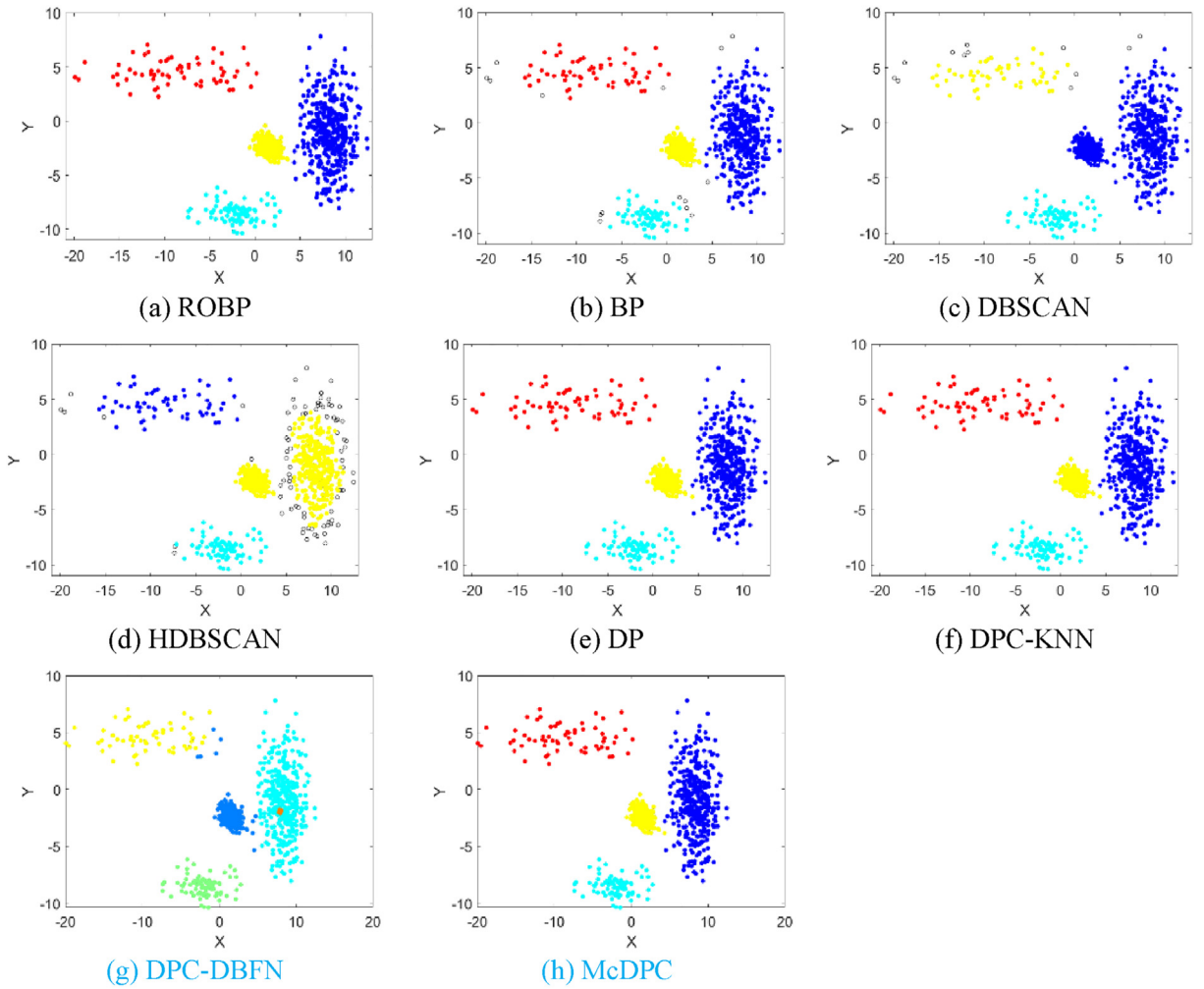

Fig. 3. Clustering results on the Diffdensity dataset.

of different clusters. In other words, BP based on linkage criterion A is prone to yield undesirable large clusters especially in presence of noise.

To avoid to the above problem, we introduce the concept of shared nearest neighbors (SNN) to design an effective linkage criterion. To describe the proposed linkage criterion, we firstly give several definitions.

Definition 8 (*(Shared nearest neighbors)*). [9]. *For any points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, the shared nearest neighbors of two points are the intersection of the sets of the k nearest neighbors of two points, as followed*:

$$SNN_k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \{\boldsymbol{x}_l | \boldsymbol{x}_l \in N_k(\boldsymbol{x}_i) \cap N_k(\boldsymbol{x}_j)\} \tag{14}$$

where $N_k(\boldsymbol{x}_i)$ denotes the set of k nearest neighbors of $\boldsymbol{x}_i$ and $N_k(\boldsymbol{x}_j)$ denotes the set of k nearest neighbors of $\boldsymbol{x}_j$.

Definition 9 (*(SNN similarity)*). [9]. *The SNN similarity of two points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, is defined as the cardinality of the shared nearest neighbors of two points, as followed*:

$$sim_{SNN}(\boldsymbol{x}_i, \boldsymbol{x}_j) = |SNN_k(\boldsymbol{x}_i, \boldsymbol{x}_j)| \tag{15}$$

Definition 10 (*(Current border-point set.)*. *The set of the data points is called as a current border-point set, if the points in the set have been already peeled up to the current iteration but not marked as noise points* (*see* Definition 4). *The current border-point set $\boldsymbol{X}_{CBS}^{(t)}$ is given by*

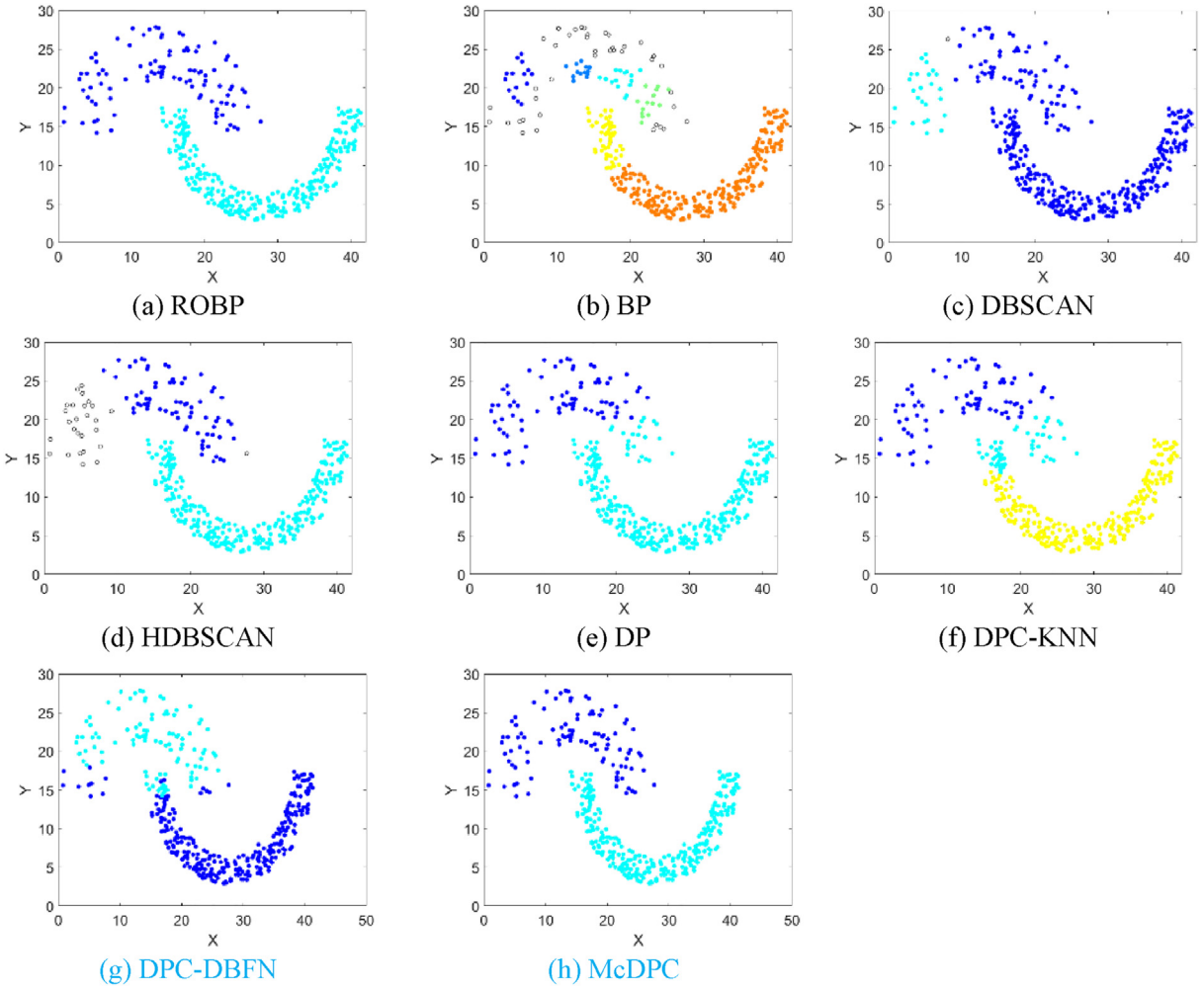

Fig. 4. Clustering results on the Jain dataset.

$$\boldsymbol{X}_{CBS}^{(t)} = \left\{ \boldsymbol{x}_i \middle| \boldsymbol{x}_i \in \cup_{r=1}^{t} \boldsymbol{X}_B^{(t)} \right\} = \left\{ \boldsymbol{x}_i \middle| \boldsymbol{x}_i \in \boldsymbol{X}_B^{(1)} \cup \boldsymbol{X}_B^{(2)} \cup \cdots \cup \boldsymbol{X}_B^{(t)} \right\} \tag{16}$$

Definition 11 ((*Neighbouring border points*).). *For any one of the border points* $\boldsymbol{x}_i \in \boldsymbol{X}^{(t)}$, *the set of neighbouring border points of* $\boldsymbol{x}_i$ *is the intersection of the sets of the k nearest neighbors of* $\boldsymbol{x}_i$ *and the current border-point set* (*see* Definition 10). *The set of neighbouring border points of* $\boldsymbol{x}_i$ *is denoted by* $N_{CBS,k}^{(t)}(\boldsymbol{x}_i)$, *as followed*:

$$N_{CBS,k}^{(t)}(\boldsymbol{x}_i) = \left\{ \boldsymbol{x}_j \middle| \boldsymbol{x}_j \in \left(N_k(\boldsymbol{x}_i) \cap \boldsymbol{X}_{CBS}^{(t)} \right) \right\} \tag{17}$$

Definition 12 ((*Linkage criterion B*).). *For any a point* $\boldsymbol{x}_i \in \boldsymbol{X}_B^{(t)}$, *we create a linkage between two points, only if the SNN similarity between the point* $\boldsymbol{x}_i$ *and its neighbouring border point is greater than or equal to* $k/2$ [25]. *In other words, there is a linkage between the point* $\boldsymbol{x}_i$ *and its neighbouring border point, only if at least half of their respective k neighborhoods are their common neighborhoods, i.e., satisfying*:

$$sim_{SNN}(\boldsymbol{x}_i, \boldsymbol{x}_j) \geqslant \frac{k}{2}, \text{ where } \boldsymbol{x}_i \in \boldsymbol{X}_B^{(t)} \text{ and } \boldsymbol{x}_j \in N_{CBS,k}^{(t)}(\boldsymbol{x}_i) \tag{18}$$

Otherwise, the linkage is not created.

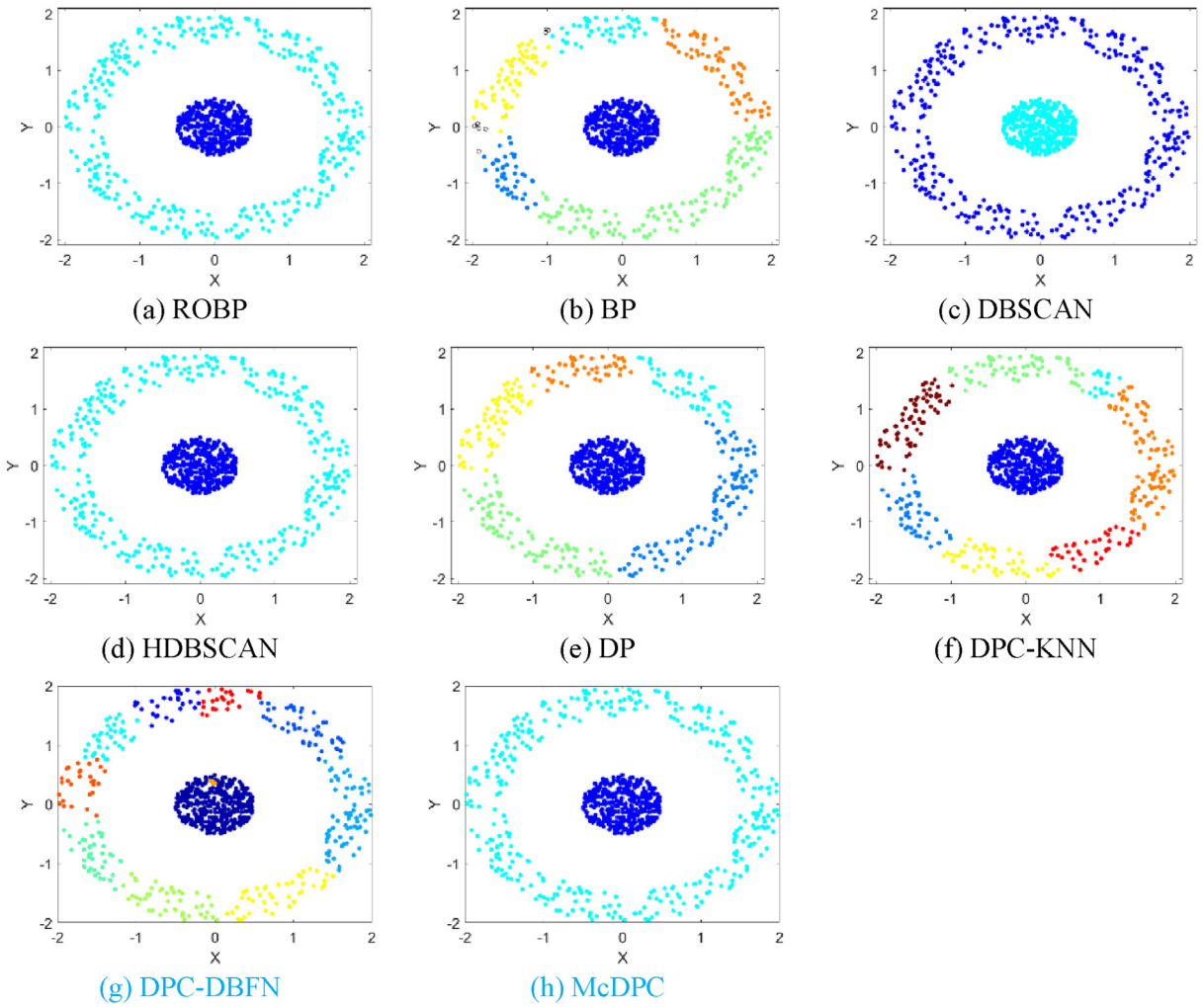

Fig. 5. Clustering results on the Target dataset.

On the basis of the uni-directional association strategy (see Definition 5) and the linkage criterion B (see Definition 12), we integrate them and further propose a bi-directional association strategy, in which each border point is associated with a non-peeled point (according to the uni-directional association strategy), as well as, with other border points (according to the linkage criterion B).

4.1.3. Clustering phase

After border points are repeatedly peeling off, the remaining core points $\boldsymbol{X}^{(T+1)}$ are better separated and easy to cluster. Following BP, we use the concept of reachable neighborhoods to cluster the remaining set of core points.

For sake of clarity, in the following, we imitate the concepts in DBSCAN, and give some definitions.

Definition 13 (*(Directly reachable neighborhoods).*). *Two core points $\boldsymbol{x}_i \in \boldsymbol{X}^{(T+1)}$ and $\boldsymbol{x}_j \in \boldsymbol{X}^{(T+1)}$ are said to be a pair of directly reachable neighborhoods, if $\boldsymbol{x}_j$ is one of K nearest neighbors of $\boldsymbol{x}_i$ from the core point set (or $\boldsymbol{x}_i$ is one of K nearest neighbors of $\boldsymbol{x}_j$ from the core point set) and $d(\boldsymbol{x}_i, \boldsymbol{x}_j) \leqslant max\left(l_i^{(T)}, l_j^{(T)}\right)$, where l_i is a spatially-variant threshold value, as stated earlier.*

Definition 14 (*(Reachable neighborhoods).*). *Two core points $\boldsymbol{x}_i \in \boldsymbol{X}^{(T+1)}$ and $\boldsymbol{x}_j \in \boldsymbol{X}^{(T+1)}$ are said to be a pair of reachable neighborhoods, if there is a sequence of core points $\boldsymbol{x}_i = \boldsymbol{x}_{k_1}, \ldots, \boldsymbol{x}_{k_m} = \boldsymbol{x}_j$ such that $\boldsymbol{x}_{k_r}$ and $\boldsymbol{x}_{k_{r+1}}$ are a pair of directly reachable neighborhoods.*

Different from the concepts of directly density-reachable and density-reachable in DBSCAN, which are in general not symmetric in case one core point and one border point are involved, "directly reachable" and "reachable" in BP and ROBP are symmetric relations. Because the two concepts in ROBP and BP are not defined for border points, but for core points.

Intuitively, a cluster of core points is a maximal set of reachable core points. Mathematically, we have the following definition.

Definition 15 ((*Clusters of core points).*). *A cluster C_c of core points is a nonempty subset of $\boldsymbol{X}^{(T+1)}$ satisfying the following two conditions*

1. $\forall \boldsymbol{x}_i, \boldsymbol{x}_j \in \boldsymbol{X}^{(T+1)}$, if $\boldsymbol{x}_i \in C_c$ and $\boldsymbol{x}_j$ is a reachable neighborhood of $\boldsymbol{x}_i$, then $\boldsymbol{x}_j \in C_c$(maximality).
2. $\forall \boldsymbol{x}_i, \boldsymbol{x}_j \in C_c$, $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are reachable (connectivity).

4.2. Algorithm description

The following algorithm is a summary of the proposed ROBP.

Algorithm1. ROBP algorithm.

Inputs:
Dataset: $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\} \in R^m$
Parameter: number of neighbors k
Outputs:
Cluster indices: $\boldsymbol{y} = \{y_1, \ldots, y_n\}$
Method:
Step 1: Calculate distance matrix according to Eq. (1)
Step 2: Calculate SNN similarity according to Eq. (15)
Step 3: Initialize $t \leftarrow 1, \boldsymbol{X}^{(1)} \leftarrow \boldsymbol{X}$
Step 4: Repeat the peeling iteration
  Step 4.1: Construct RKNN graph of $\boldsymbol{X}^{(t)}$ according to Eq. (3)
  Step 4.2: Calculate density influence values $\hat{\boldsymbol{b}}^{(t)}$ of $\boldsymbol{X}^{(t)}$ according to Eq. (13)
  Step 4.3: Select the set of peeled points $\boldsymbol{X}_B^{(t)}$ according to Eq.7
  Step 4.4: Update $\boldsymbol{X}^{(t+1)} \leftarrow \boldsymbol{X}^{(t)} \; \boldsymbol{X}_B^{(t)}$
  Step 4.5: Implement uni-directional association strategy for $\boldsymbol{X}_B^{(t)}$ (see Definition 5)
  Step 4.6: Implement linkage criterion B for $\boldsymbol{X}_B^{(t)}$ (see Definition 12)
Step 5: Generate clusters of core points (see Definition 15)
Step 6: Assign peeled border points to clusters according to their association to the clustered core points

4.3. Complexity analysis

Now, we give the time complexity of the ROBP algorithm. To be consistent with the above notations, we assume that n is the number of points in the data set; k is the number of nearest neighbors of one point; T is the number of peeling iterations; n_b is the average number of the peeled border points at each iteration, where $n_b \ll n$. The computational time of distance matrix is $O(n^2)$. Note that this step can be performed in parallel, thus its running time can be reduced to a low level. For calculating SNN similarity, ROBP needs to find out the k -nearest-neighbors of each point. the cost of this step is $O(kn^2)$. Step 2 not only can be performed in parallel but also can be accelerated using indexing structures such as KD-tree and R*-tree. In the peeling process, constructing RKNN graph is the most time-consuming step. Its computational complexity is no more than $O(n_b n)$. Actually, its complexity is asymptotic with respect to the number of points. Specifically, the cost of Step 4.1 is $O(n_b \cdot (n - tn_b))$ at the t^{th} iteration. Step 4.6 is our proposed linkage criterion, and its complexity is no more than $O(n_b n)$. Similarly to Step 4.1, the cost of this step actually is $O(n_b \cdot tn_b)$ at the t^{th} iteration. Thus, the total cost of Step 4 is approximately $O(T \cdot nn_b)$. The cost from Step 5 to Step 6 is approximately $O(n^2)$. The total time cost of ROBP is $O(n^2 + kn^2 + Tn_b n + n^2) \sim O(kn^2)$. Therefore, the overall computational complexity of ROBP is on the order of $O(kn^2)$.

5. Experiments

In this section, we compare ROBP's performance with seven counterparts over synthetic and real-world datasets. In addition, we compare the robustness of ROBP and BP, and evaluate their running time.

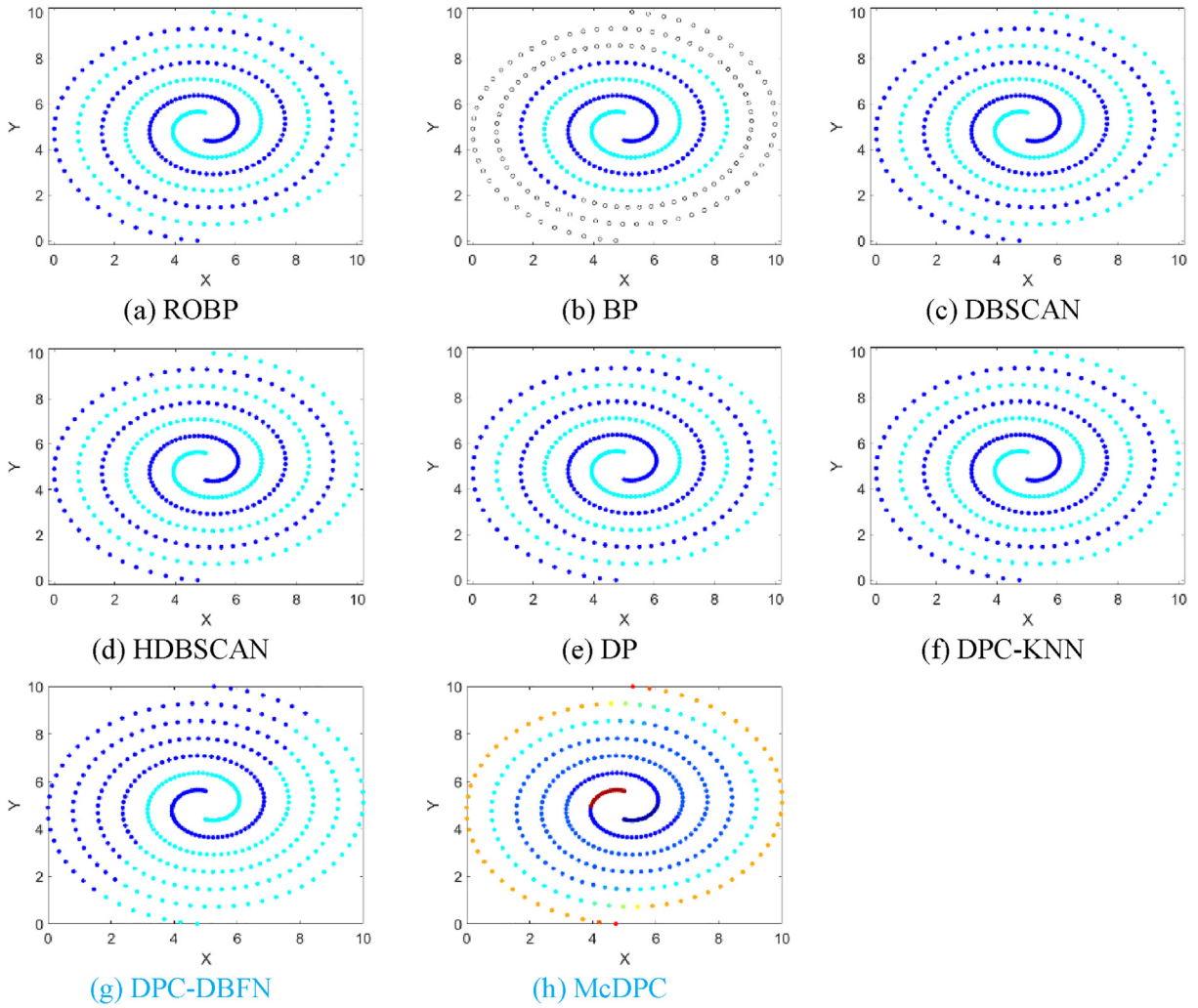

Fig. 6. Clustering results on the Twospirals dataset.

5.1. Experimental settings

5.1.1. Baseline methods and settings

To demonstrate the effectiveness of the proposed algorithm, we select seven representative density-based clustering (or hierarchical clustering) algorithms as baseline methods, including border-peeling (BP) clustering [3], DBSCAN [10], HDBSCAN [5], density peak (DP) clustering [36], and DPC-KNN [8], DPC-DBFN [27], McDPC [45].

For BP, we reimplement a Matlab version based on a Python implementation provided by the author. For DBSCAN, we use the implementation provided by Matlab. For HDBSCAN, DP, DPC-KNN, DPC-DBFN and McDPC, we use the publicly available Matlab source code published by the authors of these methods. These clustering methods can automatically discover natural structures of the dataset and do not accept the number of clusters as a parameter. However, in DP, DPC-KNN, and DPC-DBFN, cluster centroids (or so-called density peaks) are manually selected from the decision graph. For a fair comparison, we use a simple determination strategy to automatically find the correct number of clusters, and thus we introduce a parameter α into them.

To more objectively reflect the actual results of various algorithms, all clustering methods are run several times with distinct values of the parameters for each dataset, and we take the parameters leading to the most 'promising' results, according to the Adjusted Rand Index metric (see Section 5.1.2). In the following, we detail the parameter settings for these methods.

For BP and ROBP, we tune one major parameter (i.e., the number of neighbors k), and the parameter k of those algorithms varies from 2 to 25. DBSCAN has two parameters, ε (so-called *Eps*, i.e., radius of neighborhood) and k (so-called *Minpts*, i.e., a core point must have at least k points within an ε-neighborhood). Following the search range of ε in [3], it is 20 linearly sampled values in the range $[0.1,\ 5.1]$. For a fair comparison, we also select the parameter k continuously from 2 to 25. HDBSCAN

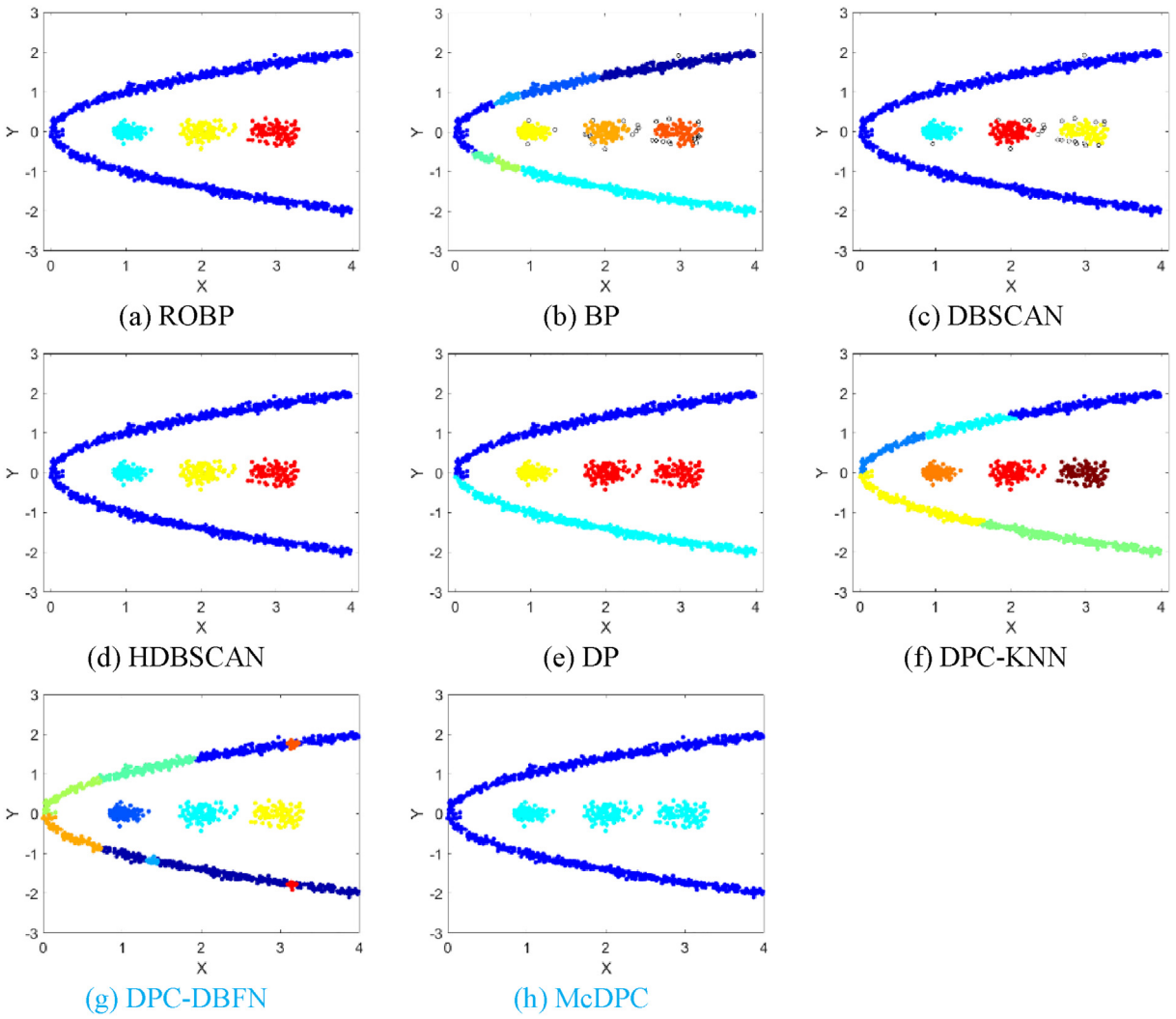

Fig. 7. Clustering results on the Data5 dataset.

has two major parameters, N_c (i.e., smallest size to be considered a cluster) and k (i.e., the number of neighbors). We select the parameter N_c from $\{5,\ 10,\ 15,\ 20\}$. The search range of the parameter k is the same as BP and ROBP. DP has one parameter d_c (i.e., a cutoff distance). The parameter is expressed as a percentage, and it varies from 1% to 10% in 1% increments. DPC-KNN and DPC-DBFN have one parameter k (i.e., the number of neighbors). The search range of the parameter k is the same as BP and ROBP. For DP, DPC-KNN and DPC-DBFN, their clustering processes rely on intervention from the user. In particular, the user must select suitable centroids from the decision graph manually. In order to automatically discover centroids, we use a simple determination strategy in [7], and introduce a parameter α into them. We loop the parameter α from 1 to 5, with a step size of 0.5. Different from DP, DPC-KNN and DPC-DBFN, McDPC can automatically discover centroids. McDPC takes four parameters, namely γ, θ, λ, and d_c. Parameter γ varies from 0.1 to 0.3 in 0.1 increments. We select the parameter θ and λ from $\{1,\ 2,\ 3\}$. Same as DP, d_c varies from 1% to 10% in 1% increments. For sake of clarity, parameter settings for the methods are summarized in Table 2.

All the experiments are measured on a computer with an Intel Core i7-8750U 2.2 GHz processor, and 16 GB RAM running Matlab 2018b.

5.1.2. Measures

For evaluating the performance of a clustering algorithm, there are a number of possible measures based on the ground truth in the field of data clustering. In our experiments, we evaluate the clustering results by calculating some widely-used evaluation indices [15,28], including Adjusted Rand Index (ARI) [20], Normalized Mutual Information (NMI) [29], and Adjusted Mutual Information (AMI) [43]. In order to further evaluate the performance of each method, in the supplementary

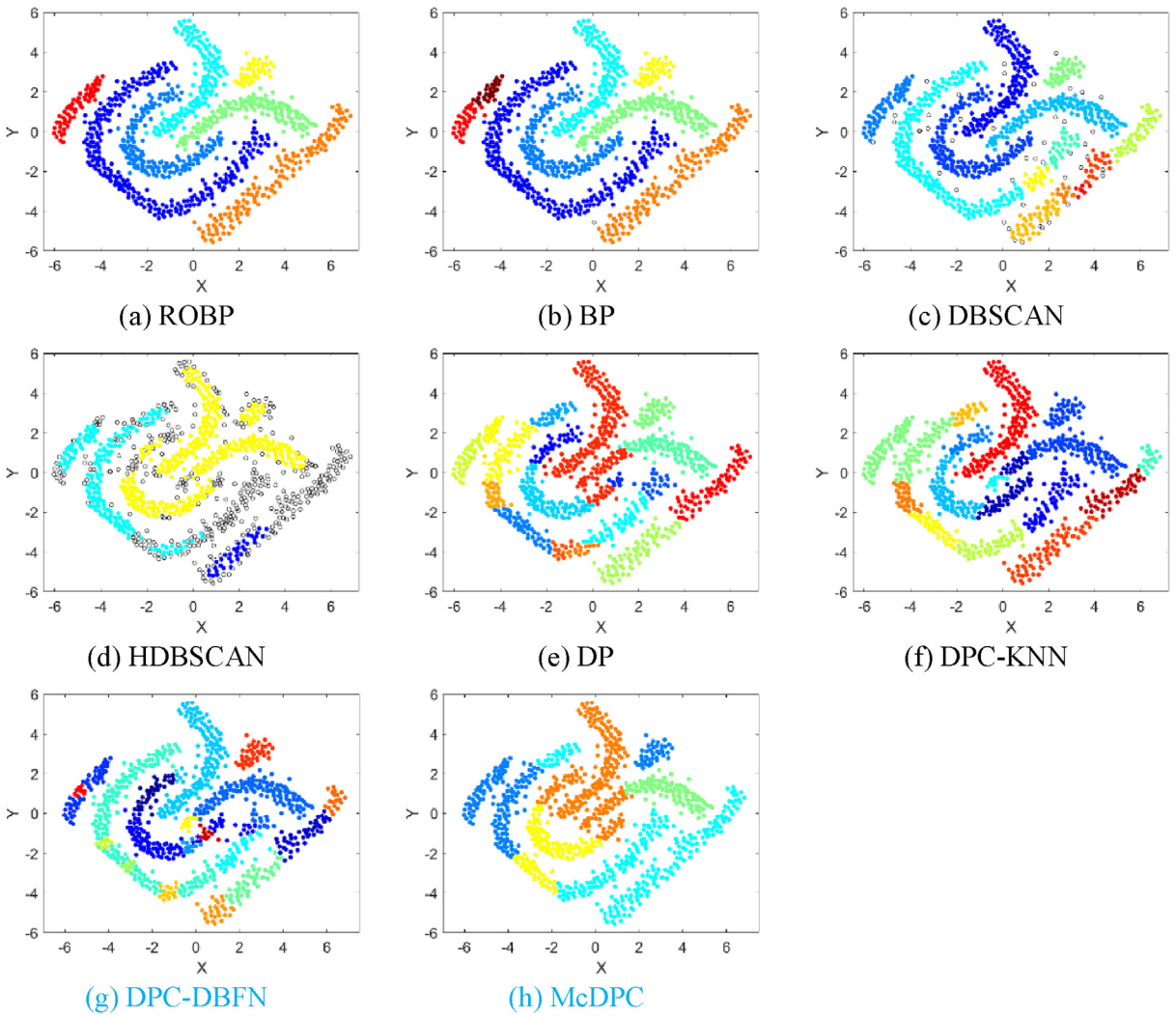

Fig. 8. Clustering results on the DS5 dataset.

material, we introduce Fowlkes–Mallows Index (FMI) [12] and Clustering Validation Index based on Nearest Neighbors (CVNN) [26], and we also provide qualitative results accordingly.

Let $\boldsymbol{C} = \{c_1, c_2, \ldots, c_l\}$ and $\hat{\boldsymbol{C}} = \{\hat{c}_1, \hat{c}_2, \ldots, \hat{c}_{l'}\}$be two clusterings which to be compared. ARI is defined as follows:

$$ARI\left(\boldsymbol{C}, \hat{\boldsymbol{C}}\right) = \frac{RI\left(\boldsymbol{C}, \hat{\boldsymbol{C}}\right) - E\left[RI\left(\boldsymbol{C}, \hat{\boldsymbol{C}}\right)\right]}{\max\left(RI\left(\boldsymbol{C}, \hat{\boldsymbol{C}}\right)\right) - E\left[RI\left(\boldsymbol{C}, \hat{\boldsymbol{C}}\right)\right]} \tag{19}$$

where $RI(\cdot,\cdot)$ is the Rand Index (RI) [35], and $E[\cdot]$ is the expected value, and $\max(\cdot)$ is the maximum value the index can take.

The NMI between $\boldsymbol{C}$ and $\hat{\boldsymbol{C}}$ is defined as

$$NMI\left(\boldsymbol{C}, \hat{\boldsymbol{C}}\right) = \frac{I\left(\boldsymbol{C}, \hat{\boldsymbol{C}}\right)}{\sqrt{H(\boldsymbol{C})H\left(\hat{\boldsymbol{C}}\right)}} \tag{20}$$

where $I\left(\boldsymbol{C}, \hat{\boldsymbol{C}}\right) = H(\boldsymbol{C}) + H\left(\hat{\boldsymbol{C}}\right) - H\left(\boldsymbol{C}, \hat{\boldsymbol{C}}\right)$ is the mutual information between $\boldsymbol{C}$ and $\hat{\boldsymbol{C}}$, and $H(\cdot)$is the entropy.

The AMI between $\boldsymbol{C}$ and $\hat{\boldsymbol{C}}$ is defined as

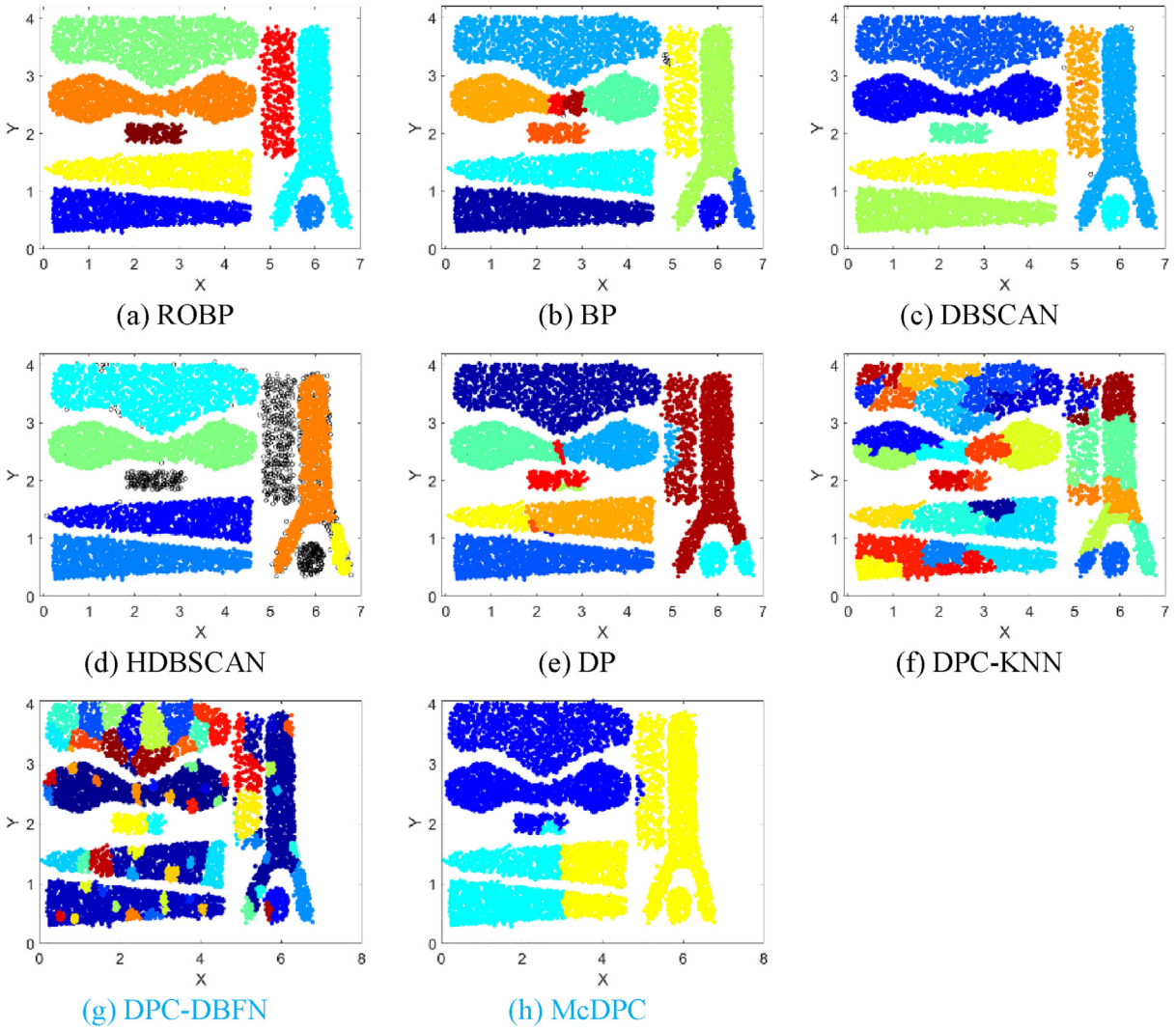

Fig. 9. Clustering results on the T8 dataset.

$$AMI\left(\mathbf{C},\hat{\mathbf{C}}\right)=\frac{I\left(\mathbf{C},\hat{\mathbf{C}}\right)-E\left[I\left(\mathbf{C},\hat{\mathbf{C}}\right)\right]}{\sqrt{H(\mathbf{C})H\left(\hat{\mathbf{C}}\right)}-E\left[I\left(\mathbf{C},\hat{\mathbf{C}}\right)\right]} \tag{21}$$

Values of three measures lie between 0 and 1. The higher their scores, the better the clustering performance. However, values of ARI can be negative which is of no practical interest.

5.2. Experiments on synthetic datasets

In this subsection, we present the clustering results of ROBP and seven counterparts on eight synthetic datasets.

5.2.1. Synthetic datasets

These synthetic datasets used in the experiments are illustrated in Table 3. The first three datasets are all of clusters of different densities. On Target, Twospirals, and Data5, one or more clusters are not relatively "homogeneous". Other datasets are of clusters with complex manifold.

5.2.2. Results on synthetic datasets

The Diffdensity dataset consists of four Gaussian clusters with different variances. In other words, local densities of four clusters in the Diffdensity dataset are considerably different. Fig. 3 shows the clustering results of each method on Diffden-

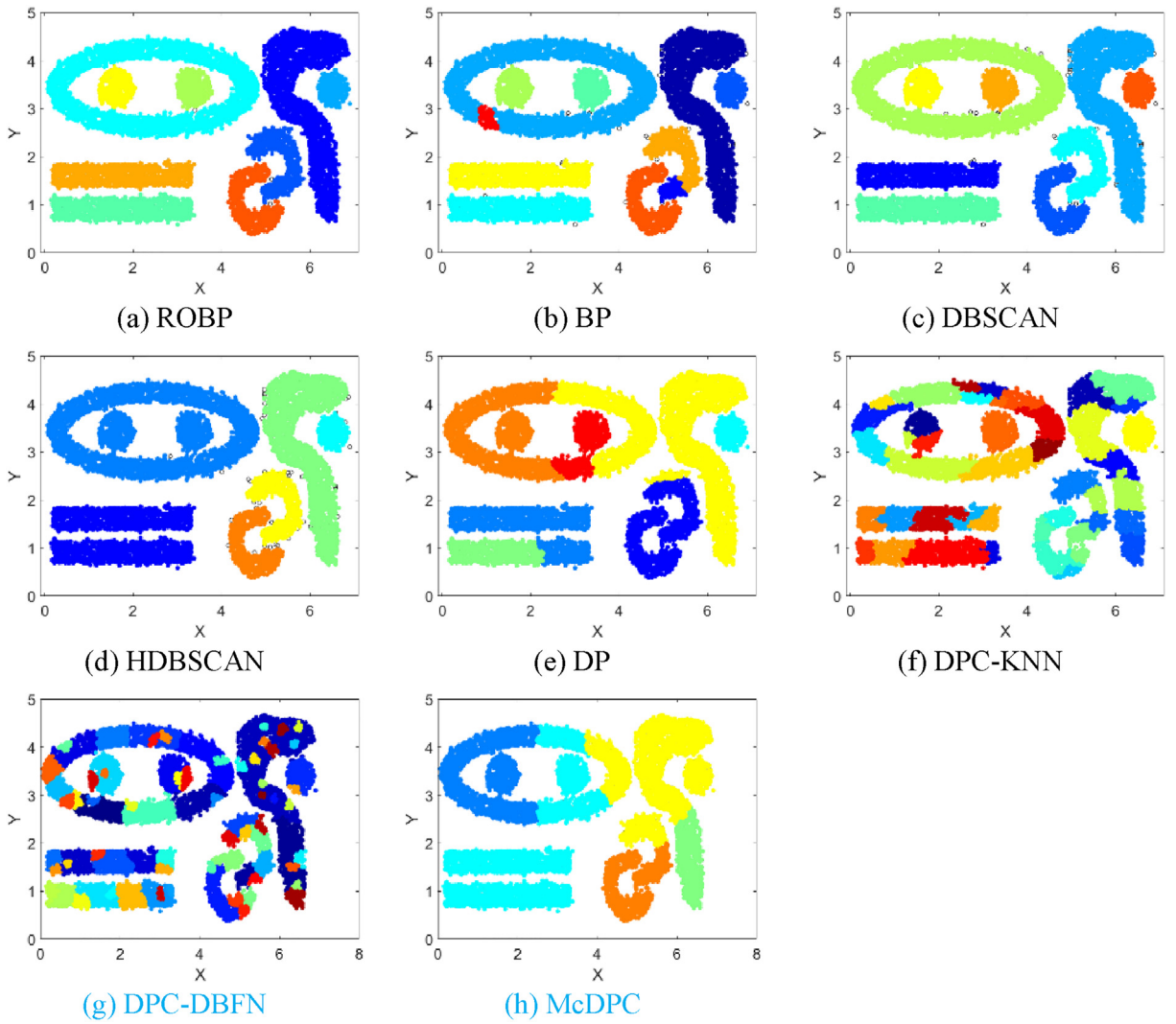

Fig. 10. Clustering results on the T7 dataset.

sity, where the clusters are colored in unique colors, while black bubbles (or circles) represent outliers. ROBP, DP, DPC-KNN and McDPC can detect the true structure of the dataset. Although BP is also able to correctly find the general shapes of each cluster, some points are wrongly marked as outliers. DBSCAN and HDBSCAN fail to obtain good results, because they incorrectly merge the rightmost two clusters into a single cluster. For DPC-DBFN, a few points are erroneously clustered.

The Jain dataset consists of two crescent-shaped clusters with different densities. Because the Jain dataset has non-convex clusters that are not only of various densities, but also intertwined with each other, it is more challenging than the Diffdensity dataset. As shown in Fig. 4, McDPC and our algorithm can truly handle that well. However, the BP algorithm divides the upper sparse cluster into four different clusters and some outliers. Other algorithms are not able to perfectly distinguish two clusters.

The Target dataset has one dense spherical cluster and one sparse ring-shaped cluster. Additionally, it is important to note that the ring-shaped cluster is not relatively "homogeneous". That means the density varies between different regions in the cluster. Fig. 5 shows the results of the Target dataset. As shown, ROBP, DBSCAN, HDBSCAN and McDPC can accurately identify two clusters. Unlike our method, the BP, DP, DPC-KNN and DPC-DBFN methods partition the sparse cluster into several small clusters.

As shown in Figs. 3-5, when clusters in different regions of the data space have considerably different local densities or when clusters with different density levels are nested, the proposed algorithm can still yield the optimal structure of clusters. Notice that our algorithm gets better performance compared to the BP algorithm over these datasets. BP tends to peel off points in low-density regions. Especially, when the sparse clusters have complex shapes, BP may peel off points in the core region of a sparse cluster rather than those in the border region. This causes BP incorrectly partitions a sparse cluster

Table 4
Performance of the evaluated algorithms on synthetic datasets.

Algorithm	ARI	AMI	NMI	Clu#	ARI	AMI	NMI	Clu#
	Diffdensity (4 classes)				Jain (2 classes)			
ROBP	**1.000**	**1.000**	**1.000**	4	**1.000**	**1.000**	**1.000**	2
BP	0.987	0.963	0.963	4	0.583	0.659	0.656	6
DBSCAN	0.393	0.681	0.679	3	0.256	0.261	0.256	2
HDBSCAN	0.334	0.596	0.594	3	0.943	0.888	0.887	2
DP	0.996	0.993	0.993	4	0.758	0.676	0.675	2
DPC-KNN	0.996	0.993	0.993	4	0.732	0.681	0.679	3
DPC-DBFN	0.957	0.938	0.939	5	0.67	0.514	0.516	2
McDPC	0.996	0.993	0.993	4	**1.000**	**1.000**	**1.000**	2
	Target (2 classes)				Twospirals (2 classes)			
ROBP	**1.000**	**1.000**	**1.000**	2	**1.000**	**1.000**	**1.000**	2
BP	0.649	0.649	0.690	6	0.418	0.515	0.514	2
DBSCAN	**1.000**	**1.000**	**1.000**	2	**1.000**	**1.000**	**1.000**	2
HDBSCAN	**1.000**	**1.000**	**1.000**	2	**1.000**	**1.000**	**1.000**	2
DP	0.643	0.694	0.693	6	**1.000**	**1.000**	**1.000**	2
DPC-KNN	0.619	0.662	0.661	8	**1.000**	**1.000**	**1.000**	2
DPC-DBFN	0.579	0.624	0.626	11	0.109	0.08	0.082	2
McDPC	**1.000**	**1.000**	**1.000**	2	0.004	0.042	0.057	12
	Data5 (4 classes)				DS5 (7 classes)			
ROBP	**1.000**	**1.000**	**1.000**	4	**1.000**	**1.000**	**1.000**	7
BP	0.223	0.599	0.595	10	0.989	0.981	0.980	8
DBSCAN	0.993	0.953	0.953	4	0.781	0.848	0.845	13
HDBSCAN	0.203	0.632	0.629	8	0.210	0.390	0.386	3
DP	0.427	0.697	0.696	4	0.451	0.729	0.721	21
DPC-KNN	0.203	0.632	0.629	8	0.479	0.743	0.737	15
DPC-DBFN	0.208	0.613	0.616	11	0.641	0.783	0.79	19
McDPC	0.925	0.825	0.826	2	0.42	0.625	0.628	5
	T8 (8 classes)				T7 (9 classes)			
ROBP	**1.000**	**0.999**	**0.999**	8	**1.000**	**0.999**	**0.999**	9
BP	0.878	0.930	0.930	12	0.970	0.973	0.973	11
DBSCAN	0.999	0.998	0.998	9	0.994	0.991	0.991	9
HDBSCAN	0.928	0.925	0.925	6	0.790	0.883	0.883	6
DP	0.812	0.875	0.875	12	0.538	0.739	0.739	7
DPC-KNN	0.296	0.683	0.681	32	0.247	0.703	0.701	39
DPC-DBFN	0.592	0.735	0.738	63	0.315	0.709	0.713	76
McDPC	0.424	0.597	0.597	3	0.322	0.578	0.578	5

Table 5
Real-world datasets.

Data Sets	Class#	Dimension#	Number#
Soybean	4	35	47
Zoo	7	16	101
Iris	3	4	150
Wine	3	13	175
Glass	6	10	214
Dermatology	6	34	366
WDBC	2	30	569
MSRA	12	256	1799
YTF	41	10	10,000
Penbased	10	16	10,992

into several small sub-clusters. We use the proposed bi-directional association strategy described in Section 4.1.2 to solve this problem elegantly. The experimental results shown in Fig. 3(a)-5(a) demonstrate ROBP's ability to recognize clusters with different densities.

The Twospirals dataset is composed of two spiral-shaped clusters with non-uniform distributions. There are considerably different local densities in different regions of the same cluster. The "head" (i.e., the points in the central area) has a higher density rather than the "tail". Fig. 6 displays the results on the Twospirals dataset. Most algorithms can recognize successfully finds two clusters. However, BP mistakenly marks some points in low-density region as outliers. DPC-DBFN and McDPC fail to obtain good results.

The Data5 dataset consists of three spherical clusters and one crescent-shaped cluster. The crescent-shaped cluster is elongated and has a non-uniform distribution. Fig. 7 clearly shows the results of ROBP and HDBSCAN are perfect. For DBSCAN, although some points are marked as outliers, the general shapes of each cluster are correct. BP, DP, DPC-KNN

Table 6
Performance of the evaluated algorithms on real-world datasets.

Algorithm	ARI	AMI	NMI	Clu#	ARI	AMI	NMI	Clu#
	Soybean (4 classes)				Zoo (7 classes)			
ROBP	**1.000**	**1.000**	**1.000**	4	0.929	**0.899**	**0.908**	4
BP	0.661	0.814	0.830	3	0.931	0.897	0.906	4
DBSCAN	0.960	0.959	0.963	4	**0.933**	0.883	0.897	6
HDBSCAN	0.674	0.809	0.826	3	0.873	0.815	0.832	4
DP	NA	NA	NA	NA	0.74	0.755	0.767	3
DPC-KNN	**1.000**	**1.000**	**1.000**	4	0.892	0.851	0.864	5
DPC-DBFN	0.701	0.743	0.765	4	0.071	0.319	0.359	3
McDPC	0.501	0.699	0.709	3	0.487	0.587	0.646	12
	Iris (3 classes)				Wine (3 classes)			
ROBP	0.831	0.800	0.803	3	**0.732**	**0.748**	**0.751**	3
BP	0.569	0.652	0.661	5	**0.732**	**0.748**	**0.751**	3
DBSCAN	0.635	0.651	0.655	2	0.426	0.554	0.559	2
HDBSCAN	0.568	0.759	0.761	2	0.294	0.420	0.428	2
DP	0.720	0.781	0.784	3	0.672	0.707	0.710	3
DPC-KNN	**0.886**	**0.861**	**0.862**	3	0.727	0.741	0.743	3
DPC-DBFN	0.674	0.757	0.76	3	0.495	0.63	0.634	3
McDPC	0.568	0.759	0.761	2	0.321	0.344	0.361	5
	Glass (6 classes)				Dermatology (6 classes)			
ROBP	**0.650**	**0.724**	**0.739**	7	**0.853**	**0.916**	**0.918**	5
BP	0.529	0.703	0.721	8	0.444	0.658	0.664	4
DBSCAN	0.630	0.700	0.716	7	0.441	0.634	0.641	4
HDBSCAN	0.423	0.577	0.593	4	0.438	0.627	0.636	5
DP	0.551	0.551	0.560	3	0.836	0.834	0.839	10
DPC-KNN	0.435	0.489	0.553	25	0.798	0.840	0.844	7
DPC-DBFN	0.469	0.491	0.501	3	0.711	0.801	0.804	4
McDPC	0.173	0.269	0.288	3	0.411	0.618	0.624	4
	WDBC (2 classes)				MSRA (12 classes)			
ROBP	0.682	0.558	0.559	2	0.652	0.868	0.872	24
BP	0.683	0.539	0.540	2	0.586	0.822	0.827	23
DBSCAN	0.371	0.340	0.341	2	**0.672**	**0.879**	**0.883**	30
HDBSCAN	0.369	0.357	0.358	2	0.443	0.746	0.756	35
DP	0.283	0.244	0.248	7	0.545	0.798	0.815	105
DPC-KNN	0.676	0.575	0.575	2	0.582	0.826	0.831	23
DPC-DBFN	0.11	0.175	0.177	2	0.419	0.656	0.667	28
McDPC	**0.701**	**0.578**	**0.579**	2	0.399	0.741	0.751	41
	YTF (41 classes)				Penbased (10 classes)			
ROBP	**0.734**	**0.886**	**0.892**	83	0.618	**0.820**	**0.821**	11
BP	0.647	0.858	0.865	91	0.661	0.787	0.788	13
DBSCAN	0.007	0.293	0.293	31	0.541	0.708	0.710	24
HDBSCAN	0.636	0.842	0.852	113	0.504	0.687	0.688	26
DP	0.585	0.846	0.859	221	0.635	0.754	0.754	10
DPC-KNN	0.545	0.843	0.857	206	0.618	0.782	0.784	61
DPC-DBFN	0.063	0.546	0.585	321	**0.676**	0.759	0.765	144
McDPC	0.004	0.195	0.195	59	0.116	0.339	0.339	3

Table 7
Performance of the evaluated algorithms on Olivetti face dataset.

Algorithm	ARI	NMI	AMI	Clu#
ROBP	**0.957**	**0.972**	**0.979**	11
BP	0.718	0.909	0.926	7
DBSCAN	0.787	0.859	0.895	12
HDBSCAN	0.843	0.893	0.916	9
DP	0.834	0.87	0.904	14
DPC-KNN	0.854	0.893	0.922	14
DPC-DBFN	0.689	0.797	0.833	8
McDPC	0.388	0.673	0.715	6

and DPC-DBFN divide the elongated, crescent-shaped cluster into several small sub-clusters. McDPC mistakenly merges three spherical clusters into one cluster.

As shown in Figs. 5-7, when different regions of the same cluster have considerably different local densities, the BP clustering method is not able to handle this problem. Especially, when the non-uniformly-distributed clusters are very elon-

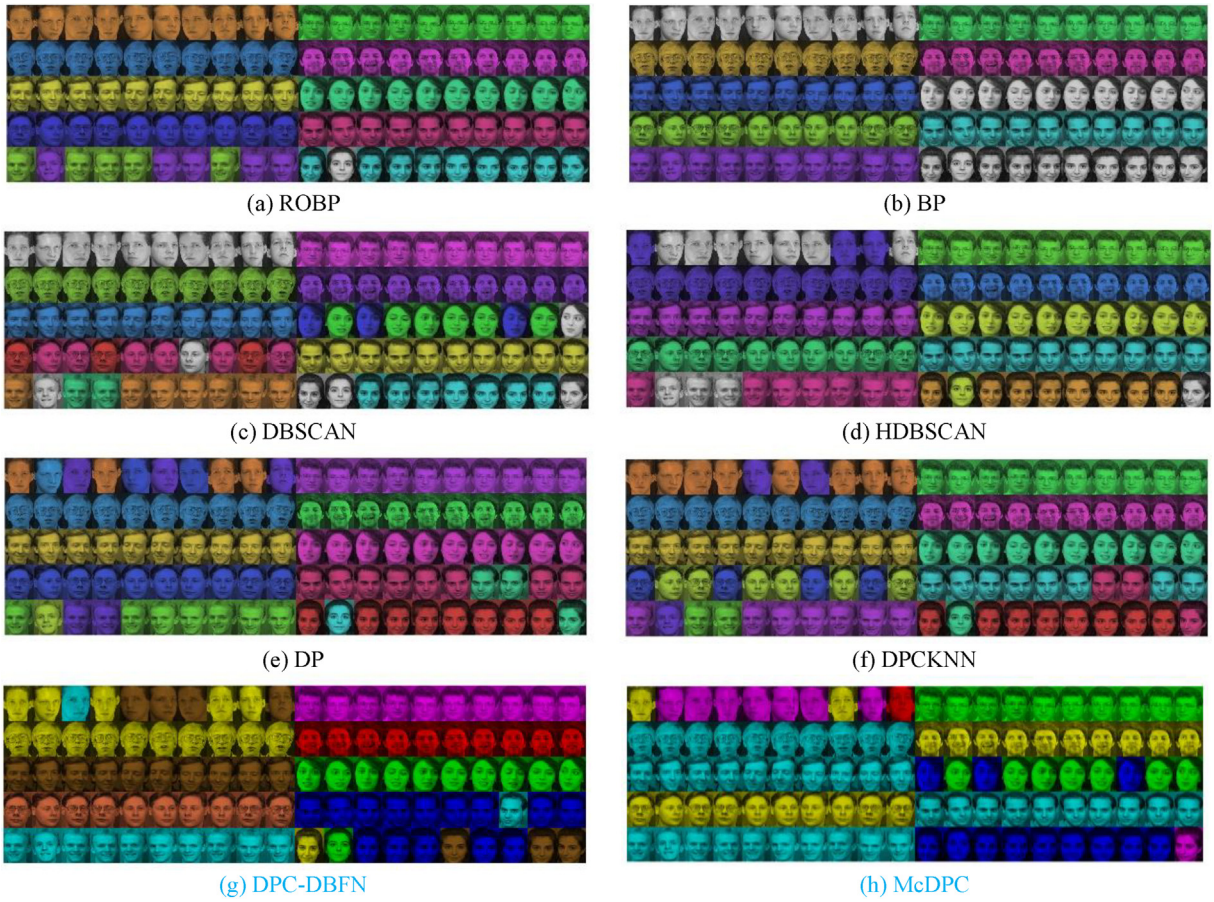

Fig. 11. Visualized clustering results on Olivetti face dataset.

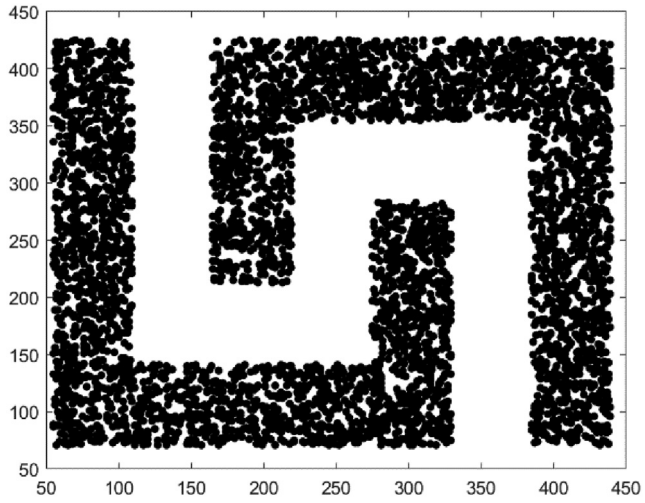

Fig. 12. Visualization of the C2 dataset.

gated, BP is prone to divide them into undesirable small sub-clusters. This is in part because the local scaling function based on Gaussian kernel may ignore the contributions of some faraway nearest neighbors. In other words, a density estimator based on Gaussian kernel is sensitive to changes in the density. This may cause some points in the core areas of the clusters

Table 8
Synthetic datasets used in robustness experiments.

Data Sets	Class#	Dimension#	Number#
C2	2	2	5000
C2D1	2	2	4375
C2D2	2	2	3750
C2D3	2	2	3125
C2N1	2	2	5250
C2N2	2	2	5500
C2N3	2	2	5720

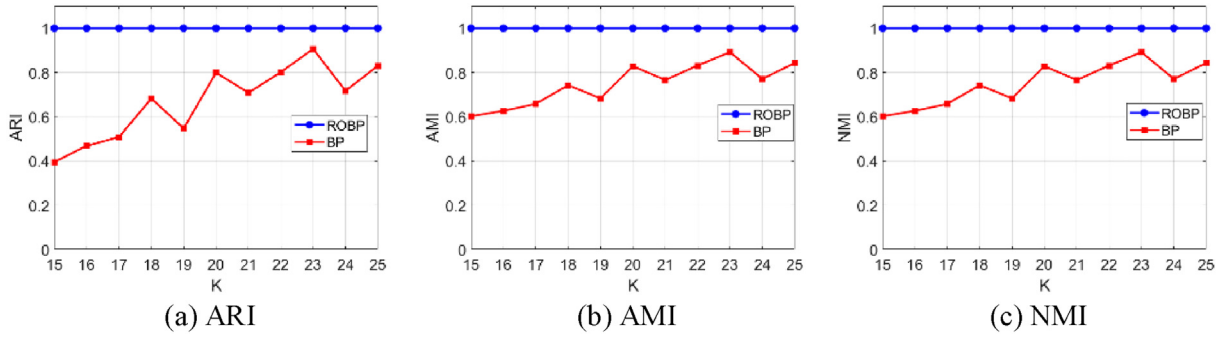

Fig. 13. Performance of BP and ROBP on the C2 dataset with different k parameter values.

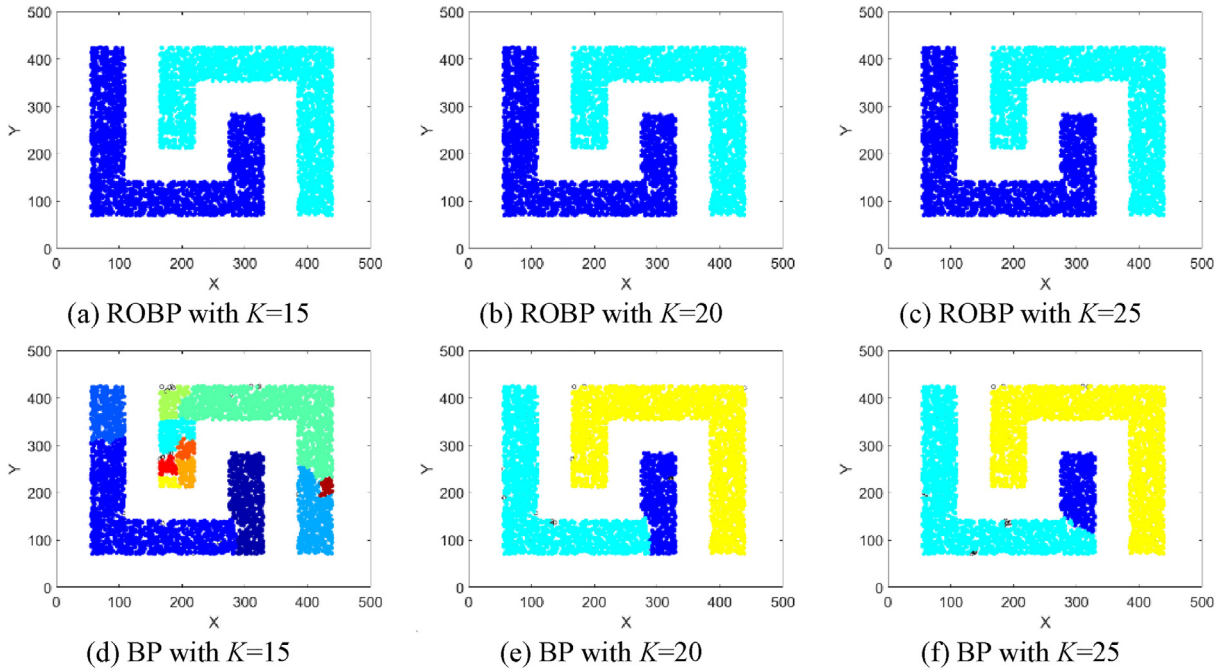

Fig. 14. Results of BP and ROBP on the C2 dataset with three representative parameter values.

are mistakenly peeled off. Since the subsequent peeling is closely related to the previously peeled points, when some points are erroneously peeled off, the subsequent peeling of the points will be misdirected, resulting in a larger peeling error. To overcome this problem, we use Cauchy kernel to improve the density influence, and propose the linkage criterion B to avoid over-partitioning. The experimental results illustrate our algorithm is very effective in finding clusters with non-uniform densities.

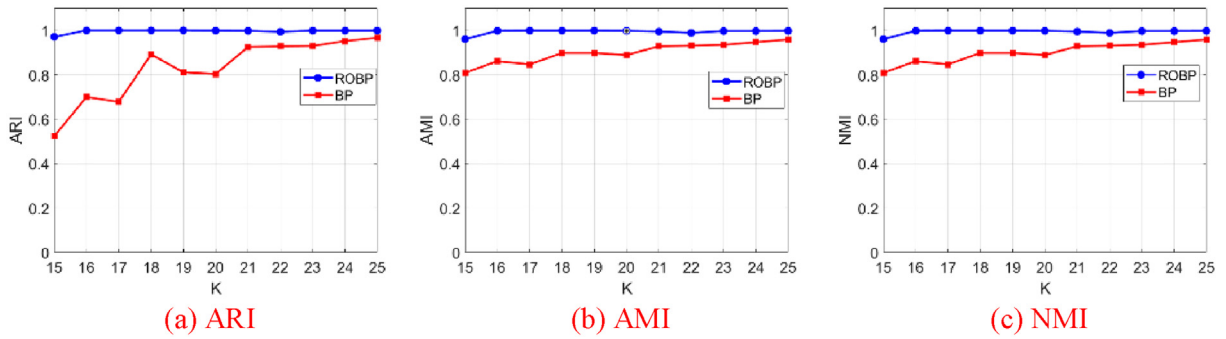

Fig. 15. Performance of BP and ROBP on the T7 dataset with different k parameter values.

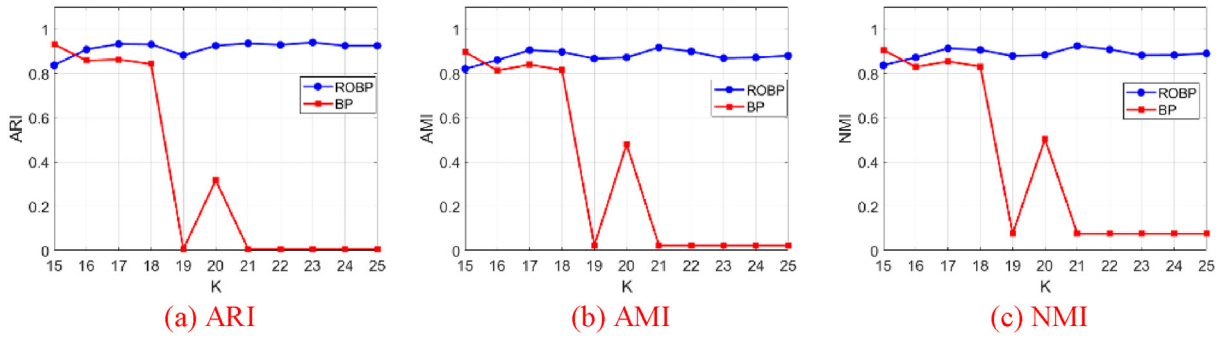

Fig. 16. Performance of BP and ROBP on the Zoo dataset with different k parameter values.

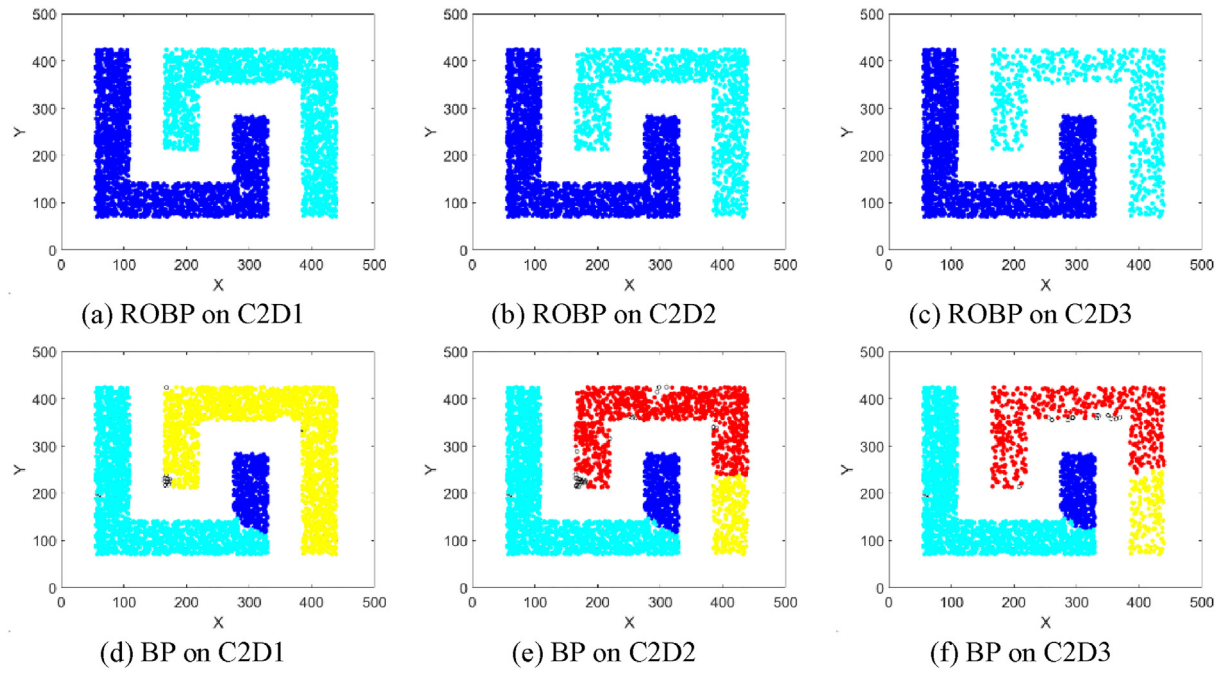

Fig. 17. Results of BP and ROBP on C2D1, C2D2 and C2D3 datasets.

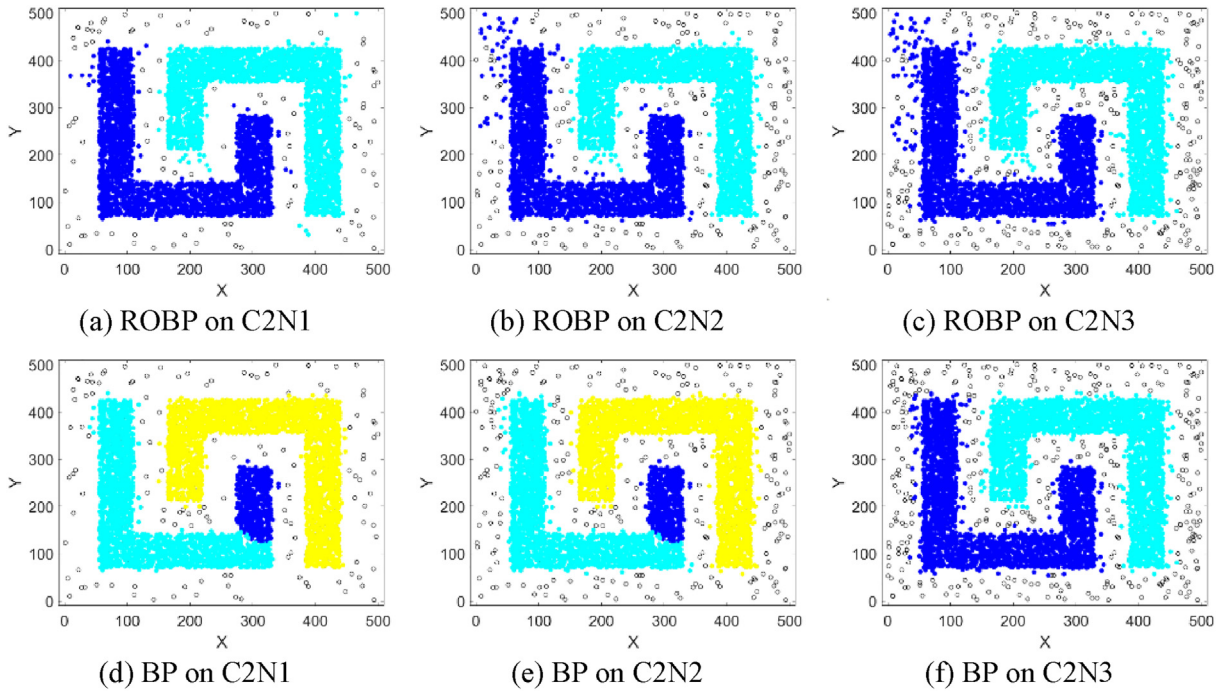

Fig. 18. Results of BP and ROBP on the C2N1, C2N2 and C2N3 datasets.

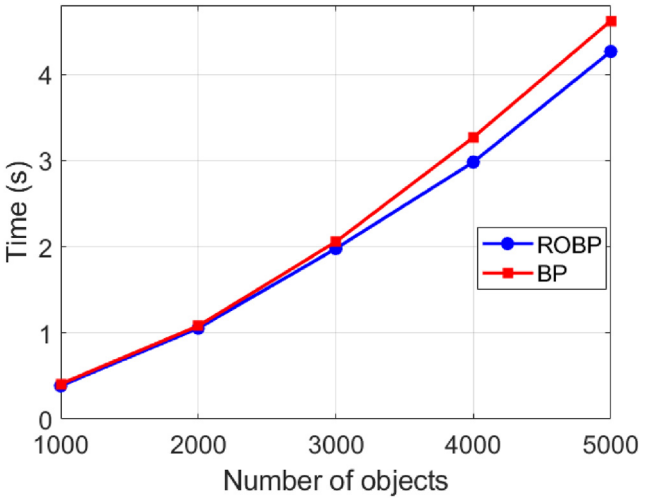

Fig. 19. Execution time vs. number of objects.

The DS5 dataset consists of seven intertwined, complex-shaped clusters. For the DS5 dataset shown in Fig. 8, only ROBP correctly identify all clusters. It is noteworthy that BP gets better clustering performance compared to other six clustering algorithms. As in the former cases, BP divides the upper-left cluster into two sub-clusters.

The T8 dataset is composed of eight clusters with different sizes and different shapes. Fig. 9 shows the clustering results of each approach on the T8 dataset. Despite other methods fail to obtain good results, our algorithm does an excellent job in clustering the dataset. Actually, the clustering obtained by DBSCAN is close to the optimal structure of clusters. Relatively small amounts of points are wrongly clustered.

The T7 dataset consists of nine intertwined, or nested clusters. As shown in Fig. 10, the BP and DBSCAN algorithms outperforms other six methods on the dataset. Notice that the result obtained by BP is almost as good as the one obtained by ROBP. Only some points in two small regions are erroneously clustered.

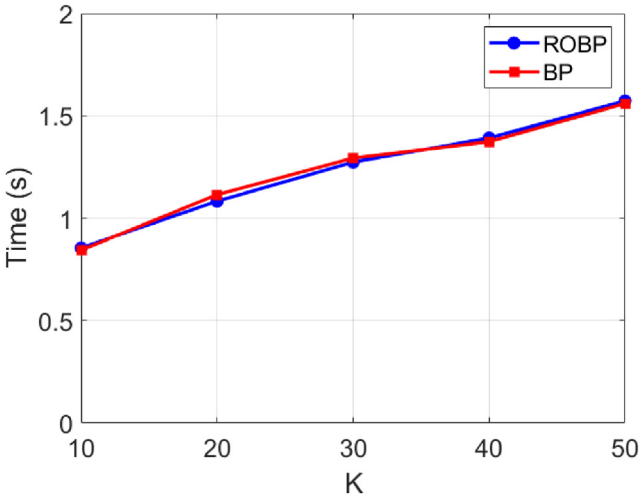

Fig. 20. Execution time vs. parameter k.

Figs. 8-10 demonstrate the performance of ROBP is better than those of other methods in finding clusters with complex shapes.

Table 4 illustrates quantitative results on these synthetic datasets, where Clu# means the total number of output clusters obtained by each evaluated algorithm. For each dataset, the maximum achieved scores are highlighted in bold. In conclusion, the proposed algorithm clearly outperforms other algorithms through most of the cases. In addition, we present more results, including parameter values, FMI scores and CVNN scores in the supplementary material.

5.3. Experiments on real-world data sets

5.3.1. Real-world datasets

Real-world data sets used in the experiments include Soybean Database (Soybean), Zoo database (Zoo), Iris Plants Database (Iris), Wine Recognition Database (Wine), Glass Identification Database (Glass), Dermatology Database (Dermatology), Wisconsin Diagnostic Breast Cancer (WDBC), YouTube Face database (YTF), MSRA Database (MSRA) and Pen-Based Recognition of Handwritten Digits (Penbased). Table 5 presents real-world datasets used in the experiments. It is important to note that we use the version of YTF used in [3]. This dataset contains 10,000 samples of 41 faces, where the feature vectors are generated using auto-encoders.

5.3.2. Results on real-world datasets

Table 6 reports results on real-world datasets. For each dataset, the maximum achieved scores are highlighted in bold. Alongside the evaluation indices, we also present the number of clusters (Clu#) generated by each algorithm. NA means that the algorithm is not applicable for the corresponding dataset. The main reason is that the DP clustering obtains the same γ values for all data objects on the Soybean dataset. Thus, the method fails to distinguish the cluster centers according to the corresponding γ-graphs. In addition, we present more results, including parameter values, FMI scores and CVNN scores in the supplementary material.

As shown in Table 6, our method outperforms other methods in most test cases. Specifically, both our method and DPC-KNN find the optimal structure of clusters on the Soybean dataset. For the Zoo dataset, ROBP shows a small advantage in terms of the most evaluation indices. For the Iris dataset, our method is slightly inferior to DPC-KNN. The clustering results obtained by ROBP and BP are superior to those obtained by other methods on the Wine dataset. For the Glass and Dermatology datasets, the performance of our method is significantly better than competitors. ROBP, BP and DPC-DBFN have similar performance on the WDBC dataset. Our method is slightly inferior to DBSCAN on the MSRA dataset. For the YTF dataset, the clustering results obtained by ROBP are superior to those obtained by other methods. By comparison with counterparts, the proposed algorithm shows a small advantage in terms of the AMI and NMI measures on the Penbased dataset. In addition to the scores on each dataset, Table 6 also reports the number of clusters found by each algorithm. Our algorithm finds the optimal number of clusters on four of the ten datasets, and the number of clusters detected by ROBP is close to the true one for most of the rest datasets.

5.3.3. Results on Olivetti face dataset

The Olivetti Face Databases is a widespread database in machine learning fields. Like [26], we use the first 100 faces of the Olivetti Face Database in the following experiment. The images of each face are taken 10 different angles, and the of each

image is 92×112. In other words, each class is composed of 10 face images, each with 10,304 features. We use principal component analysis to the original features down to 28.

Table 7 shows the results of all evaluated algorithms. For each dataset, the maximum achieved scores are highlighted in bold. As shown, though ROBP fails to detect the correct cluster number, it achieves the better result than other methods in the three measures. The visualized results are presented in Fig. 11. The clusters are colored in unique colors, and the outliers are colored in gray. Fig. 11(a) shows that our algorithm yields the optimal structure of clusters for the first eight individuals. However, the images of the 9th individual is erroneously divided into two sub-clusters, and one image of the 10th individual is incorrectly marked as an outlier.

5.4. Robustness analysis

In this section, we further compare the robustness of the proposed algorithm with the one of the BP clustering algorithm from several aspects, including the sensitivity of the parameters, the capability of finding clusters with very different densities as well as the robustness to noise.

To further validate the robustness of our algorithm, we create seven synthetic datasets. The C2 dataset, a base dataset, has 5000 data points with 2 features, equally distributed into two classes: G_1 and G_2, as shown in Fig. 12. To create the datasets with different densities, we select the sampling ratioβ_d from $\{3/4,\ 2/4,\ 1/4\}$. The class G_2 is sampled with probability β_d, and the class G_1 is sampled with probability 1. Based on the C2 data set, we create three datasets, including C2D1 ($\beta_d = 3/4$), C2D2 ($\beta_d = 2/4$) and C2D3 ($\beta_d = 1/4$). In addition, three datasets are created as variations of the C2 dataset by adding successively higher degree (5 percent, 10 percent and 15 percent) of random noise. The three datasets created for this experiment are C2N1, C2N2 and C2N3. The seven synthetic datasets are summarized in Table 8.

5.4.1. Robustness to parameter setting

As discussed previously, k is a major parameter in ROBP and BP. For the BP algorithm, the parameter k determines the structure of the RKNN graph, the density influence value of each point and the merging of the core points. For ROBP, it is closely related to the association of border points. In order to evaluate sensitivity on the parameter k, we do the following experiment on the C2 dataset. Same as [3], we set k with values ranging between 15 and 25. In Fig. 13, we illustrate the results of ROBP and BP on the C2 dataset with different k parameter values. It is obvious that results obtained by ROBP are not affected by changes in k. On the other hand, the scores generated by BP are not stable across the different parameter values. Fig. 14 shows their results with three representative parameter values (k = 10, k = 20, and k = 30). As shown, ROBP generates the optimal structure of clusters on the dataset.

Figs. 15 and 16 show the results of two algorithms on the T7 and Zoo datasets with different k parameter values. The results obtained by our method are more stable than those obtained by BP. Generally, the performance of ROBP is robust to the parameter setting.

5.4.2. Robustness to data with different densities

In this part, we evaluate the robustness of two approaches to finding clusters with very different densities. For C2D1, C2D2 and C2D3, there are varying degrees of density level variance between clusters. The degree of density level variance in C2D1 is the least, while the one in C2D3 is the greatest. Fig. 17 shows the results of ROBP and BP (the parameter k value is kept fixed at 20) on three datasets. As shown in Fig. 17 (d-f), the BP clustering algorithm cannot completely describe the inherent structure of these datasets. Fig. 17 (a-c) demonstrate that our algorithm can address the challenge of finding clusters with very different densities.

5.4.3. Robustness to noise

C2N1, C2N2 and C2N3 datasets have two classes with varying noise levels. Fig. 18 displays the results of ROBP and BP (the parameter k value is kept fixed at 20). As shown, ROBP has better performance for data with different noise levels. It is interesting to observe that as the noise levels increase, the performance of BP improves, shown in Fig. 18 (d-f). We conjecture that as the number of noise points increases, it is easier that BP correctly peels off noise points rather than core points to reveal the latent clusters. Generally, ROBP has better robustness against noise.

5.5. Runtime evaluation

As discussed previously, the computational complexities of both ROBP and BP are on the order of $O(kn^2)$. In this section, we evaluate the runtime of the BP and ROBP methods in two investigations. The first investigation is that the numbers of objects is kept fixed while the parameter k value is varied. The second one is that the parameter k value is kept fixed while the numbers of objects is varied. We generate five synthetic datasets (n = 1000, 2000, 3000, 4000, 5000). These datasets consist of two classes with two numerical attributes. The numerical attribute values are generated by sampling normal distributions with different means and standard deviations for each class. For the first class, the two numerical attributes are distributed as $N(\mu = 10, \sigma = 1)$; for the second class, the distributions are $N(\mu = 20, \sigma = 2)$. We run every algorithm with the same parameter setting (k = 20) 50 times on each data set and get the average. Fig. 19 shows the execution time of

two methods on the datasets. Both BP and ROBP scale linearly with the number of objects. This observation is not consistent with the previous analysis. The reason for this result may be that the computation of distance matrix and construction of RKNN graph are accelerated using vectorization in Matlab. It is remarkable that the running time of ROBP is slightly better than that of BP, especially when the data size is very large. This is in part because that Cauchy kernel is significantly more computational efficient than Gaussian kernel in the computation of the density influence.

Fig. 20 shows the execution time of two methods to cluster 2000 objects with varying parameter values (k = 10, 20, 30, 40, 50). As previously described, run every algorithm 50 times using each parameter value. Both BP and ROBP scale linearly with the parameter value. As shown in Fig. 20, the result is consistent with the previous analysis. In the experiment, the actual execution times of two methods are close, within a difference of 0.1 s.

6. Conclusion

This paper proposes a robust density-based clustering. A kernel, namely, Cauchy kernel, is used in estimate density estimation (so-called the density influence). The proposed density influence can not only reveal better the structure of clusters, but also reduce its computation time. In addition, a linkage criterion based on the shared nearest neighbors of the data points is proposed, which can avoid the problem with over-segmentation of the clusters in the BP clustering algorithm.

Experimental results on eight synthetic datasets and eleven real-world datasets show the power of the proposed algorithm. In most cases, ROBP outperforms seven competitors. Additionally, we create seven synthetic datasets and compare the robustness of ROBP and BP. Our evaluation shows that our algorithm is robust to noise and differences in cluster density, and insensitive to the parameter value. Moreover, runtime comparisons with BP illustrate our algorithm is competitive to it.

For future work, we will attempt to design an internal validation which is suitable for ROBP. Moreover, we will try to introduce the idea of parallel computing into ROBP, and further reduce its running time.

CRediT authorship contribution statement

Mingjing Du: Conceptualization, Methodology, Software, Writing - original draft. **Ru Wang:** Investigation, Data curation, Writing - original draft. **Ru Ji:** Investigation, Data curation, Writing - original draft, Visualization. **Xia Wang:** Writing - review & editing. **Yongquan Dong:** Supervision.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

We acknowledge Nadav Bar for providing their original Python code of the border-peeling algorithm. We also acknowledge many other researchers who provide the original code of their algorithms and the available datasets.

This work is partly funded by the National Natural Science Foundation of China (nos. 62006104 and 61872168), the Natural Science Foundation of the Jiangsu Higher Education Institutions (no. 20KJB520012), and the Scientific Research Foundation for Young Teachers by Jiangsu Normal University (no. 18XLRX006).

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.ins.2021.04.089.

References

[1] M.A. Abbas, A.A. Shoukry, Cmune: A clustering using mutual nearest neighbors algorithm, in: Proceedings of the 11th International Conference on Information Science, Signal Processing and their Applications, 2012, pp. 1192-1197.
[2] M. Ankerst, M.M. Breunig, H.-P. Kriegel, J. Sander, OPTICS: ordering points to identify the clustering structure, in, in: Proceedings of the ACM International Conference on Management of Data, 1999, pp. 49–60.
[3] H. Averbuch-Elor, N. Bar, D. Cohen-Or, Border-peeling clustering, IEEE Trans. Pattern Anal. Mach. Intell. 42 (2020) 1791–1797.
[4] R.J. Campello, P. Kröger, J. Sander, A. Zimek, Density-based clustering, Wiley Interdisciplinary Reviews, Data Min. Knowl. Disc. 10 (2020) e1343.
[5] R.J. Campello, D. Moulavi, J. Sander, Density-based clustering based on hierarchical density estimates, in, in: Proceedings of the 17th Pacific-Asia Conference on Knowledge Discovery and Data Mining, 2013, pp. 160–172.
[6] J. Ding, X. He, J. Yuan, B. Jiang, Automatic clustering based on density peak detection using generalized extreme value distribution, Soft. Comput. 22 (2018) 2777–2796.
[7] S. Ding, M. Du, T. Sun, X. Xu, Y. Xue, An entropy-based density peaks clustering algorithm for mixed type data employing fuzzy neighborhood, Knowl.-Based Syst. 133 (2017) 294–313.
[8] M. Du, S. Ding, H. Jia, Study on density peaks clustering based on k-nearest neighbors and principal component analysis, Knowl.-Based Syst. 99 (2016) 135–145.
[9] L. Ertöz, M. Steinbach, V. Kumar, Finding clusters of different sizes, shapes, and densities in noisy, high dimensional data, in, in: Proceedings of the 3rd SIAM International Conference on Data Mining, 2003, pp. 47–58.

[10] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in, in: Proceedings of the 2nd ACM International Conference on Knowledge Discovery and Data Mining, 1996, pp. 226–231.
[11] X. Fang, Z. Tie, Y. Guan, S. Rao, Quasi-cluster centers clustering algorithm based on potential entropy and t-distributed stochastic neighbor embedding, Soft. Comput. 23 (2019) 5645–5657.
[12] E.B. Fowlkes, C.L. Mallows, A method for comparing two hierarchical clusterings, J. Am. Stat. Assoc. 78 (1983) 553–569.
[13] Y. Geng, Q. Li, R. Zheng, F. Zhuang, R. He, N. Xiong, RECOME: A new density-based clustering algorithm using relative KNN kernel density, Inf. Sci. 436 (2018) 13–30.
[14] Z. Geng, G. Chen, Y. Han, G. Lu, F. Li, Semantic relation extraction using sequential and tree-structured LSTM with attention, Inf. Sci. 509 (2020) 183–192.
[15] Z. Geng, Q. Meng, J. Bai, J. Chen, Y. Han, Q. Wei, Z. Ouyang, A model-free Bayesian classifier, Inf. Sci. 482 (2019) 171–188.
[16] Z. Geng, Y. Zhang, Y. Han, Joint entity and relation extraction model based on rich semantics, Neurocomputing 429 (2021) 132–140.
[17] Y. Han, G. Chen, Z. Li, Z. Geng, F. Li, B. Ma, An asymmetric knowledge representation learning in manifold space, Inf. Sci. 531 (2020) 1–12.
[18] Y. Han, S. Zhang, Z. Geng, Q. Wei, Z. Ouyang, Level set based shape prior and deep learning for image segmentation, IET Image Proc. 14 (2019) 183–191.
[19] J. Huang, Q. Zhu, L. Yang, D. Cheng, Q. Wu, QCC: a novel clustering algorithm based on Quasi-Cluster Centers, Machine Learning 106 (2017) 337–357.
[20] L. Hubert, P. Arabie, Comparing partitions, J. Classif. 2 (1985) 193–218.
[21] N. Jardine, R. Sibson, The construction of hierarchic and non-hierarchic classifications, The Computer Journal 11 (1968) 177–184.
[22] G. Karypis, E.-H. Han, V. Kumar, Chameleon: Hierarchical clustering using dynamic modeling, Computer 32 (1999) 68–75.
[23] S.-S. Li, An Improved DBSCAN Algorithm Based on the Neighbor Similarity and Fast Nearest Neighbor Query, IEEE Access 8 (2020) 47468–47476.
[24] P. Lin, Z. Hong, W. Feng, Y. Li, L. Wu, Design and Implementation of an Improved DBSCAN Algorithm, in, Proceedings of IEEE 3rd Advanced Information Management, Communicates, Electronic and Automation Control Conference (2019) 1834–1839.
[25] R. Liu, H. Wang, X. Yu, Shared-nearest-neighbor-based clustering by fast search and find of density peaks, Inf. Sci. 450 (2018) 200–226.
[26] Y. Liu, Z. Li, H. Xiong, X. Gao, J. Wu, S. Wu, Understanding and enhancement of internal clustering validation measures, IEEE Trans. Cybern. 43 (2013) 982–994.
[27] A. Lotfi, P. Moradi, H.J.P.R. Beigy, Density Peaks Clustering Based on Density Backbone and Fuzzy Neighborhood, Pattern Recogn. 107 (2020) 107449.
[28] M. Lu, X.-J. Zhao, L. Zhang, F.-Z. Li, Semi-supervised concept factorization for document clustering, Inf. Sci. 331 (2016) 86–98.
[29] M. Meilă, Comparing clusterings—an information based distance, Journal of Multivariate Analysis 98 (2007) 873–895.
[30] Y. Meng, R. Shang, L. Jiao, W. Zhang, S. Yang, Dual-graph regularized non-negative matrix factorization with sparse and orthogonal constraints, Eng. Appl. Artif. Intell. 69 (2018) 24–35.
[31] F. Nie, C.-L. Wang, X. Li, K-multiple-means: A multiple-means clustering method with specified K clusters, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 959-967.
[32] Y. Pang, J. Xie, F. Nie, X. Li, Spectral clustering by joint spectral embedding and spectral rotation, IEEE Trans. Cybern. 50 (2020) 247–258.
[33] Y. Qin, Z.L. Yu, C.-D. Wang, Z. Gu, Y. Li, A Novel clustering method based on hybrid K-nearest-neighbor graph, Pattern Recogn. 74 (2018) 1–14.
[34] M.H. Rad, M. Abdolrazzagh-Nezhad, A new hybridization of DBSCAN and fuzzy earthworm optimization algorithm for data cube clustering, Soft. Comput. 24 (2020) 15529–15549.
[35] W.M. Rand, Objective criteria for the evaluation of clustering methods, J. Am. Stat. Assoc. 66 (1971) 846–850.
[36] A. Rodriguez, A. Laio, Clustering by fast search and find of density peaks, Science 344 (2014) 1492–1496.
[37] N.N. Schraudolph, A fast, compact approximation of the exponential function, Neural Comput. 11 (1999) 853–862.
[38] R. Shang, Y. Meng, W. Wang, F. Shang, L. Jiao, Local discriminative based sparse subspace learning for feature selection, Pattern Recogn. 92 (2019) 219–230.
[39] R. Shang, W. Wang, R. Stolkin, L. Jiao, Non-negative spectral learning and sparse regression-based dual-graph regularized feature selection, IEEE Trans. Cybern. 48 (2017) 793–806.
[40] R. Shang, Z. Zhang, L. Jiao, W. Wang, S. Yang, Global discriminative-based nonnegative spectral clustering, Pattern Recogn. 55 (2016) 172–182.
[41] P.H. Sneath, The application of computers to taxonomy, Microbiology 17 (1957) 201–226.
[42] L. van der Maaten, G. Hinton, Visualizing data using t-SNE, Journal of Machine Learning Research 9 (2008) 2579–2605.
[43] N.X. Vinh, J. Epps, J. Bailey, Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance, The Journal of Machine Learning Research 11 (2010) 2837–2854.
[44] G.-G. Wang, S. Deb, L.D.S. Coelho, Earthworm optimisation algorithm: a bio-inspired metaheuristic algorithm for global optimisation problems, International Journal of Bio-Inspired Computation 12 (2018) 1–22.
[45] Y. Wang, D. Wang, X. Zhang, W. Pang, C. Miao, A.-H. Tan, Y. Zhou, McDPC: multi-center density peak clustering, Neural Comput. Appl. 32 (2020) 13465–13478.
[46] W.T. Williams, J.M. Lambert, Multivariate methods in plant ecology: V. Similarity analyses and information-analysis, The, J. Ecol. (1966) 427–445.
[47] 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) 19–40.
[48] X. Zhang, Y. Sun, H. Liu, Z. Hou, F. Zhao, C. Zhang, Improved clustering algorithms for image segmentation based on non-local information and back projection, Inf. Sci. 550 (2021) 129–144.