

Dense members of local cores-based density peaks clustering algorithm[☆]

Dongdong Cheng, Sulan Zhang^{*}, Jinlong Huang

College of Big Data and Intelligent Engineering, Yangtze Normal University, Chongqing 408100, China

ARTICLE INFO

Article history:

Received 24 January 2019

Received in revised form 12 November 2019

Accepted 27 December 2019

Available online 2 January 2020

Keywords:

Dense members

Local cores

Density peaks

Clustering

ABSTRACT

An efficient clustering algorithm by fast search and find of density peaks (DP) was proposed and attracted much attention from researchers. It assumes that cluster centers are surrounded by lower density points and have a larger distance from points with higher densities. According to the characteristic of cluster centers, we can easily obtain centers from decision graph. However, DP algorithm fails to cluster manifold data sets, especially when there are a lot of noises in the manifold data sets. In this paper, we propose a dense members of local core-based density peaks clustering algorithm DLORE-DP. First, we find local cores to represent the data set. After that, only dense members of local cores are taken into consideration when computing the graph distance between local cores, avoiding the interference of noises. Then, natural neighbor-based density and the new defined graph distance are used to construct decision graph on local cores and DP algorithm is employed to cluster local cores. Finally, we assign each remaining point to the cluster its representative belongs to. The new defined graph distance helps our algorithm cluster manifold data sets and the elimination of low density points makes it more robust. Moreover, since we only calculate the graph distance between local cores, instead of all pairs of points, it greatly reduces the running time. The experimental results on synthetic and real data sets show that DLORE-DP is more effective, efficient and robust than other algorithms when clustering manifold data sets with noises.

© 2020 Elsevier B.V. All rights reserved.

1. Introduction

Clustering is an important unsupervised learning method for data analysis. Its aim is to divide objects into clusters, so that similar objects are in the same cluster while different objects are in different clusters.

Many different clustering algorithms have been proposed over the past few decades. Among them, partitioning and density-based algorithms are the most popular because of their simple principle. The typical partitioning methods include K-means [1] and K-medoids [2]. However, the selection of initial cluster centers affects their performance. In order to avoid choosing cluster centers, Frey and Dueck proposed Affinity Propagation (AP) algorithm [3], which regards all data points as potential centers. Zhang et al. [4] improved the performance of AP algorithm by introducing a constraint in the process of message passing.

However, because a data point is always assigned to the nearest center, these algorithms cannot discover arbitrary-shaped clusters.

In order to discover clusters with arbitrary shapes, density-based clustering algorithms are proposed. DBSCAN [5] defines clusters as dense regions separated by sparse regions. It detects noises in the process of clustering. Dcore [6] assumes that each cluster has a shrunken density core that roughly retains the shape of the cluster and each such core consists of a set of loosely connected local density peaks of higher density than their surroundings. RECOME [7] is a density-based clustering algorithm with relative k-nearest neighbor kernel density (RNKD). It identifies core objects with unit RNKD and partitions non-core objects into atom clusters by successively following higher density neighbor relations toward core objects. Core objects and their corresponding atom clusters are then merged through α -reachable paths on KNN graph.

In order to avoid the interference of noises, Wang et al. [8] propose a minimum spanning tree-based clustering algorithm (denoted as LOF-MST in this paper) which employs local outlier factor (LOF) method [9] to remove noises whose density factors are larger than a threshold. DBSCAN, Dcore, RECOME, LOF-MST can discover clusters with arbitrary shapes, but they have to set parameters without any prior knowledge.

[☆] No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to <https://doi.org/10.1016/j.knosys.2019.105454>.

^{*} Corresponding author.

E-mail address: slzhang@cqu.edu.cn (S. Zhang).

A novel clustering algorithm by fast search and find of density peaks (DP) [10] was proposed in Science, 2014. It is based on the idea that cluster centers are characterized by a higher density than their neighbors and a relatively large distance from points with higher densities. It can quickly and effectively identify cluster centers without iteratively calculating the objective function. However, it has to set cutoff distance to calculate densities of points, and the δ distance of decision graph computed with Euclidean distance cannot properly reflect the dissimilarity between objects on manifold. Additionally, the assignment strategy for the remaining points can create a Domino Effect, that is, once one point is assigned wrongly, there may be more points subsequently misassigned. Many improved DP algorithms [11–14] have been proposed to solve the problems of DP algorithm.

To solve the above problems, we propose an improved DP clustering algorithm based on dense members of local cores, called DLORE-DP. First, we introduce natural neighbor to compute the local density and obtain local cores, which does not need to set parameters. After that, we eliminate low density points and exploit the dense members of local cores to redefine the graph distance between local cores to avoid the interference of noises. Then, we use the local density and the redefined graph distance to construct decision graph on local cores and employ DP algorithm to cluster local cores. Finally, the remaining points (including low density points) are assigned to the cluster their representatives belong to. The experiments by comparing our method with partitioning methods (including K-means [1] and AP [3]), DP family methods (including DP [10], DPC-KNN-PCA [13] and SNN-DPC [11]), and robust clustering methods (including DBSCAN [5] and LOF-MST [8] algorithms) on synthetic data sets and real data sets show that our algorithm can faster and better deal with data sets containing manifold clusters and a lot of noises.

The rest of this paper is organized as follows. Section 2 reviews the related researches about DP algorithm and improved DP algorithms. Section 3 introduces natural neighbor and natural neighbor-based density. Section 4 presents the proposed clustering algorithm DLORE-DP and Section 5 shows the experimental results and analysis. Section 6 concludes this study.

2. Related work

2.1. DP algorithm

DP algorithm [10] assumes that cluster centers are surrounded by neighbors with lower local density and they are at a relatively large distance from any points with a higher local density. For each point i , the authors define its local density $\rho(i)$ and distance $\delta(i)$ from points of higher density. For each point i , the local density is computed with cutoff distance method shown in Eq. (1) or kernel distance method shown in Eq. (2):

$$\rho(i) = \sum_j \chi(d(i, j) - d_c) \quad (1)$$

$$\rho(i) = \sum_{i \neq j} \exp(-(\frac{d(i, j)}{d_c})^2) \quad (2)$$

where $\chi(x) = 1$ if $x < 0$ and $\chi(x) = 0$ otherwise, $d(i, j)$ is the Euclidean distance between points i and j , and d_c is a cutoff distance. It means that the local density of point i equals to the number of points whose distances to i are closer than d_c . The δ distance is defined as follows:

$$\delta(i) = \min_{j: \rho_j > \rho_i} (d(i, j)) \quad (3)$$

For the point i with the highest density, its $\delta(i)$ is defined as follows:

$$\delta(i) = \max_{i \neq j} (\delta(j)) \quad (4)$$

The definition of δ distance shows that only points with local or global maximum density have much larger δ value. DP algorithm constructs decision graph to map the data set into a two-dimensional graph w. r. t. the local density ρ and δ distance. Cluster centers stand out with anomalously large value of ρ and δ . As shown in Fig. 1, Fig. 1(a) is the decision graph of the data set shown in Fig. 1(b). The red points in Fig. 1(b) are cluster centers we select from decision graph.

After cluster centers have been found, each remaining point is assigned to the same cluster as its nearest neighbor of higher density. The DP algorithm is detailed in Algorithm 1. When given the distance matrix, the time complexity of DP is $O(N)$, where N is the number of points in a data set.

Algorithm 1: DP

Input: ρ : the density, d : the distance matrix

Output: CL : the cluster label of each point

$N = \text{size}(\rho)$;

(sorted ρ , Index ρ) = sort(ρ , 'descend');

for $i = 2 : N$ **do**

$x = \text{Index}\rho(i)$;

$p = \arg \min_{o \in \text{Index}\rho(1:i-1)} (d(x, o)) \delta(x) = d(x, p)$;

$N\text{Neighbor}(x) = p$;

end

$\delta(\text{Index}\rho(1)) = \max(\delta)$;

Construct decision graph;

Determine ρ_{\min} and δ_{\min} according to the decision graph by manually selection;

$N\text{cluster} = 1$;

for each point p **do**

if $\rho(p) > \rho_{\min}$ and $\delta(p) > \delta_{\min}$ **then**

$CL(p) = N\text{cluster}$;

$N\text{cluster} = N\text{cluster} + 1$;

end

end

for each point p **do**

$CL(p) = CL(N\text{Neighbor}(p))$;

end

Return CL ;

2.2. Improved DP algorithms

Decision graph provides us with an opportunity to better understand the data, thus DP algorithm can quickly select the preferred clustering result. However, according to the above definitions, it has to set cutoff distance to compute the local density and the assignment strategy cannot process data sets with manifold structures. Many improved DP algorithms are proposed to solve these problems.

Some algorithms employ k-nearest neighbors to solve the problem of setting cutoff distance and improve the density measure. FKNN-DPC [12] computes the local density according to k-nearest neighbors and assigns each remaining point to the most likely clusters in two steps. First, it assigns non-outliers by starting from cluster centers and breadth-first searches the k-nearest neighbors of each point. Second, it assigns outliers and the points unassigned by the technique of fuzzy weighted k-nearest neighbors. In [13], a clustering algorithm based on k-nearest

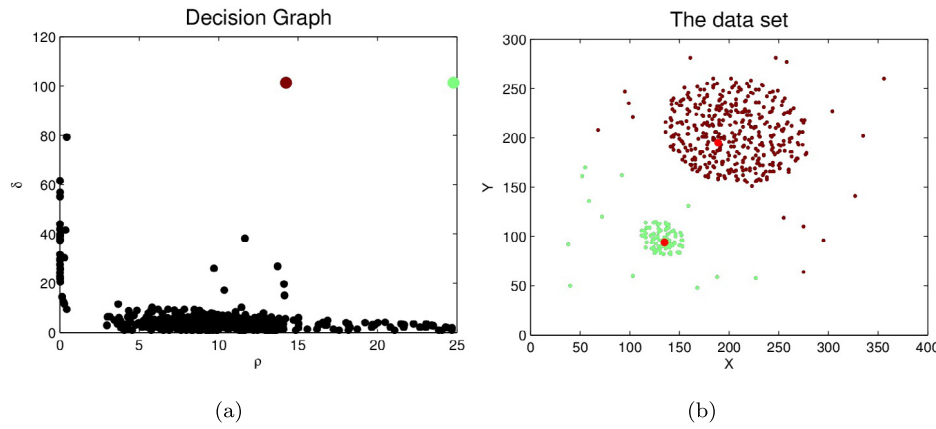


Fig. 1. DP algorithm.

neighbors and principal component analysis (PCA), named DPC-KNN-PCA, is proposed. It also introduces k -nearest neighbors to compute local density and uses PCA to process data sets with high dimensions.

Some researchers propose algorithms to optimize the selection of cluster centers and automatically determine the number of clusters. STclu [15] uses outward statistical testing to automatically determine the cluster centers. DenPEHC [14] is a hierarchical clustering algorithm, which directly generates clusters on each possible clustering layer and introduces a grid granulation framework to enable DenPEHC to cluster large-scale and high-dimensional data sets. 3DC algorithm [16] uses a divide-and-conquer strategy and the density-reachable concept in the DBSCAN framework to automatically find the correct number of clusters. In [17], the authors prove that the singular points outside the confidence interval by setting the confidence interval are cluster centers and they propose a fast cluster center determination method. QCC [18] assumes that the density of a cluster center is the highest in its k -nearest neighbors or reverse nearest neighbors. It first finds the quasi-cluster centers which correspond to initial clusters, then defines a new metric to evaluate the similarity between initial clusters and finally obtain the clustering results by merging the initial clusters.

Some improved DP algorithms are proposed to solve the problem of discovering manifold clusters. SNN-DPC [11] defines SNN similarity which takes the information of the nearest neighbors and the shared neighbors into account when computing the local density and δ distance. Then it employs a two-step allocation method: inevitable subordinate and possible subordinate. The former quickly and accurately recognizes and assigns the points certainly belong to one cluster by counting the number of shared neighbors between two points. The latter assigns the remaining points by finding the clusters to which more neighbors belong. NaNLORE [19] first finds local representatives and introduces density-adaptive distance to measure the distance between local representatives. The density-adaptive distance between two local representatives is the shortest path on weighted k -nearest neighbor graph of the whole data set. Then, it constructs decision graph on local representatives according to the new defined distance, which helps to solve the problem of clustering data sets with complex structures.

There are also some researchers apply DP algorithm to image processing [20], community detection [21–23], extracting multi-document abstracts [24] and noise removal [25].

3. Preliminaries

3.1. Natural neighbor

K -nearest neighbors are utilized to evaluate the local density of points in many algorithms, but how to set k value is a key problem. Natural neighbor [26] is a new neighbor concept. It can adapt to the distribution of data sets and obtain the k value automatically. Its application in clustering analysis [19], instance reduction [27] and outlier detection [28] illustrates its effectiveness. The idea of natural neighbor is inspired by the reality that the friendship between two objects should be mutual. If everyone has at least one friend or the number of persons who take him or her as friends does not change, we call it reaches natural stable state.

Given a data set D , let $d(p, q)$ denote the Euclidean distance between point p and q . Suppose o is the k th nearest neighbor of point p . The definitions of k -nearest neighbors and reverse k -nearest neighbors are as follows.

Definition 1 (k -Nearest Neighbors). The k -nearest neighbors of point p are a set of point x that satisfies $d(p, x) \leq d(p, o)$, that is, $NN_k(p) = \{x \in D | d(p, x) \leq d(p, o)\}$

Definition 2 (Reverse k -Nearest Neighbors). The reverse k -nearest neighbors of point p are a set of point x who considers p as one of its k -nearest neighbors, that is, $RNN_k(p) = \{x \in D | p \in NN_k(x)\}$.

The process of obtaining natural neighbor information is as follows: we initialize r with 1; in each iteration, we search r th-nearest neighbor and count $nb(p)$ the number of reverse neighbors for each point p , and after that we add one to the value of r ; when the number of points without reverse neighbors does not change, iteration terminates and r at this moment is called natural characteristic value λ ; finally, it returns λ and λ -nearest neighbors of points. The detailed algorithm is shown in Algorithm 2.

When introducing KD-tree to search r -nearest neighbor, the time complexity of NaN-Searching can be reduced to $O(N \log N)$ (N is the number of objects in a data set), otherwise, its time complexity is $O(N^2)$.

3.2. Natural neighbor-based density

The density of points in a dense region is usually larger than that in a sparse region, while the sum of the distances between a point and its neighbors is smaller in a dense region than that in a sparse region. For two points, we find the same number of neighbors, the larger the sum of distances is, the sparser the region is.

Algorithm 2: NaN-Searching

Input: D: the data set
Output: λ , NN_λ
 Initializing: $r = 1$, $nb(i) = 0$, $NN_0(i) = \phi$, $RNN_0(i) = \phi$;
while true do
 for each data point p in D do
 Find the r th neighbor q of p ;
 $nb(q) = nb(q) + 1$;
 $NN_r(p) = NN_{r-1}(p) \cup \{q\}$;
 $RNN_r(q) = RNN_{r-1}(q) \cup \{p\}$;
 end
 Compute the number of points with no neighbor (i.e.,
 $nb(p) = 0$) Numb;
if the number Numb does not change then
 Break;
end
 $r = r + 1$;
end
 $\lambda = r$;
 Output the λ , NN_λ ;

point lies, and vice versa. STclu [15] defines k-density as the ratio of k and the sum of distances with its k -nearest neighbors and proves that k-density performs better in cluster center detection than that in [10]. The natural neighbor-based density is defined in a similar way, and the local density of point p is computed as shown in Eq. (5):

$$Den(p) = \frac{k}{\sum_{x \in NN_k(p)} d(p, x)} \quad (5)$$

where k is the maximum value of nb . Different from definition in [15], the new defined local density does not need to set parameter k , because the process of obtaining natural neighbor information is committed to find the appropriate parameter k .

4. The proposed algorithm

Most existing DP-based algorithms cannot discover manifold clusters, not to mention that a data set containing manifold clusters and a large number of noises at the same time. Some researchers [29] suggest using geodesic distance to describe the relationship on manifold clusters. In [30], the authors point out that geodesic distance can be approximated with graph distance when there are enough samples from the manifold. The graph distance is the shortest path between two points on a graph. The algorithms in [19,29,31] employ k-nearest neighbor graph to compute graph distance. However, calculating the shortest path on the whole data set is time consuming. In this paper, we propose a novel DP-Based clustering algorithm, called DLORE-DP, which only computes the graph distance between local cores to solve the problem of time consuming when detecting manifold clusters and only takes dense members of local cores into consideration to avoid the influence of noises.

4.1. Local cores

Local cores are obtained on the basis of local neighbors. First, we give some related definitions.

Definition 3 (Local Neighbors). Local neighbors of p is the λ -nearest neighbors, that is, $LN(p) = NN_\lambda(p)$.

Definition 4 (Representative). If the density of point q is the maximum among p and its local neighbors, then q is the representative of p and its local neighbors. We denote it as $Rep(p) = q$.

According to Definition 4, there exists a situation that a point has two or more representatives at the same time, then how to choose the representative is determined by representative competition rule. In addition, if there is a chain in the representative relation, that is, the representative of p is q , and the representative of q is r , then we will use representative transfer rule to determine the final representative of p . The detailed definitions are as follows.

Representative competition rule (RCR) For point p , if $Rep(p) = R1$ and $Rep(p) = R2$ at the same time, then $Rep(p) = \arg \min_{x \in \{R1, R2\}} \{d(p, x)\}$, that is, the representative closer to point p will be the final representative of p .

Representative transfer rule (RTR) If $Rep(p) = q$, and $Rep(q) = r$, then $Rep(p) = r$.

Definition 5 (Local Core). After updating representatives of points according to RCR and RTR, point p is a local core if $Rep(p) = p$. We denote it as $LORE = \{p | Rep(p) = p\}$.

RCR helps each object to find a more reasonable representative. RTR ensures each object selects a point within a region as large as possible as its representative, which avoids obtaining too many local cores. Local cores are points with local maximum density. The local cores searching algorithm (LORE-Searching) is detailed in Algorithm 3.

Algorithm 3: LORE-Searching

Input: $LN(p)$: the local neighbors of each point p , $Den(p)$: the natural neighbor-based density of each point p
Output: LORE: the local cores, Rep: the representative
 Initializing: $Rep(i) = \phi$, $LORE = \phi$;
for each point i in the data set do
 $y = \arg \max_{x \in LN(i)} (Den(x))$;
 for each point p in LN(i) do
 if Rep(p) == ϕ then
 $Rep(p) = y$;
 end
 if Rep(p) == x and $x \neq y$ then
 Determine $Rep(p)$ according to RCR;
 end
 for each point z in the data set do
 if Rep(z) == p then
 Determine $Rep(z)$ according to RTR;
 end
 end
 end
 $K = 1$;
for each point p in the data set do
 if Rep(p) == p then
 $LORE(K) = p$;
 $K = K + 1$;
 end
end

4.2. Graph distance with dense members of local cores

The graph distance in this paper is not based on a k-nearest neighbor graph, but on a shared neighbors-based distance matrix of local cores. In order to avoid the influence of noises, when

computing shared neighbors-based distance between local cores, we need to eliminate low density points and only keep dense points. Therefore, we determine a density threshold $den_threshold$ to distinguish low density points and high density points. The density threshold is obtained by setting a ratio of low density points to all points α . That is, we sort points in ascending order by the density and the first $\alpha * N$ points are low density points (N is the number of points in a data set). In Section 5.3, we will discuss how to select α .

Definition 6 (*Dense Members of Local Core*). For a local core p , its dense members are a set of point x whose representative is p and $Den(x) \geq den_threshold$. We denote it as $DMLORE(p) = \{x | Rep(x) = p \cap Den(x) \geq den_threshold\}$.

Definition 7 (*Dense Neighbors of Local Core*). For a local core p , its dense neighbors are the union of λ nearest neighbors of dense members of p . We denote it as $DNLORE(p) = \cup_{q \in DMLORE(p)} NN_{\lambda}(q)$.

Definition 8 (*Dense Shared Neighbors Between Local Cores*). Dense shared neighbors of two local cores p and q are the intersection of their dense neighbors. We denote it as $DSLORE(p, q) = DNLORE(p) \cap DNLORE(q)$.

Definition 9 (*Dense Shared Neighbors-Based Distance*). For local cores p and q , their dense shared neighbors-based distance is computed as Eq. (6).

$$SD(p, q) = \begin{cases} \frac{d(p, q)}{|DSLORE(p, q)| \times \sum_{o \in DSLORE(p, q)} Den(o)} & , \text{ if } |DSLORE(p, q)| \neq 0 \\ maxd & , \text{ otherwise} \end{cases} \quad (6)$$

where $d(p, q)$ is the Euclidean distance between local cores p and q , $Den(o)$ is the natural neighbor-based density of point o and $maxd$ is the maximum value of Euclidean distance between all pairs of local cores.

In order to capture the intrinsic geometric features of manifold data set, we define the graph distance between local cores. Given the shared neighbors-based distance matrix, let $P = \{p_1, p_2, \dots, p_m\}$ represent the shortest path from p_1 (i.e., p) to p_m (i.e., q), $1 \leq k < m$. Then, the graph distance between local cores p and q is computed as Eq. (7).

$$GD(p, q) = \sum_{k=1}^{m-1} SD(p_k, p_{k+1}) \quad (7)$$

where p_k and p_{k+1} are the local cores along the shortest path P .

4.3. Clustering local cores with DP

Like DP algorithm, we construct decision graph on local cores with respect to their density ρ and δ distance. The cluster centers are selected according to the decision graph. First, we redefine the density ρ and the δ distance for each local core p . The density $\rho(p)$ is its natural neighbor-based density, which is computed as

$$\rho(p) = Den(p) \quad (8)$$

Its $\delta(p)$ distance is computed as

$$\delta(p) = \min_{q \in LORE, \rho(q) > \rho(p)} GD(p, q) \quad (9)$$

For the local cores p with highest density, its δ distance is computed as:

$$\delta(p) = \max_{o \in LORE, o \neq p} (\delta(o)) \quad (10)$$

In fact, the decision graph provides us with an opportunity to better understand the data and choose the preferred clustering result, it is better to retain this user-algorithm interaction. So we manually select cluster centers from the decision graph and the local cores with relatively large density ρ and δ distance are selected as cluster centers. After cluster centers have been found, each remaining local core is assigned to the same cluster as the local core of higher density and with the minimum graph distance.

Since we only take dense members of local cores into consideration when computing the shared neighbors-based distance between local cores, it rules out noises and helps us to obtain correct clustering result. Take a data set with noises shown in Fig. 2 for example, the first row is the result of our method without eliminating low density points and the second row is the result of our method eliminating low density points. From the results, we can learn that if we do not rule out the low density points, the graph distances between local cores R3 and R1, R3 and R2 are shortened because they have shared neighbors enclosed in the red box, thus the decision graph in Fig. 2(b) does not correctly reflect the relationship between local cores on manifold and the cluster R3 belonging to is assigned the cluster R1 belonging to. However, when we eliminate the low density points, the graph distance between R3 and R1, R3 and R2 are expanded because there are no shared neighbors between them, therefore, the decision graph in Fig. 2(e) correctly reflects the relationship between local cores on manifold and we get correct clustering result for local cores.

4.4. Dense neighbors of local cores-based DP algorithm (DLORE-DP)

Based on the above definitions, we propose a novel dense members of local cores-based DP algorithm (DLORE-DP). The main steps of DLORE-DP include: first find local cores and obtain the dense members of local cores; then define the shared neighbors-based distance between local cores according the dense members of local cores and compute the graph distance with the shared neighbors-based distance; after that employ DP algorithm to cluster local cores; and finally obtain the clustering results for the original data sets by assigning the remaining points (including low density points) to their corresponding representatives. The DLORE-DP algorithm is detailed in Algorithm 4, where NaN-Searching() is the natural neighbor searching algorithm described in Algorithm 2, LORE-Searching() is the local cores searching algorithm described in Algorithm 3 and DP() is the DP algorithm described in Algorithm 1.

The proposed algorithm DLORE-DP mainly contain three steps: find local cores according to the result of NaN-Searching algorithm, compute the graph distance between local cores and employ DP algorithm to cluster local cores. The time complexity of NaN-Searching algorithm is $O(N \log N)$ and LORE algorithm is $O(N)$, thus the time complexity of the first step is $O(N \log N)$. Assuming the number of local cores is N_l ($N_l \ll N$), the time complexity of the second step is $O(N_l^2)$. Since we only cluster local cores with DP algorithm, its time complexity is $O(N_l)$. The overall time complexity of DLORE-DP algorithm is $O(N \log N)$.

5. Experimental analysis

In order to evaluate the performance of the proposed method DLORE-DP, our experiment includes three parts. First, we compare the decision graphs of our method with that of DP and SNN-DPC algorithms to show the advantages of our algorithm in discovering manifold clusters. Then, we do experiments on several synthetic data sets and real data sets from UCI to further

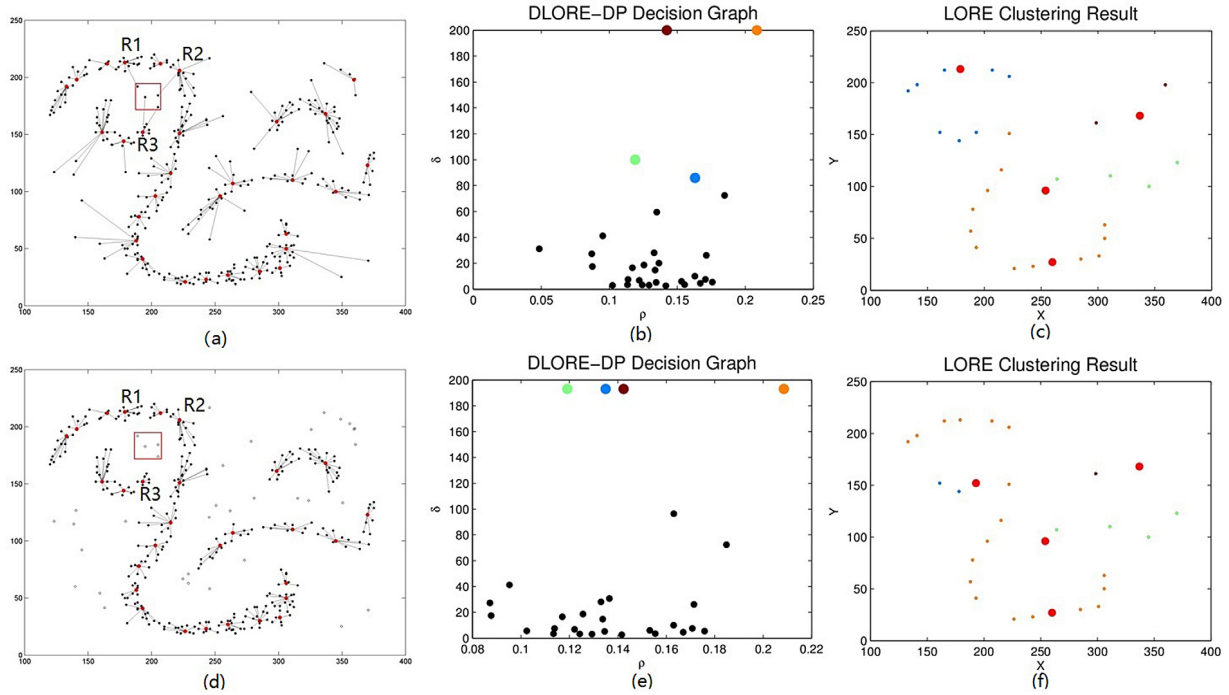


Fig. 2. The comparison of decision graphs for a data set with noises. (a) The members of local cores without eliminating low density points. (b) and (c) are the decision graph and the clustering result on local cores corresponding to (a), respectively. (d) The members of local cores eliminating low density points. (e) and (f) are the decision graph and the clustering result on local cores corresponding to (d), respectively.

Algorithm 4: DLORE-DP

Input: D : the data set, α : the ratio of low density points
Output: CL : the cluster labels
 $N = \text{size}(D)$;
 $(\lambda, LN) = \text{NaN-Searching}(D)$;
 Compute the density $Den(p)$ for each point p according to Eq. (5);
 $(LORE, Rep) = \text{LORE-Searching}(LN, Den)$;
 $SortedDen = \text{Sort}(Den, \text{ascend})$;
 $den_threshold = SortedDen(\alpha * N)$;
for each local core R with $Den(R) < den_threshold$ **do**
 Find the nearest local core R' with $Den(R') \geq den_threshold$;
 $Rep(Rep == R) = R'$;
end
for $i = 1:K$ **do**
 $DMLORE(i) = \text{find}(Rep = LORE(i) \& \& Den \geq den_threshold)$;
end
 Compute the dense shared neighbors-based distance matrix SD according to Eq. (6);
 Compute the graph-based distance matrix GD between local cores according to Eq. (7);
 $LORE_CL = DP(Den(LORE), GD)$;
for each point x in D **do**
 $CL(x) = CL(Rep(x))$;
end

demonstrate the effectiveness of our method. Finally, we do experiments on several data sets which consist of complex clusters and a lot of noises to demonstrate the robustness of our method.

We use accuracy (ACC) and normalized mutual information (NMI) [32] to evaluate the clustering performance. ACC and NMI both range from 0 to 1. A higher value means a better clustering result. The value of 0 means that the clustering result is useless and the value of 1 means that the clustering result perfectly

matches the class label vector. The configuration of the computer used in our experiment is as follows: processor is Intel Core i7 3.6 GHz; memory size is 16GB; programming environment is MATLAB R2013a.¹

5.1. The comparison of decision graphs on manifold data sets

We first compare the decision graphs of DP, SNN-DPC with that of our method on two synthetic data sets (Jain and Db2). Jain is from [33], involves two concave clusters with large variation in density, a total of 373 points. Db2 is from [5], consists of four manifold clusters, a total of 315 points. For DP algorithm, the cutoff distance of DP is set to the 2%th shortest distance, which is suggested by [10], so that the average number of neighbors is around 1% to 2% of the total number of points in the data set and the local density is computed with the kernel distance method. For SNN-DPC, k is set to 9, which is also suggested by the authors. For DLORE-DP, the ratio of low density point α is set to 0. The results are shown in Figs. 3 and 4, where the red points are the selected cluster centers.

As for Jain, DP selects two centers both from the dense cluster and no center from the sparse cluster, thus it does not obtain the desired clustering result; SNN-DPC and DLORE-DP correctly find two centers and discover the two clusters. As for Db2, DP and SNN-DPC both choose three centers from the long concave cluster and the long concave cluster is divided into three clusters, while DLORE-DPC correctly selects four centers from the four clusters and consequently obtains correct clustering result. The results show that DP cannot apply to manifold data sets; SNN-DPC redefines local density and δ distance using shared-neighbors, which helps it apply to manifold data sets with simple structures, but it still has difficulty in processing data sets with complex structures; DLORE-DP uses the dense shared neighbors-based distance to calculate the graph distance which correctly

¹ The MATLAB code is available from <https://github.com/DongdongCheng/DLORE-DP>.

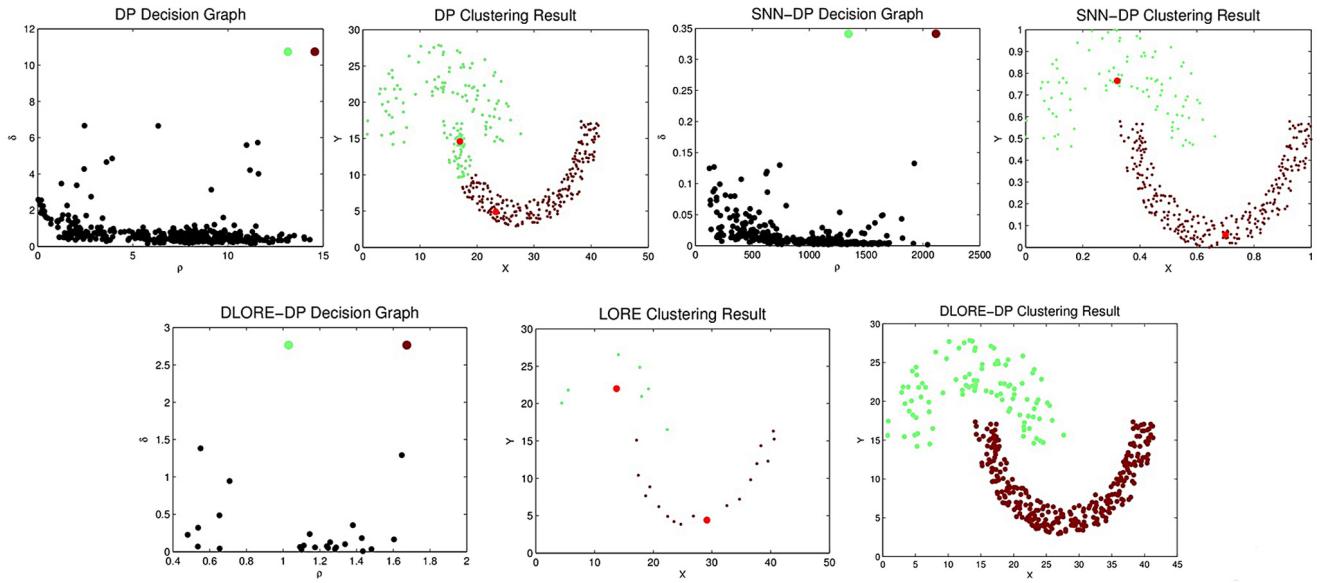


Fig. 3. The comparison of decision graphs on Jain.

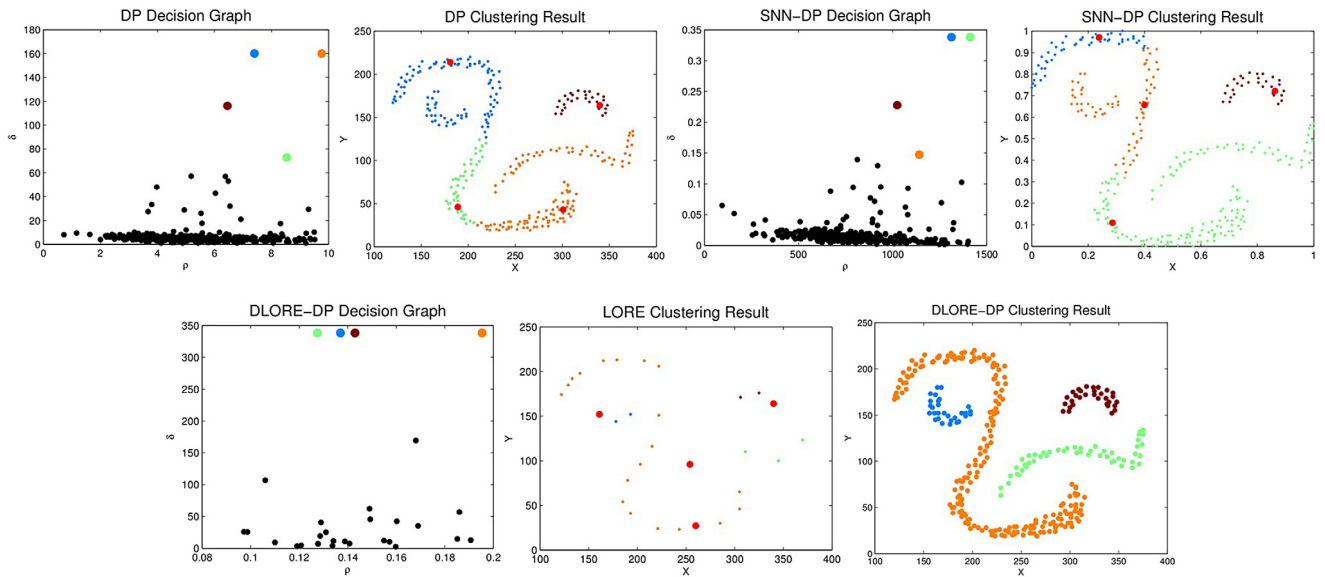


Fig. 4. The comparison of decision graphs on Db2.

reflects the dissimilarity between points on manifold clusters and makes it suitable for dealing with complex manifold data sets.

5.2. Experiments on synthetic data sets and real data sets

In order to further demonstrate the effectiveness of DLORE-DP in discovering complex structured clusters, we first do experiments on four synthetic data sets containing spherical clusters, manifold clusters and few noises by comparing the proposed method with K-means, AP, DP, DPC-KNN-PCA and SNN-DPC algorithms. Since there are only few noises in these data sets, the ratio of low density point α of DLORE-DP is also set to 0, which means that the noises are not treated specially. For K-means, the initial centers are selected randomly. For AP, the preference is set to the median value. The parameters of DP, DPC-KNN-PCA, and SNN-DPC are set to the same value as the previous experiment settings.

The synthetic data sets are shown in Fig. 5. The characteristic of synthetic data sets are listed as Table 1. The results of K-means

(see Fig. 6), AP (see Fig. 7), DP (see Fig. 8), DPC-KNN-PCA (see Fig. 9) and SNN-DPC (see Fig. 10) and DLORE-DP (see Fig. 11) are shown in Figs. 6–11 and the comparison of ACC, NMI and time is presented in Table 2.

Seen from the results, we find that K-means and AP algorithms cannot identify nonspherical clusters. Therefore, the ACC and NMI scores of K-means and AP for the four data sets are not high. DP, DPC-KNN-PCA correctly discover the clusters in Dataset 1. However, when there are complex clusters or even few noises in the data sets, like Datasets 2, 3 and 4, they fails to find the correct clusters, because they cannot choose right centers according to the decision graph. SNN-DPC has a capacity to cluster manifold data sets, like Dataset 2, but when processing data sets with complex structures or long concave clusters, it makes mistakes. The clustering results and ACC and NMI scores show that DLORE-DP successfully find the clusters contained in these synthetic data sets, which proves that DLORE-DP does well in discovering clusters with complex structures. Besides, the running time of DLORE-DP is far less than that of SNN-DPC. In terms of the

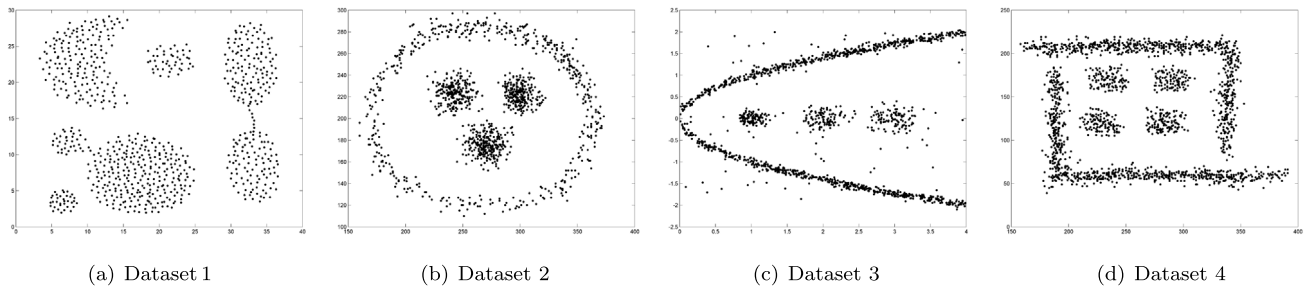


Fig. 5. The synthetic data sets.

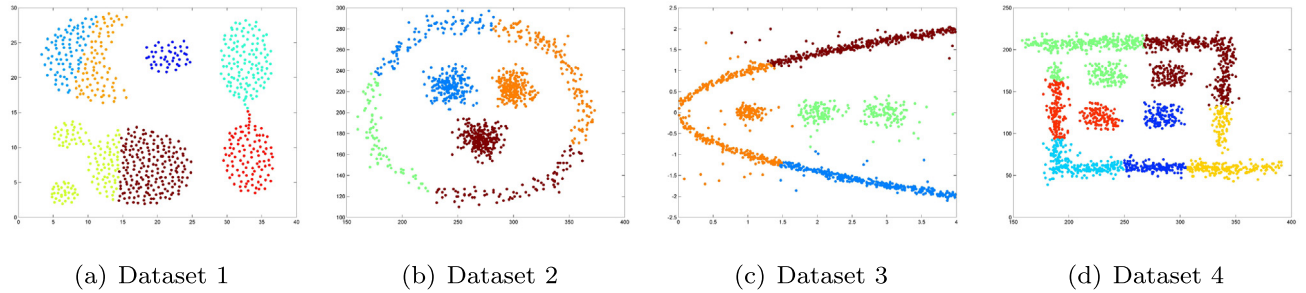


Fig. 6. The clustering results of K-means.

Table 1
The synthetic data sets.

Dataset	Instances	Attributes	Clusters	Source
Dataset 1	788	2	7	[34]
Dataset 2	1156	2	4	[35]
Dataset 3	1378	2	4	[36]
Dataset 4	1741	2	6	[35]

Table 2
The comparison of ACC, NMI and time on synthetic data sets.

		Dataset 1	Dataset 2	Dataset 3	Dataset 4
K-means	ACC	0.76	0.68	0.69	0.49
	NMI	0.85	0.57	0.67	0.52
	Time(s)	0.03	0.04	0.04	0.05
AP	ACC	0.76	0.58	0.4	0.38
	NMI	0.78	0.51	0.65	0.56
	Time(s)	5.54	8.67	7.41	13.64
DP	ACC	1.00	0.72	0.39	0.65
	NMI	1.00	0.68	0.28	0.68
	Time(s)	0.16	0.26	0.38	0.59
DPC-KNN-PCA	ACC	0.99	0.60	0.74	0.49
	NMI	0.99	0.57	0.83	0.44
	Time(s)	0.14	0.18	0.31	0.52
SNN-DPC	ACC	0.97	1.00	0.37	0.67
	NMI	0.95	1.00	0.21	0.72
	Time(s)	32.57	69.09	99.56	158.83
DLORE-DP	ACC	0.99	1.00	1.00	1.00
	NMI	0.99	1.00	1.00	1.00
	Time(s)	0.68	0.34	0.71	1.02

Table 3
The real data sets from UCI.

Datasets	Instances	Attributes	Clusters
Iris	150	4	3
Wine	178	13	3
Seed	210	7	3
Ionosphere	351	34	2
Cancer	569	30	2
Control	600	60	6

Table 4
The comparison of ACC, NMI and time on real data sets.

		Iris	Wine	Seed	Ionosphere	Cancer	Control	Average
DP	ACC	0.91	0.98	0.89	0.68	0.80	0.56	0.80
	NMI	0.81	0.91	0.70	0.09	0.35	0.75	0.60
	Time(s)	0.04	0.04	0.04	0.05	0.12	0.18	
SNN-DPC	ACC	0.95	0.71	0.78	0.56	0.94	0.44	0.73
	NMI	0.84	0.63	0.63	0.07	0.68	0.67	0.59
	Time(s)	1.21	1.71	2.36	6.54	17.20	19.18	
DLORE-DP	ACC	0.91	0.97	0.84	0.71	0.86	0.58	0.81
	NMI	0.81	0.88	0.63	0.28	0.48	0.81	0.65
	Time(s)	0.05	0.04	0.05	0.07	0.13	0.19	

performance on recognizing complex clusters and the running time, DLORE-DP outperforms the compared algorithms.

From the above experiments, we can learn that DP and SNN-DPC perform better than K-means, AP and DPC-KNN-PCA. Thus, we choose DP and SNN-DPC to compare with our algorithm on several benchmarking real data sets from UCI, including Iris, Wine, Seed, Ionosphere, Cancer and Control. The details of these data sets are shown in Table 3. The comparison of ACC, NMI and time is illustrated in Table 4. The best results are shown

in bold. From the results, we can learn that the ACC and NMI scores of DLORE-DP are the highest for Ionosphere and control. For Iris, wine, Seed and Cancer, DLORE-DP gets the second best results. The average ACC and NMI scores of our algorithm on real data sets are the highest. Especially, Ionosphere and Control have more complex cluster structures than other data sets and our method get the best results, which further illustrates the effectiveness of our method. Besides, the running time of DLORE-DP is a little longer than DP algorithm and much less than that of SNN-DPC. Therefore, in terms of the performance and running time, DLORE-DP is more effective and efficient than other algorithms.

5.3. Experiments on data sets with noises

In order to illustrate the robustness of our algorithm, we compare the proposed method with DBSCAN and LOF-MST algorithms. DBSCAN algorithm defines clusters as dense regions

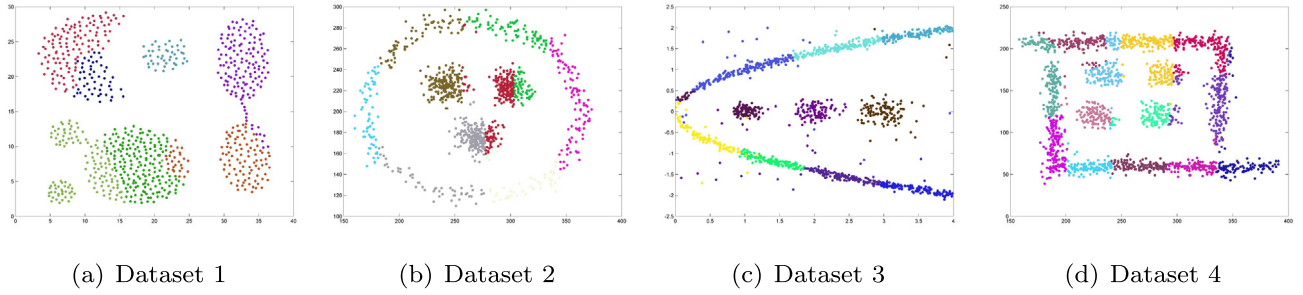


Fig. 7. The clustering results of AP.

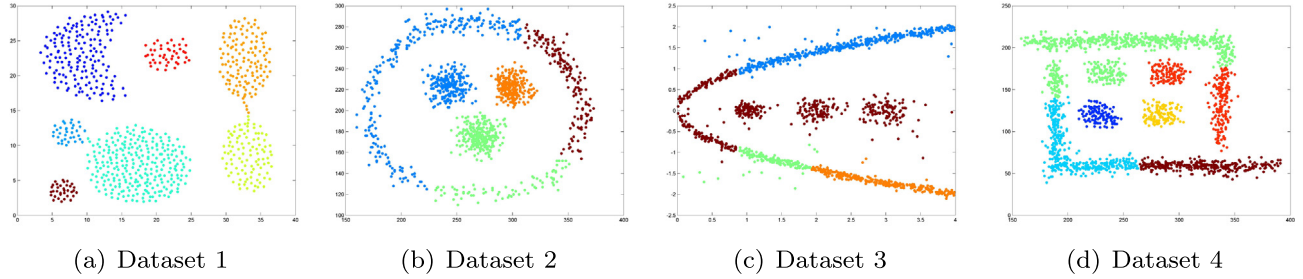


Fig. 8. The clustering results of DP.

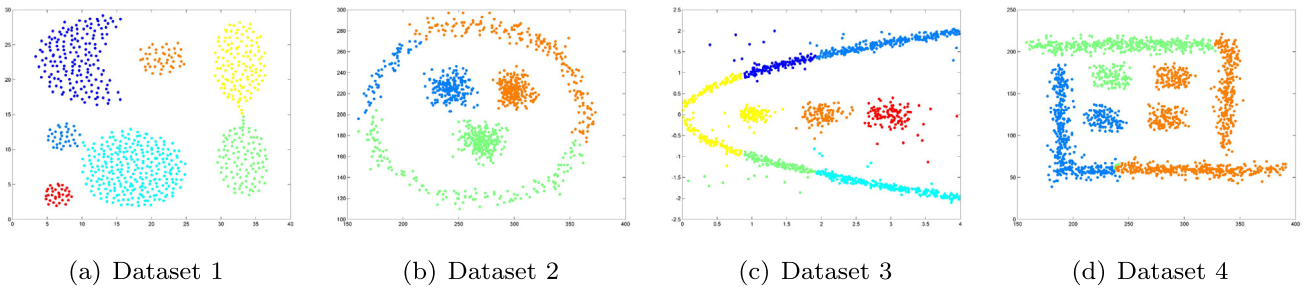


Fig. 9. The clustering results of DPC-KNN-PCA.

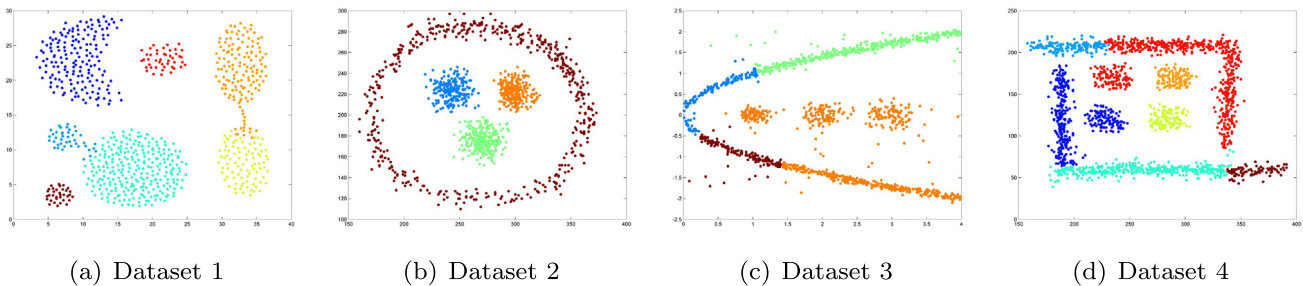


Fig. 10. The clustering results of SNN-DPC.

separated by sparse regions and detects low density points as noises in the process of clustering, therefore, it performs well in discovering arbitrary shaped clusters and is robust to noises. LOF-MST algorithm enhances clustering results by outlier detection method LOF. In this section, we compare DLORE-DP with them on three synthetic data sets containing complex structured clusters and a lot of noises, shown in Fig. 12. Datasets 5 and 6 are both from [37], including 8000 points. Dataset 7 is also used in [19], a total of 8537 points.

DBSCAN algorithm has to set two parameters: the radius Eps and the minimum number of point $MinPts$. LOF-MST algorithm has to set parameters k , the threshold of LOF value α to remove noises and the desired cluster number NC . In the experiment, we

tune the parameters to obtain the best clustering results. The best results of DBSCAN are shown in Fig. 13 and the corresponding parameters are set as $Eps = 5.4$, $MinPts = 4$ for Dataset 5, $Eps = 7.9$, $MinPts = 5$ for Dataset 6, and $Eps = 0.7$, $MinPts = 8$ for Dataset 7. The best results of LOF-MST are shown in Fig. 14 and the corresponding parameters are set as $k = 30$, $\alpha = 0.2$, $NC = 6$ for Dataset 5, $k = 30$, $\alpha = 0.27$, $NC = 9$ for Dataset 6 and $k = 30$, $\alpha = 0.15$, $NC = 7$ for Dataset 7. For DLORE-DP, we only need to set the ratio of low density points α . The values of α are all set to 0.15 for the three data sets and the corresponding clustering results are shown in Fig. 15. The ACC, NMI and running time are shown in Table 5.

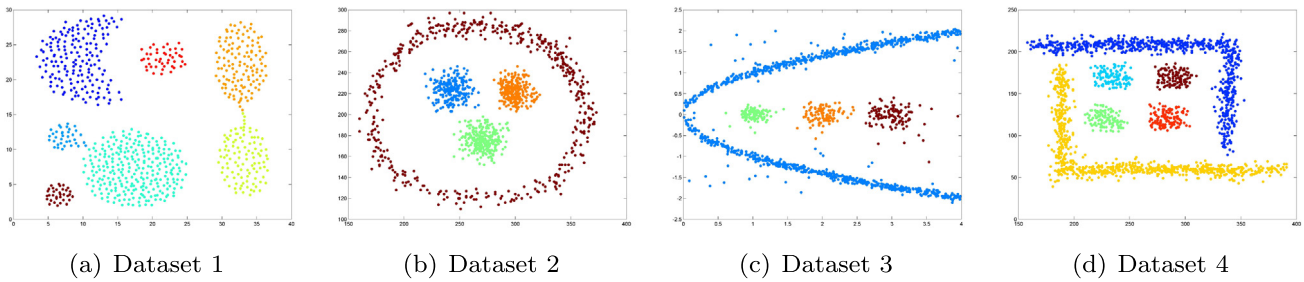


Fig. 11. The clustering results of DLORE-DP.

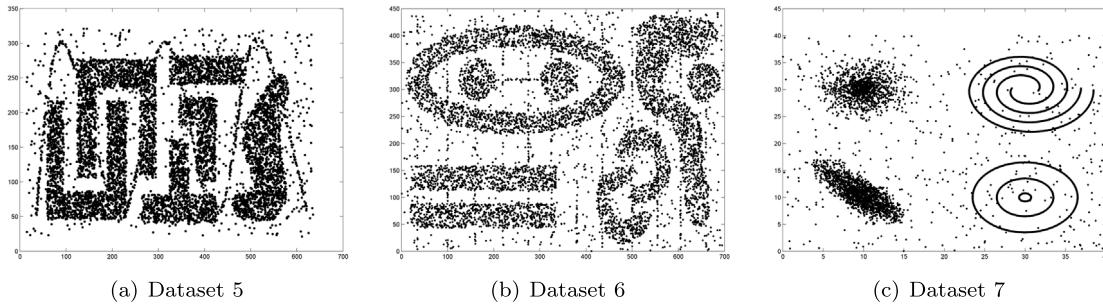


Fig. 12. The data sets with noises.

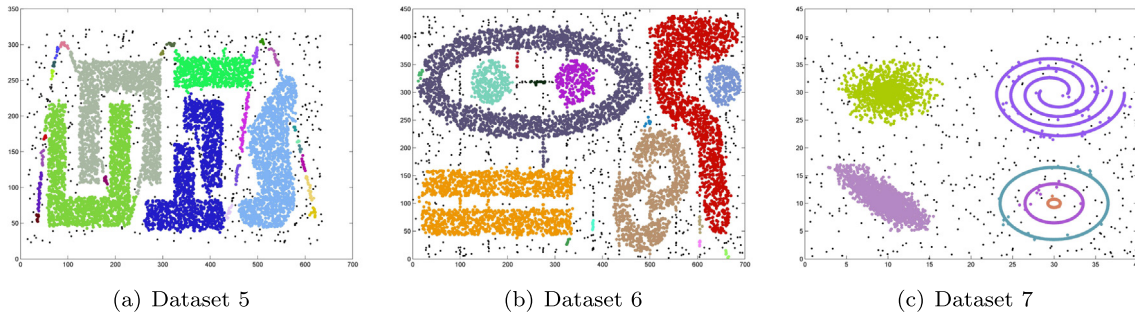


Fig. 13. The clustering results of DBSCAN on data sets with noises.

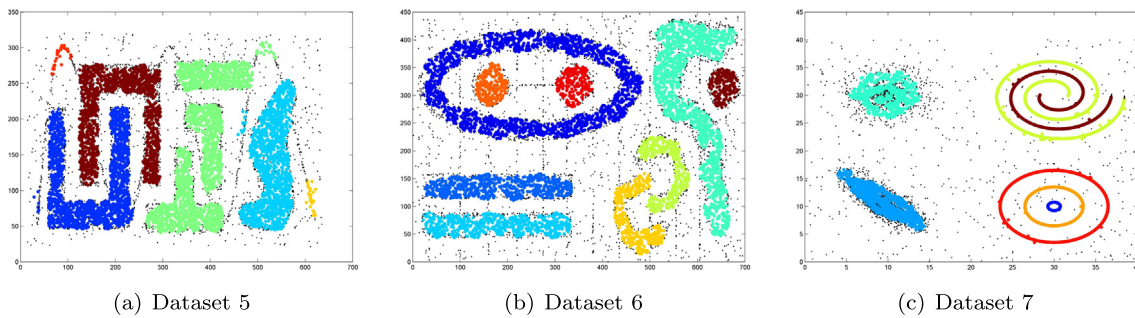


Fig. 14. The clustering results of LOF-MST on data sets with noises.

From Fig. 13, DBSCAN algorithm detects most noises in these data sets. However, we find that DBSCAN tends to detect some noises as small clusters and the clusters close to each other are identified as one cluster. Fig. 14 shows that LOF-MST algorithm removes noises so that the boundary between clusters is clear, but it is easy to find that the middle three clusters in Dataset 5 are still recognized as one cluster and a lot of normal points are also detected as noises in the three data sets. From Fig. 15 and Table 5, the proposed algorithm correctly finds the complex structured clusters in these data sets although there are a lot of noises. This is because we only take the dense members of local cores

into consideration when computing graph distance between local cores, which helps to eliminate the interference of noises. It is worth noting that although we eliminate low density points when calculating graph distance, after clustering local cores, we still assign the remaining points (including low density points) to the cluster their representatives belong to. Therefore, we obtain clustering results on the original data set, not the data set after eliminating low density points.

Although our algorithm need to set the ratio of low density points α , in fact its setting is simple. In general, we believe that the number of noises in a data set will not exceed 20%, so it

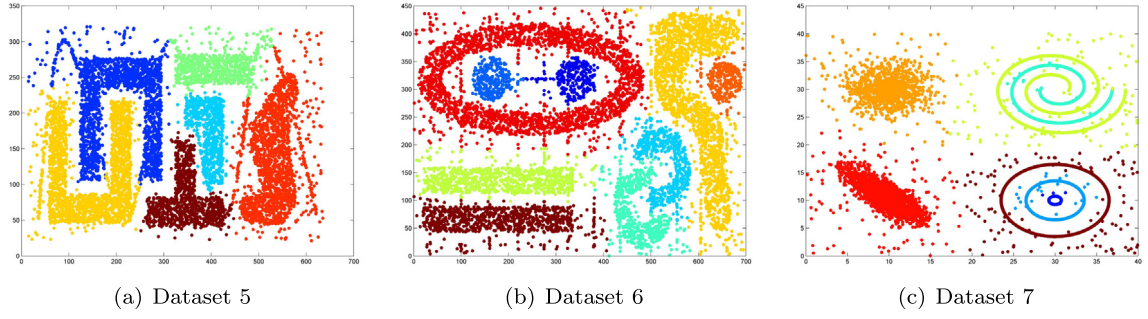


Fig. 15. The clustering results of DLORE-DP on data sets with noises.

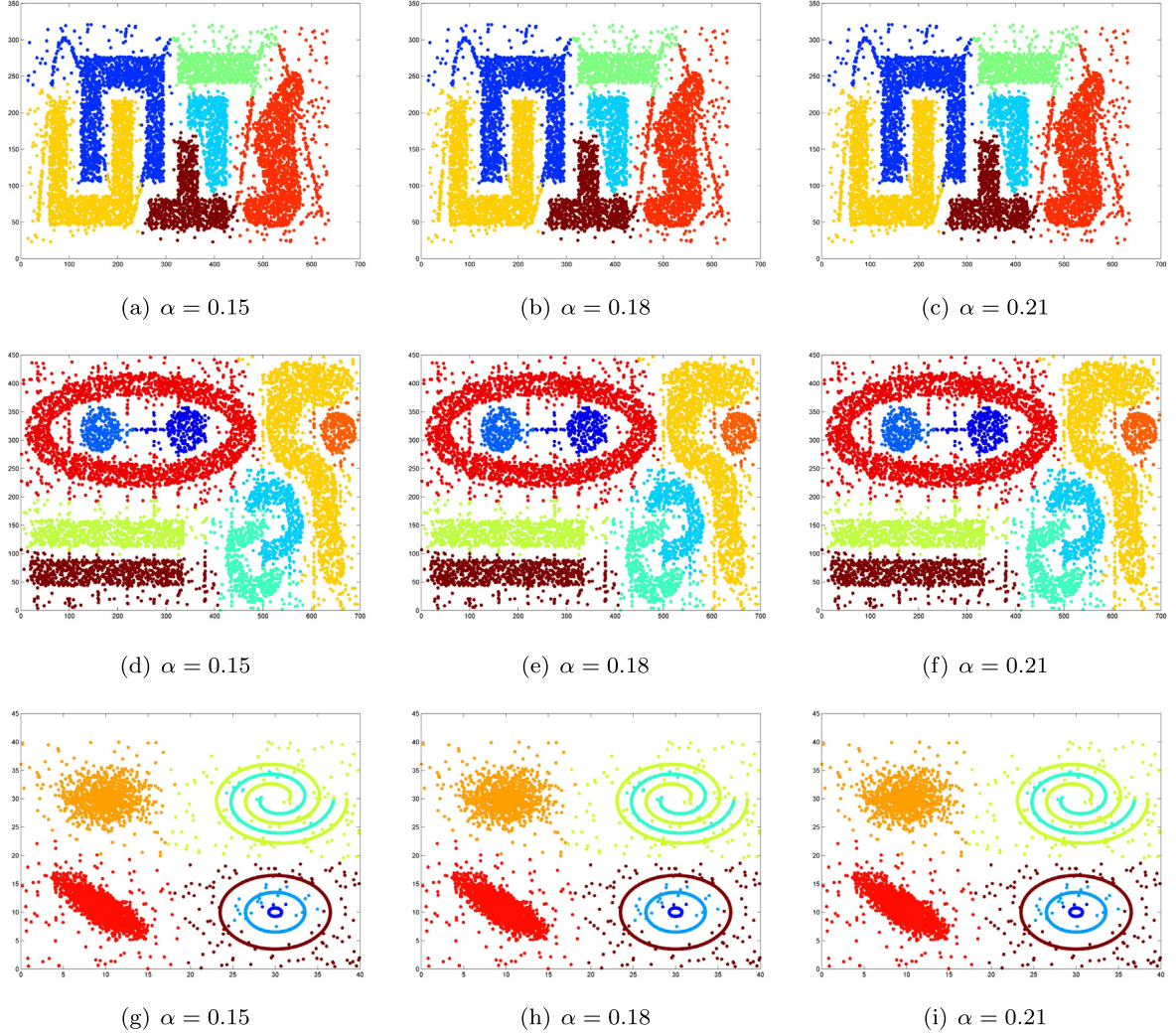


Fig. 16. The robustness experiments of DLORE-DP with respect to α .

ranges from 0 to 0.2. When there are few noises, we can set α to 0. Moreover, our algorithm is robust to α , which means that when the value of α is within a certain range, our algorithm will get the same clustering result. In order to demonstrate this, we do experiment on Datasets 5, 6, 7 by ranging α from 0.15 to 0.21, the results are shown in Fig. 16. We can learn from the results that when α is within a certain range, our algorithm obtains the desired clustering results for the three data sets.

6. Conclusion

In this paper, we propose a dense members of local cores-based density peaks algorithm DLORE-DP. First, we get local cores and find its dense members and dense neighbors. After that, we compute the graph distance with dense shared neighbors-based distance between local cores. The shared neighbors are the intersection of dense neighbors, which eliminates the interference of noises. After that, we employ DP algorithm to cluster local cores. Finally, the remaining points (including low density points) are assigned to the clusters their representatives belong to. Since we

Table 5

The comparison of ACC, NMI and time on data sets with noises.

		Dataset 5	Dataset 6	Dataset 7
DBSCAN	ACC	0.92	0.77	0.79
	NMI	0.87	0.83	0.86
	Time(s)	1.46	1.51	1.68
LOF-MST	ACC	0.67	0.73	0.85
	NMI	0.66	0.72	0.86
	Time(s)	42.81	44.49	55.39
DLORE-DP	ACC	0.99	0.98	0.99
	NMI	0.98	0.95	0.99
	Time(s)	19.88	20.21	19.92

utilize the graph distance to evaluate the dissimilarity between local cores, DLORE-DP is effective for clustering manifold data sets and at the same time it avoids low density points participating in the calculation of graph distance, so it is more robust than DBSCAN and LOF-MST algorithms. Besides, because we only need to obtain the shared neighbors of local cores and compute the graph distance between local cores, instead of all pairs of points, it greatly reduces the running time. The experimental results demonstrate that DLORE-DP is significantly more effective, efficient and robust than the compared methods when clustering data sets containing manifold clusters and a lot of noises.

CRedit authorship contribution statement

Dongdong Cheng: Conceptualization, Methodology, Software, Writing - original draft. **Sulan Zhang:** Writing - review & editing, Supervision, Funding acquisition. **Jinlong Huang:** Validation, Formal analysis, Resources, Data curation, Visualization.

Acknowledgment

This work is supported by Project of Chongqing Natural Science Foundation, China under Grant cstc2019jcyj-msxmX0683, National Natural Science Foundation of China under Grant 61601060, 61702060, and Project of Chongqing Municipal Education Commission, China under Grant KJZH17104, KJZD-M20190140.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at <https://doi.org/10.1016/j.knosys.2019.105454>.

References

- [1] J. Macqueen, Some methods for classification and analysis of multivariate observations, in: Proc. of Berkeley Symposium on Mathematical Statistics and Probability, 1967, pp. 281–297.
- [2] L. Kaufman, P.J. Rousseeuw, *Finding Groups in Data. an Introduction to Cluster Analysis*, Wiley, 2009.
- [3] B.J. Frey, D. Delbert, Clustering by passing messages between data points, *Science* 315 (5814) (2007) 972–976.
- [4] X. Zhang, W. Wang, K. Norvag, M. Sebag, K-AP: generating specified K clusters by efficient affinity propagation, in: Data Mining (ICDM), 2010 IEEE 10th International Conference on, IEEE, 2010, pp. 1187–1192.
- [5] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in: Kdd, Vol. 96, 1996, pp. 226–231.
- [6] Y. Chen, S. Tang, L. Zhou, C. Wang, J. Du, T. Wang, S. Pei, Decentralized clustering by finding loose and distributed density cores, *Inform. Sci.* 433–434 (2018) 510–526.
- [7] Y. ao Geng, Q. Li, R. Zheng, F. Zhuang, R. He, N. Xiong, RECOME: A new density-based clustering algorithm using relative KNN kernel density, *Inform. Sci.* 436–437 (2018) 13–30, <http://dx.doi.org/10.1016/j.ins.2018.01.013>.
- [8] X. Wang, X.L. Wang, C. Chen, D.M. Wilkes, Enhancing minimum spanning tree-based clustering by removing density-based outliers, *Digit. Signal Process.* 23 (5) (2013) 1523–1538.
- [9] M.M. Breunig, H.P. Kriegel, R.T. Ng, J. Sander, LOF: Identifying density-based local outliers, in: Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, May 16–18, 2000, Dallas, Texas, USA, 2000.
- [10] A. Rodriguez, A. Laio, Clustering by fast search and find of density peaks, *Science* 344 (6191) (2014) 1492–1496.
- [11] R. Liu, H. Wang, X. Yu, Shared-nearest-neighbor-based clustering by fast search and find of density peaks, *Inform. Sci.* 450 (2018) 200–226, <http://dx.doi.org/10.1016/j.ins.2018.03.031>.
- [12] 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, *Inform. Sci.* 354 (2016) 19–40, <http://dx.doi.org/10.1016/j.ins.2016.03.011>.
- [13] 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, <http://dx.doi.org/10.1016/j.knosys.2016.02.001>.
- [14] J. Xu, G. Wang, W. Deng, DenPEHC: Density peak based efficient hierarchical clustering, *Inform. Sci.* 373 (2016) 200–218, <http://dx.doi.org/10.1016/j.ins.2016.08.086>.
- [15] G. Wang, Q. Song, Automatic clustering via outward statistical testing on density metrics, *IEEE Trans. Knowl. Data Eng.* 28 (8) (2016) 1971–1985, <http://dx.doi.org/10.1109/TKDE.2016.2535209>.
- [16] Z. Liang, P. Chen, Delta-density based clustering with a divide-and-conquer strategy: 3DC clustering, *Pattern Recognit. Lett.* 73 (2016) 52–59, <http://dx.doi.org/10.1016/j.patrec.2016.01.009>.
- [17] C. Jinyin, L. Xiang, Z. Haibing, B. Xintong, A novel cluster center fast determination clustering algorithm, *Appl. Soft Comput.* 57 (2017) 539–555, <http://dx.doi.org/10.1016/j.asoc.2017.04.031>.
- [18] J. Huang, Q. Zhu, L. Yang, D. Cheng, Q. Wu, QCC: a novel clustering algorithm based on Quasi-Cluster Centers, *Mach. Learn.* 106 (3) (2017) 337–357, <http://dx.doi.org/10.1007/s10994-016-5608-2>.
- [19] D. Cheng, Q. Zhu, J. Huang, L. Yang, Q. Wu, Natural neighbor-based clustering algorithm with local representatives, *Knowl.-Based Syst.* 123 (2017) 238–253, <http://dx.doi.org/10.1016/j.knosys.2017.02.027>.
- [20] Y. Shi, Z. Chen, Z. Qi, F. Meng, L. Cui, A novel clustering-based image segmentation via density peaks algorithm with mid-level feature, *Neural Comput. Appl.* 28 (1) (2017) 29–39, <http://dx.doi.org/10.1007/s00521-016-2300-1>.
- [21] D. Liu, Y. Su, X. Li, Z. Niu, A novel community detection method based on cluster density peaks, in: X. Huang, J. Jiang, D. Zhao, Y. Feng, Y. Hong (Eds.), *Natural Language Processing and Chinese Computing*, Springer International Publishing, Cham, 2018, pp. 515–525.
- [22] X. Bai, P. Yang, X. Shi, An overlapping community detection algorithm based on density peaks, *Neurocomputing* 226 (2017) 7–15, <http://dx.doi.org/10.1016/j.neucom.2016.11.019>.
- [23] M. Wang, W. Zuo, Y. Wang, An improved density peaks-based clustering method for social circle discovery in social networks, *Neurocomputing* 179 (2016) 219–227, <http://dx.doi.org/10.1016/j.neucom.2015.11.091>.
- [24] B. Wang, J. Zhang, Y. Liu, Y. Zou, Density peaks clustering based integrate framework for multi-document summarization, *CAAI Trans. Intell. Technol.* 2 (1) (2017) 26–30, <http://dx.doi.org/10.1016/j.trit.2016.12.005>.
- [25] S. Wang, D. Wang, C. Li, Y. Li, G. Ding, Clustering by fast search and find of density peaks with data field, *Chin. J. Electron.* 25 (03) (2016) 397–402.
- [26] Q. Zhu, J. Feng, J. Huang, Natural neighbor: A self-adaptive neighborhood method without parameter K, *Pattern Recognit. Lett.* 80 (2016) 30–36, <http://dx.doi.org/10.1016/j.patrec.2016.05.007>.
- [27] L. Yang, Q. Zhu, J. Huang, D. Cheng, Adaptive edited natural neighbor algorithm, *Neurocomputing* 230 (2017) 427–433, <http://dx.doi.org/10.1016/j.neucom.2016.12.040>.
- [28] J. Huang, Q. Zhu, L. Yang, F. Ji, A non-parameter outlier detection algorithm based on natural neighbor, *Knowl.-Based Syst.* 92 (C) (2015) 71–77.
- [29] E. Tu, L. Cao, J. Yang, N. Kasabov, A novel graph-based k-means for nonlinear manifold clustering and representative selection, *Neurocomputing* 143 (2014) 109–122, <http://dx.doi.org/10.1016/j.neucom.2014.05.067>.
- [30] J.B. Tenenbaum, V. de Silva, J.C. Langford, A global geometric framework for nonlinear dimensionality reduction, *Science* 290 (5500) (2000) 2319–2323.
- [31] P. Yang, Q. Zhu, B. Huang, Spectral clustering with density sensitive similarity function, *Knowl.-Based Syst.* 24 (5) (2011) 621–628, <http://dx.doi.org/10.1016/j.knosys.2011.01.009>.
- [32] W. Chen, Y. Song, H. Bai, C. Lin, E.Y. Chang, Parallel spectral clustering in distributed systems, *IEEE Trans. Pattern Anal. Mach. Intell.* 33 (3) (2011) 568–586, <http://dx.doi.org/10.1109/TPAMI.2010.88>.
- [33] A.K. Jain, M.H.C. Law, Data clustering: A user's dilemma, in: *Pattern Recognition and Machine Intelligence*, Proceedings, Vol. 3776, 2005, pp. 1–10.

- [34] A. Gionis, H. Mannila, P. Tsaparas, Clustering aggregation, *ACM Trans. Knowl. Discov. Data* 1 (1) (2007).
- [35] D. Cheng, Q. Zhu, J. Huang, Q. Wu, L. Yang, A novel cluster validity index based on local cores, *IEEE Trans. Neural Netw. Learn. Syst.* 30 (4) (2019) 985–999.
- [36] J. Ha, S. Seok, J.-S. Lee, Robust outlier detection using the instability factor, *Knowl.-Based Syst.* 63 (2014) 15–23, <http://dx.doi.org/10.1016/j.knosys.2014.03.001>.
- [37] G. Karypis, E.-H. Han, V. Kumar, Chameleon: hierarchical clustering using dynamic modeling, *Computer* 32 (8) (1999) 68–75, <http://dx.doi.org/10.1109/2.781637>.