Pattern Recognition 156 (2024) 110772

Contents lists available at ScienceDirect

Pattern Recognition

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

Flexible density peak clustering for real-world data

Jian Hou [a,*], Houshen Lin [a], Huaqiang Yuan [a], Marcello Pelillo [b,c]

[a] School of Computer Science and Technology, Dongguan University of Technology, Dongguan 523808, China
[b] DAIS, Ca' Foscari University, Venice 30172, Italy
[c] European Centre for Living Technology, Ca' Foscari University, Venice 30123, Italy

ARTICLE INFO

Keywords:
Clustering
Density peak
Real-world data
Number of clusters

ABSTRACT

In density based clustering, the density peak algorithm has attracted much attention due to its effectiveness and simplicity, and a vast amount of clustering approaches have been proposed based on this algorithm. Some of these works require manual selection of cluster centers with a decision graph, where human involvement leads to uncertainty in clustering results. In order to avoid human involvement, some other algorithms depend on user-specified number of clusters to determine cluster centers automatically. However, it is well known that accurate estimation of number of clusters is a long-standing difficulty in data clustering. In this paper we present a sequential density peak clustering algorithm to extract clusters one by one, thereby determining the number of clusters automatically and avoiding manual selection of cluster centers in the meanwhile. Starting from a density peak, our algorithm generates an initial cluster surrounding the density peak in the first step, and then obtains the final cluster by expanding the initial cluster based on the relative density relationship among neighboring data points. With a peeling-off strategy, we obtain all the clusters sequentially. Our algorithm works well with clusters of Gaussian distribution and is therefore potential for clustering of real-world data. Experiments with a large number of synthetic and real datasets and comparisons with existing algorithms demonstrate the effectiveness of the proposed algorithm.

1. Introduction

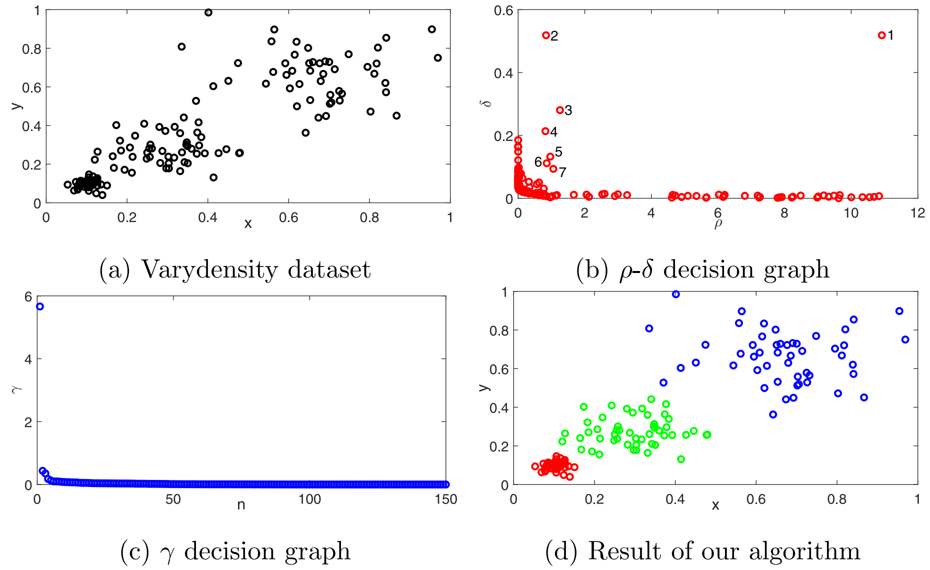
Some well-known clustering algorithms include k-means, DBSCAN [1], mean shift and normalized cut [2]. Existing algorithms are traditionally classified into partition-based [3], hierarchical clustering, distribution-based and density-based [4]. Recently, much attention is directed to subspace clustering [5], multi-view clustering [6,7], ensemble clustering [8] and their variants [9]. In addition, the dominant set (DSet) [10] and affinity propagation [11] algorithms also attract research interests due to their interesting properties.

Many density based clustering algorithms. e.g., DBSCAN, are able to generate clusters of arbitrary shapes. While DBSCAN relies on a density threshold to group data into clusters, the density peak clustering (DPC) algorithm [12] firstly identifies cluster centers from local density peaks, and then assigns other data into clusters surrounding cluster centers. Therefore identifying cluster centers is a key step in the DPC algorithm. Existing DPC-based algorithms typically solve this problem in two different ways. First, the original DPC and some algorithms [13,14] build a decision graph and identify the cluster centers by human observation. However, human involvement leads to uncertainties in the clustering results, as different users may select different (number of) cluster centers from the same decision graph. Second, some other works require to specify the number n_c of clusters, and then determine the clusters automatically by selecting the n_c data with largest $\rho\delta$ [15], with ρ denoting local density and δ representing distance to nearest data of larger density. While some methods [16,17] are proposed to estimate the number of clusters, our experiments show that it is still difficult to obtain an accurate estimation of this parameter. In addition, even if the number n_c is accurate, selecting the n_c data with largest $\rho\delta$ does not guarantee that each cluster is assigned a cluster center, especially with datasets where different clusters have significantly different densities.

In this paper we present a sequential density peak clustering algorithm, which avoids both manual selection of cluster centers and specifying the number of clusters simultaneously. Starting from the largest-density data point, we generate an initial cluster following DBSCAN, and then expand the initial cluster based on DPC to obtain the first cluster. In the remaining unclustered data, we obtain the next cluster with the same method, and repeat this process until all the data are included into clusters, thereby determining the number of clusters automatically. In extracting each cluster, we simply use the largest-density unclustered data as the cluster center, thereby avoiding

[*] Corresponding author.
E-mail address: houjian@dgut.edu.cn (J. Hou).

https://doi.org/10.1016/j.patcog.2024.110772
Received 31 July 2023; Received in revised form 7 April 2024; Accepted 8 July 2024
Available online 14 July 2024



(a) Varydensity dataset (b) ρ-δ decision graph (c) γ decision graph (d) Result of our algorithm

Fig. 1. Demonstration of DPC on the varydensity dataset.

the (manual) selection of cluster centers. We further present a non-parametric method to determine candidate data in cluster expansion for each cluster adaptively.

The contributions of this paper are as follows. First, we propose to do density peak clustering sequentially to determine the number of clusters and avoid manual selection of cluster centers. Second, in extracting each cluster, we use DBSCAN to generate an initial cluster surrounding the largest-density data point, eliminating the necessity to select cluster centers. Third, in expanding the initial cluster, we present a non-parametric method which avoids both under-expansion and over-expansion. Finally, we use a large number of synthetic and real datasets in experiments to validate the effectiveness of our algorithm.

Our algorithm is proposed on the basis of the DSet-DPC algorithm in [18] and aims to solve some major problems of the latter. First, the DSet-DPC algorithm uses the DSet algorithm [10] to generate initial clusters, resulting in a large computation load. In contrast, our algorithm uses DBSCAN to generate initial clusters surrounding local density peaks, and is shown to be much more efficient that DSet-DPC. Second, as the initial clusters from the DSet algorithm tend to be of spherical shapes and DBSCAN generates initial clusters of arbitrary shapes, our algorithm is more adapted to real datasets where clusters may be of irregular shapes. This advantage is validated by our experiments on 25 real datasets in Section 4. Third, the DSet-DPC algorithm involves a DSet-related parameter, which could only be determined by experiments. Whereas in our algorithm, the DBSCAN parameters can be inferred more reasonably. These improvements together enable our algorithm to generate better results than DSet-DPC with much less running time, showing a significant advantage over the latter.

2. Related works

In this part we firstly introduce the original density peak clustering algorithm, followed by a brief review of some recent works based on density peak.

2.1. Density peak clustering algorithm

The density peak clustering algorithm is composed of two major steps, i.e., cluster center identification and non-center data allocation, and each step is based on an assumption. Specifically, it is assumed that cluster centers are local density peaks, and each data is in the same cluster as its nearest neighbor with larger density. These two steps are introduced in more details below.

A local density peak refers to a data point surrounded by neighboring data with smaller densities. By assuming that cluster centers are local density peaks, we see that a cluster center has a larger density than its neighbors. This property has two consequences. First, a cluster center has a large local density ρ. Second, a cluster center is distant from any data with larger density. If we denote the distance between one data point and its nearest neighbor with larger density by δ, then a cluster center has a large δ. This means that a cluster center has a large ρ and a large δ. In contrast, it is easy to check that a non-center data has a small ρ and/or a small δ. DPC uses this difference to distinguish between cluster centers and non-center data. Specifically, a ρ-δ decision graph is built to demonstrate the distribution of all data points in the ρ-δ space (Fig. 1(b)). Evidently, in the ρ-δ decision graph some local density peaks are isolated from the majority of data points, and they will be selected as cluster centers. As identifying clusters based on the ρ-δ decision graph involves two thresholds of ρ and δ, the DPC algorithm further builds a γ decision graph with $\gamma = \rho\delta$ (Fig. 1(c)), where the data points are sorted in the decreasing order of γ values. In this case, only one threshold of γ is needed to identify cluster centers.

After the cluster centers are identified, the non-center data are grouped surrounding the cluster centers. In implementation, all the non-center data are sorted in the decreasing order of local density, and then each non-center data is assigned to be in the same cluster as its nearest neighbor with larger density.

2.2. Density peak based algorithms

The DPC clustering results are influenced mainly by three factors, i.e., density definition, cluster center identification and non-center data allocation. We review some recent DPC-based algorithms from these three directions.

The original DPC algorithm presents the cutoff and Gaussian kernels to calculate local density, involving a parameter d_c denoting the cutoff distance. To avoid the problems caused by d_c, a non-parametric kernel is presented based on heat diffusion [13], and another is based on shared-nearest-neighbors [15]. The k nearest neighbors (kNN) are used frequently in density estimation [14,19] to encode local data distribution. In addition, a nearest neighbor fuzzy kernel is defined in [20] by combining kNN and fuzzy neighborhood. The local density is defined in [21] as the number of points whose neighbors include this point, to treat cluster centers in dense and sparse regions equally.

In the original DPC algorithm, cluster centers are selected manually from a decision graph, and the human involvement leads to uncertainties in clustering results. In order to solve this problem, a statistical test method and a new density calculation method is proposed in [22] to identify cluster centers automatically. The algorithm of [23] detects a large number of cluster centers and small clusters, and then merges small clusters based on a heuristic method to obtain final clusters. Motivated by support vector machines, [24] proposes to generate initial small clusters, obtain feedbacks between every two initial clusters based on support vectors, and then merge initial clusters recursively. The constraint based DPC algorithm [25] also proposes to merge initial cluster to obtain final clusters. A similar work is presented in [26], which detects and merges core points in cluster center identification. In addition, a divide-and-conquer strategy is used in [27] to identify cluster centers without prior knowledge.

As the one-step data allocation strategy in the original DPC may cause error propagation, SNN-DPC defines inevitably subordinate and possibly subordinate and presents a two-step allocation strategy. A graph-based label propagation method is presented in [19] to assign labels to non-center data, to deal with data points in the border and overlapped regions. FKNN-DPC [28] uses a uniform local density metric based on kNN to identify cluster centers and then adopts two strategies for data allocation to avoid error propagation. ADPC-KNN [29] presents a new density kernel based on kNN and merges initial clusters which are density reachable with an aggregation strategy. DPC-FWSN [20] proposes an allocation strategy for weighted shared neighbor similarity to improve sample allocation in the boundary of sparse clusters.

3. Our algorithm

Noticing that manual selection of cluster centers leads to uncertainties in clustering results of the DPC algorithm, we resort to automatic identification of cluster centers in our algorithm. As aforementioned in the Introduction, DPC faces many challenges in identifying cluster centers, including estimating the number of clusters accurately, isolating cluster centers from non-center data reliably, and dealing with the influence from large density differences across clusters. In this paper we present a sequential DPC algorithm to deal with these challenges. Our algorithm extracts clusters one by one, until all the data are grouped into clusters. In extracting each cluster, we start from the largest-density data point and generate an initial cluster, and then expand the initial cluster based on DPC to obtain the final cluster. Our algorithm adopts a similar idea to that of DSet-DPC, and outperforms the latter in both clustering result and computation efficiency. In the following we present the details of initial cluster extraction and cluster expansion.

As a density based clustering approach, our algorithm relies on the local density of all the data to be clustered. Here we use the average distance to a number of nearest neighbors to estimate local density. Specifically, with the dataset S and a data point $p_i \in S$, the local density of p_i is calculated as

$$\rho_i = \frac{d_{max}}{\frac{1}{k}\sum_{p_j \in S_{inn}} d(p_i, p_j)}, \tag{1}$$

mật độ của điểm pi =
max(matrankhoangcach) chia cho
trung bình kc từ điểm pi đến k láng giềng gần nhất
với k là 1 tham số đầu vào (>=4 : chạy thử 5-20 : 11 tôt)

where d_{max} is the maximum of all pairwise distances of the data in S, and S_{inn} is composed of the k nearest neighbors of p_i. Following the original DPC algorithm, we replace the number k by the share ϵ of all the data, i.e.,

$$k = \lfloor \epsilon |S| \rfloor. \tag{2}$$

phép nhân với tổng số điểm, lấy phần nguyên.
vậy nói dễ hiểu thực chất thì k là tham số, phụ
thuộc vào tổng số điểm |S|

The share ϵ is the first parameter of our algorithm, and its influence on clustering results will be discussed in Section 4.

In DPC-based clustering algorithms, the relationship between one data and its nearest neighbor with larger density plays a key role. For ease of presentation, we introduce the concept of *superordinate* following [30].

Definition 1. Given a data point $p_i \in S$, if its nearest neighbor with larger density is $p_i^* \in S$, i.e.,

$$p_i^* = \mathop{\arg\min}_{p_j \in S, \rho_j > \rho_i} d(p_i, p_j), \tag{3}$$

điểm có khoảng cách nhỏ nhất đến pi
mà mật độ lớn hơn pi (gọi là điểm gần nhất có mật độ lớn hơn)

then p_i^* is the superordinate of p_i.

Based on the assumption of the DPC algorithm, one non-center data belongs to the same cluster as its superordinate.

3.1. Initial cluster extraction

DPC identifies all the cluster centers in the first step, and then gathers non-center data around cluster centers based on the assumption that one data is in the same cluster as its superordinate. In other words, large-density data are firstly included into clusters, followed by small-density ones. In our algorithm, initial clusters are firstly extracted, and they are expanded to generate the final clusters based on the DPC algorithm. This means that data in the initial cluster have larger density than other data in the same cluster. In other words, an initial cluster corresponds to the large-density area of a cluster.

Our algorithm to extract such an initial cluster is based on the following observation. In the DPC algorithm, a cluster center is a local density peak, and it has the largest density in the whole cluster. Meanwhile, the densities of adjacent data points usually vary smoothly, indicating that the neighboring data of a cluster center tend to have large densities too. Therefore a cluster center and its neighboring large-density data constitute a large-density area, and can be used as the initial cluster in our algorithm. As a result, the initial cluster extraction consists of two steps, i.e., identifying the cluster center, and detecting its neighboring large-density data.

3.1.1. Identifying the cluster center

In identifying the cluster center, the original DPC algorithm distinguishes cluster centers from non-center data manually based on their difference in ρ and δ values. This practice leads to some problems in dealing with datasets of complex distributions (see Appendix A for a discussion). Taking the Varydensity dataset in Fig. 1 for example, while the real number of clusters is 3, different users may select point 1, points 1 and 2, points 1 to 4, or points 1 to 7, as cluster centers from Fig. 1(b). The γ decision graph in Fig. 1(c) has similar problems. Whereas in our sequential clustering algorithm, each cluster is obtained separately, and each time we only need to identify one cluster center. In this way, we avoid the difficulty in determining the number of clusters, and are able to obtain the good result in Fig. 1(d).

Motivated by the observation that a cluster center has the largest density in its cluster, we propose to use only the local density ρ to determine each cluster center. In the dataset S, we use the largest-density unclustered data as the cluster center p_c, i.e.,

$$p_c = \mathop{\arg\max}_{p_i \in S_{uc}} \rho_i, \tag{4}$$

điểm có mật độ lớn nhất trong số các điểm chưa phân cụm

where ρ_i is the local density of data p_i, and $S_{uc} \subseteq S$ denotes the set of unclustered data in S. For the first cluster, the cluster center is the data point with the largest ρ in all the data. Whereas for the second cluster, the cluster center is the data point with the largest ρ in all the data excluding those in the first cluster. The cluster centers in the subsequent clusters are determined similarly.

The original DPC algorithm uses ρ and δ to identify cluster centers, whereas our algorithm uses only ρ and discards δ. We argue that this difference has no influence on the clustering result. In the original DPC algorithm, δ is used to suppress the non-density-peak data points which have large ρ and small δ, and highlight the local density peaks which have both large ρ and large δ. However, δ is not applicable to the data point p_{max} with the largest density in all the data. The reason is that δ is defined as the distance between one data and its superordinate, and p_{max} has no superordinate. In implementation, the δ of p_{max} is simply

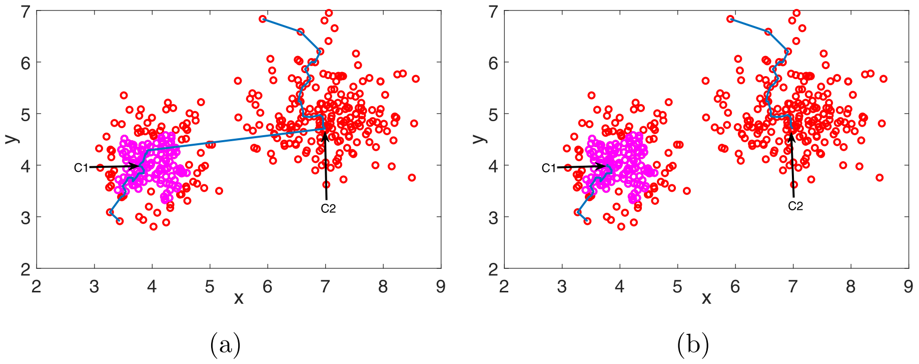


Fig. 2. Demonstration of DPC clustering. The magenta circles denote data in the initial cluster, and one data is connected to its superordinate by a blue line. C1 and C2 are the cluster centers of the left and right cluster, respectively. (a) Simple cluster expansion includes the right cluster center into the left cluster. (b) Detecting all cluster center simultaneously is able to avoid merging different clusters. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

set as the maximum in all the δ's. This means that in the DPC algorithm, the largest-density data is a unquestionable cluster center, and it does not need to compete with other density peaks. In our algorithm clusters are extracted sequentially, and we need only one density peak as the cluster center in extracting each cluster. Therefore we can simply use the largest-density unclustered data point as the cluster center.

3.1.2. Detecting neighboring large-density data

With the obtained cluster center, the next step is to detect its neighboring large-density data to constitute the initial cluster. In order to serve our purpose of expanding the initial cluster to the final cluster, we choose to generate the initial cluster based on DBSCAN and determine the involved parameters adaptively for each cluster. The reason of selecting DBSCAN here is presented in Appendix B. The DBSCAN algorithm involves two parameters, i.e., the neighborhood radius Eps and minimum number $MinPts$ of data in the neighborhood. The original DBSCAN algorithm recommends that $MinPts = 4$ is appropriate for 2-dimensional datasets. Meanwhile, it is shown in the density peak based algorithm DenMune [31] that reducing the data dimension to 2 with t-sne is able to generate better results than using the original data. Our experiments show that our algorithm also generates better results with data reduced to 2-dimension with t-sne. Motivated by these results, we use the t-sne method to transform high-dimensional data to be of 2-dimension, and then fix $MinPts = 4$ in our algorithm. In this way, we remove the parameter $MinPts$ and are left with only the parameter Eps.

We determine Eps adaptively for each cluster as follows. If we denote the distance between the cluster center and its $MinPts$th nearest neighbor by d_0, then there are $MinPts$ data points in the neighborhood of radius d_0 surrounding the cluster center. Here $MinPts = 4$ and $Eps = d_0$ is a measure of the local density of the cluster center. As the cluster center has the largest local density in its cluster, the neighboring data are with smaller density and have less than $MinPts$ data points in their d_0-radius neighborhood. In other words, if we use $MinPts = 4$ and $Eps = d_0$ to do DBSCAN clustering starting from the cluster center, the first obtained cluster consists of only the cluster center and its $MinPts$ nearest neighbors. In order to include more data into the cluster, we need to reduce the density threshold by reducing $MinPts$ and/or increasing Eps. As we have fixed $MinPts = 4$ in our algorithm, we choose to set Eps to be larger than d_0. Noticing that d_0 is the distance between the cluster center and its $MinPts$th nearest neighbor, we determine Eps as

$$Eps = d(p_c, p_{knn}), s.t. \quad \kappa > MinPts, \tag{5}$$

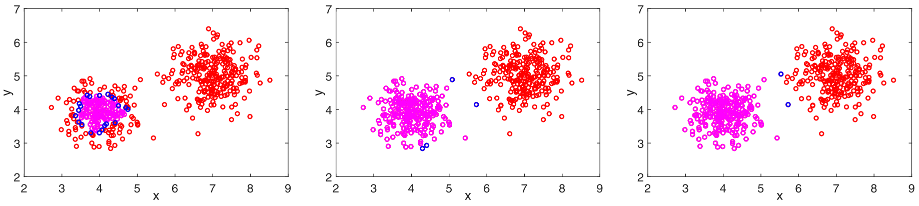
with p_{knn} denoting the κth nearest neighbor of the cluster center p_c. In this way Eps is calculated adaptively depending on the data distribution in the neighborhood of the cluster center. Here κ is the second parameter of our algorithm, and will be discussed in the experiments in Section 4.

With $MinPts = 4$ and the obtained Eps, we extract the initial cluster based on the DBSCAN algorithm. Starting from the cluster center, we group all the data in its Eps-radius neighborhood into the initial cluster. Then for each data in the initial cluster, we check if it is a core point, i.e., if there are at least $MinPts$ points in its Eps-radius neighborhood. All the data in the Eps-neighborhood of a core point will be included into the cluster. We repeat this process until all the data in the initial cluster have been visited, obtaining the final version of the initial cluster.

In our algorithm we detect the cluster center and generate an initial cluster around the cluster center, and then expand the initial cluster to obtain the final cluster. Here we choose to expand the initial cluster but not the cluster center directly, based on the following reason. DPC-based cluster expansion works following a descending order of data densities. This method is suitable for data of Gaussian distribution, but may not be able to deal with irregular distributions of data. In contrast, the DBSCAN algorithm is based on a density threshold, and the generated initial cluster can be of any complex distribution, only if densities of the inside data are large enough. Therefore generating the initial cluster based on DBSCAN is able to accommodate possible irregular distributions, and is more adapted to real-world data of complex distributions. Experiments in Section 4.4 demonstrate the advantage of expanding from the initial cluster over expanding from the cluster center directly.

3.2. Cluster expansion

In the DPC algorithm, after cluster centers are identified, the non-center data are grouped into clusters using the assumption that each data is in the same cluster as its superordinate. As shown in Appendix C, simply expanding the initial cluster based on this assumption may merge multiple clusters into one. In order to avoid including data of other clusters in cluster expansion, we choose to terminate the cluster expansion before large-density data in other clusters are visited. This means that we cannot sort the unclustered data in descending order of local density as in the original DPC algorithm. Instead, the unclustered data are sorted in ascending order according to their distances to the initial cluster, and those closest to the initial cluster are firstly visited. In this way, the small-density data of the same cluster have priority over the large-density data of other clusters in being visited. For example, in Fig. 2 the left initial cluster and the right cluster center C2 are separated by small-density data in both clusters. By sorting unclustered data in ascending order of distances to the initial cluster, the small-density data in the left cluster are firstly visited, followed by small-density data in the right cluster, before C2 is visited. This makes it possible to terminate the cluster expansion before C2 is visited, and therefore avoid merging the right cluster into the left one.



(a) The first iteration (b) A halfway iteration (c) The last iteration

Fig. 3. Cluster expansion of our algorithm. The magenta, blue and red circles denote data in the initial cluster, candidate data, and unclustered data, respectively. From (a) to (c), unclustered data in the left cluster are included into the initial cluster gradually. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

3.2.1. Cluster expansion termination criterion

The next problem is to find a criterion to terminate the cluster expansion at an appropriate point. In cluster expansion, we intend to avoid including data of other clusters and in the meanwhile include all the data in the same cluster. This means that we need to terminate the cluster expansion at the boundary between different clusters, and our method is described as follows. Denoting the initial cluster by S_{ic} and the set of unclustered data by S_{uc}, we find a candidate set S_{candi} in S_{uc}, i.e., $S_{candi} \subset S_{uc}$. The candidate set S_{candi} is composed of the unclustered data which are closest to the initial cluster, and these data are called the *candidate data* for inclusion into the initial cluster. For each candidate data in S_{candi}, if its superordinate is in S_{ic}, we include this data into S_{ic} and exclude it from S_{uc}. After all the candidate data in S_{candi} are visited, if at least one candidate data is included into S_{ic}, we continue to find a new S_{candi} and add qualified candidate data in S_{candi} into S_{ic}. Repeating this process until none of the candidate data in S_{candi} can be included into S_{ic}, we terminate the cluster expansion and treat the latest S_{ic} as the final cluster.

Taking the dataset in Fig. 3 for example, we justify this cluster expansion termination criterion as follows. In Fig. 3(a) all the candidate data lie in the left cluster S_l in the first iteration. As two clusters are separated by a density valley, the superordinates of these candidate data are very likely to be in S_l. Meanwhile, we notice that data in the initial cluster S_{ic} have larger densities than neighboring data, including these candidate data. In this case, it is probable that some or all of these candidate data have their superordinates in S_{ic}. In Fig. 3(b), some of the candidate data lie in S_l. Similar as in Fig. 3(a), this part of candidate data are likely to have their superoridinates in S_{ic}. As in Fig. 3(c) all the candidate data are in the right cluster S_r, their superordinates are very likely to be in S_r too, due to the density valley between two clusters. In other words, none of the candidate data has its superordinate in S_{ic}. Fig. 3(a) to (c) demonstrate three kinds of cluster expansion states, associated with different superordinate distributions of candidate data. Evidently the cluster expansion should be continued at the states of Fig. 3(a) to (b), and should be terminated at the state of Fig. 3(c). Therefore in our algorithm, once we find that none of the candidate data in S_{candi} can be included into the initial cluster, the cluster expansion is terminated. With this criterion, all the data in the left cluster are included into S_{ic} and none of the data in the right cluster is included. In other words, this method avoids both under-expansion and over-expansion simultaneously in expanding the initial cluster, serving our purpose perfectly.

3.2.2. Determining the candidate set

With our cluster expansion termination criterion, if none of the data in the candidate set S_{candi} can be included into the initial cluster S_{ic}, the cluster expansion is terminated. One problem left open is to determine the candidate set S_{candi}. As shown in Appendix D, the candidate set S_{candi} should consist of the unclustered data closest to the initial cluster S_{ic}. Therefore one can simply calculate the distances between S_{ic} and the data outside S_{ic}, and then select those with the minimum distances. However, one major problem with this method is that it is difficult to determine the appropriate size of S_{candi}. In order to solve this problem, we present the following method to determine S_{candi} for each cluster adaptively.

Firstly, we introduce the concept of *direct neighbor* defined as follows.

Definition 2. Direct neighbor. Given two disjoint sets S_1 and S_2 with $S_1 \cap S_2 = \emptyset$. For a data point $p_i \in S_1$, if its nearest neighbor in S_2 is p_i^{dn}, i.e.,

$$p_i^{dn} = \arg\min_{p_j \in S_2} d(p_i, p_j), \text{điểm trong S2 mà gần nhất so với pi} \tag{6}$$

then p_i^{dn} is a direct neighbor of the set S_1.

Based on this definition, for each data in S_{ic}, we find its nearest neighbor in S_{uc}, obtaining a direct neighbor of S_{ic}. The set of all these direct neighbors are used as the candidate set S_{candi} in our algorithm. Intuitively, these direct neighbors in S_{candi} form a thin layer surrounding s_{ic}, as illustrated in Fig. 3. As the data in S_{ic} have larger densities than neighboring unclustered data, and direct neighbors are the closest unclustered data to S_{ic}, it is very probable that direct neighbors have their superordinates in S_{ic}. Therefore direct neighbors are appropriate candidate data in our DPC-based cluster expansion method, and they are most probable to be included into S_{ic}.

Although each data in S_{ic} corresponds to a direct neighbor, the candidate set S_{candi} is usually much smaller than S_{ic}. The reason is that multiple data in S_{ic} often share the same nearest neighbor in S_{uc}. The size of S_{candi} is influenced by many factors, including the shape of S_{ic}, the position of S_{ic} in the whole cluster, and data distribution in the whole cluster, etc. A direct neighbor of S_{ic} is the nearest unclustered neighbor of at least one data in S_{ic}. As each direct neighbor has the support of at least one data in S_{ic}, all direct neighbors are qualified candidates in cluster expansion. By selecting all direct neighbors as candidate data, we determine the candidate set adaptively.

Our cluster expansion method determines the candidate set adaptively for each cluster, and the obtained candidate data (direct neighbors) form a thin layer surrounding the initial cluster. The thin layer means that all the candidate data touch the initial cluster directly, which is beneficial to their inclusion into the initial cluster. In contrast, if we select as candidate data a number of unclustered data with minimum distances to the initial cluster, the inappropriate number n_p of candidate data may degrade clustering results from different aspects. With a small n_p, it is possible that no candidate data is included into the initial cluster, and cluster expansion is terminated prematurely. Whereas with a large n_p, candidate data may form a thick layer and include cluster centers of other clusters, thereby merging different clusters into one. Experiments in Section 4.5 demonstrate the advantage of our method.

3.3. Algorithm

In our sequential density peak clustering algorithm, clusters are extracted one by one with a peeling-off strategy. After the first cluster

Algorithm 1 Our algorithm of extracting a single cluster.

```
Input:   $S$, $S_{uc}$, $\rho$
Output:  $S_c$                          //a cluster extracted from $S_{uc}$
1: $p_c \leftarrow \arg\max_{p_i \in S_{uc}} \rho_i$. B3. tìm tâm cụm //find the cluster center with Eq. (4)
2: Calculate $Eps$ with Eq (5) and $MinPts = 4$. tính khoảng cách từ điểm pc đếm điểm láng giềng thứ 11
Sic <-- 3: Starting from $p_c$, obtain an initial cluster $S_{ic}$ based on DBSCAN. B4. Tạo cụm ban đầu bằng DBSCAN
4: $S_{uc} \leftarrow S_{uc} \setminus S_{ic}$, $count \leftarrow 1$.          //remove $S_{ic}$ from $S_{uc}$
5: while $count > 0$ do        //if at lease one candidate is included
   into $S_{ic}$ cứ có điểm mới thêm vào Sic ở Bước 6 là tính lại Scandi
6:     $S_{candi} \leftarrow \emptyset$.
7:     for $p_i \in S_{ic}$ do b5. tìm tập ứng viên Scandi, tìm trên all các Sic
8:         $p_i^{dn} \leftarrow \arg\min_{p_j \in S_{uc}} d(p_i, p_j)$, $S_{candi} \leftarrow S_{candi} \bigcup \{p_i^{dn}\}$.
9:     end for
10:    $count \leftarrow 0$.
11:    for $p_i \in S_{candi}$ do B6. Tìm điểm cấp trên cho all các Scandi (điểm gần nhất có mật độ lớn hơn
12:        $p_i^* = \arg\min_{p_j \in S, \rho_j > \rho_i} d(p_i, p_j)$.    //find the superordinate using
   Eq. (3)
13:        if $p_i^* \in S_{ic}$ then điểm cấp trên của điểm trong Scandi mà thuộc Sic  thì thêm vào kết quả cụm là Sic
14:            $S_{ic} \leftarrow S_{ic} \bigcup \{p_i\}$, $S_{uc} \leftarrow S_{uc} \setminus \{p_i\}$, $count \leftarrow count + 1$. 1 điểm được thêm Sic
15:        end if
16:    end for
17: end while
18: $S_c \leftarrow S_{ic}$.
```

CẢI TIẾN: ĐIỂM NÀO ĐÃ PHÂN CỤM THÌ GÁN MẬT ĐỘ LÀ -1 (SỬ DỤNG THÊM 1 MẢNG ĐỂ ĐÁNH DẤU (thay vì thực hiện phép trừ tập hợp. CÁCH LƯU TRỮ NÀY CŨNG GIÚP VIỆC KIỂM TRA 1 ĐIỂM KHÔNG THUỘC Si NHANH HƠN VÌ TRUY CẬP TRỰC TIẾP PHẦN TỬ TƯƠNG ỨNG CỦA MẢNG THAY VÌ SỬ DỤNG PHÉP TOÁN TẬP HỢP ???

is extracted, we continue to extract the second one in the remaining unclustered data. This process is repeated until all the data are grouped into clusters, and the number of clusters is determined automatically.

Each cluster is obtained in the following way. In the remaining unclustered data, the one with the largest local density is selected as the cluster center. This simple method avoids the difficulty of cluster center identification in the DPC algorithm. Starting from the cluster center, we obtain an initial cluster based one DBSCAN with $MinPts = 4$ and Eps calculated with Eq. (5). The initial cluster corresponds to a large-density area surrounding the cluster center, facilitating the subsequent cluster expansion based on DPC. For each data in the initial cluster, we find a direct neighbor of the initial cluster with Eq. (6), and all the direct neighbors constitute the candidate set S_{candi}. The candidate data in S_{candi} form a thin layer surrounding the initial cluster, beneficial to terminate the cluster expansion at the appropriate point. For each candidate data p_i in S_{candi}, we find its superordinate p_i^{dn}, and include p_i into the initial cluster if p_i^{dn} is in the initial cluster. If at least one candidate data can be included into the initial cluster, we continue to find the direct neighbors of the updated initial cluster and repeat the cluster expansion process. When no candidate data in the candidate set can be included into the initial cluster, the cluster expansion is terminated, and the latest initial cluster is used as the final cluster.

The detailed process of extracting each cluster is described in Algorithm 1, and the source code of the whole algorithm is available online.[1]

4. Experiments

In this part we conduct experiments to test the proposed algorithm in a comprehensive way. First, we study the influence of two parameters, namely ϵ in Eq. (2) and κ in Eq. (5), on the clustering results and then determine the appropriate values of these two parameters. Second, we test two key parts of our algorithms separately to demonstrate their advantages, i.e., expanding from the initial cluster instead of the cluster center, and the non-parametric cluster expansion method. Third, we compare our algorithm with the DSet-DPC algorithm to show the effectiveness of our improvements. Finally, we make a comparison between our algorithm and some other algorithms, including DPC-based ones and non-DPC ones. All the experiments are conducted on a computer with Intel Core i7-10510U processor (1.8 GHz) and 16 GB RAM, and the algorithm is implemented with Matlab.

4.1. Datasets

Similar to the DSet-DPC algorithm, our algorithm is designed for real-world datasets, and hence we adopt 25 real datasets in experiments. As in many cases real-world data of the same type follow the Gaussian distribution approximately [32], we also adopt 25 synthetic datasets where clusters are of Gaussian distribution. The characteristics of these synthetic and real datasets are summarized in Table 1, where NP, ND and NC denote the number of data points, data dimension and number of clusters, respectively. In the 25 synthetic datasets, Spread1, Spread2, Spread3, Spread4 and Spread5 are generated following [16] and the remaining datasets are taken from the clustering basic benchmark[2] and the ELKI project.[3] In these datasets, the clusters of Gaussian distributions are generated with different standard deviations. The 25 real datasets are taken from the UCI machine learning repository. It can be observed from Table 1 that these datasets cover a large variance in the dataset size, data dimension and the number of clusters.

4.2. Influence of the parameter ϵ

In calculating the local density with Eq. (1), we introduce the first parameter ϵ in Eq. (2), which is used to determine the number $\lfloor \epsilon \| S \| \rfloor$ of nearest neighbors in local density calculation. In the range [0.01,0.04], we test this parameter with the step of 0.001 and report the clustering results in Fig. 4. Here the clustering results are evaluated with NMI (normalized mutual information).

In Fig. 4(a), the clustering results on the majority of synthetic datasets have little variance with respect to ϵ. In the remaining datasets, the increase of ϵ results in performance improvement on G2-2-10, G2-128-30 and G2-1024-50, and performance degradation on A3 and Spread5. Overall, ϵ in the range [0.026,0.028] generates the best or near-best results for all the synthetic datasets.

In Fig. 4(b), the clustering results on the majority of real datasets also have little variance with respect to ϵ. In the remaining datasets, the increase of ϵ results in significant performance improvement on Thyroid, Ecoli, Libras and Olivertti. The clustering results on Waveform and Ionosphere experience both performance improvement and degradation. Similar as in the case of synthetic datasets, ϵ in the range [0.026,0.028] generates the best or near-best results for all the real datasets. Based on these results with synthetic and real datasets, we select $\epsilon = 0.028$ in our algorithm.

4.3. Influence of the parameter κ

In order to determine Eps in generating the initial cluster based on DBSCAN, we introduce the second parameter κ in Eq. (5), with the constraint that $\kappa > 4$. In our algorithm, $\kappa = 4$ corresponds to the local density of the cluster center, and $\kappa > 4$ indicates a smaller density than that of the cluster center. As the initial cluster is a large-density area surrounding the cluster center, the local density of data in the initial cluster will be smaller, but not much too smaller, than that of the cluster center. Therefore κ will not be much larger than 4. In the range [5,20], we test this parameter with the step 1 and report the clustering results in Fig. 5.

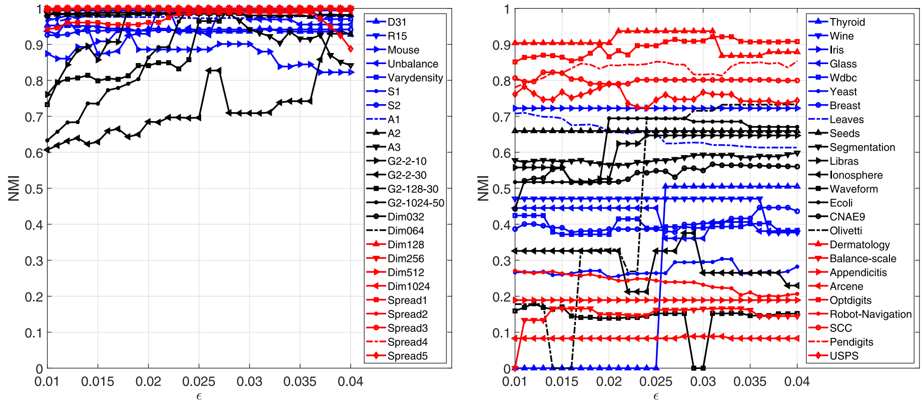
[1] https://github.com/dr-houjian/DBSCAN-DPC.

[2] http://cs.joensuu.fi/sipu/datasets/.

[3] https://elki-project.github.io/datasets/.

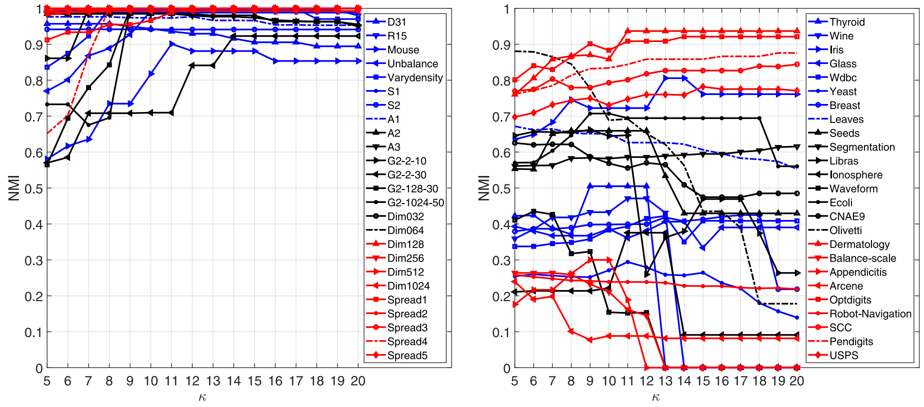
Table 1
Characteristics of datasets.

Dataset	NP	ND	NC	Dataset	NP	ND	NC	Dataset	NP	ND	NC	Dataset	NP	ND	NC
D31	3100	2	31	G2-1024-50	2048	2	2	Thyroid	215	5	3	Ecoli	336	7	8
R15	600	2	15	Dim032	1024	32	16	Wine	178	13	3	CNAE9	1080	856	9
Mouse	500	2	3	Dim064	1024	64	16	Iris	150	4	3	Olivertti	400	92 × 112	40
Unbalance	6500	2	8	Dim128	1024	128	16	Glass	214	9	6	Dermatology	366	33	6
Varydensity	150	2	3	Dim256	1024	256	16	Wdbc	569	30	2	Balance-scale	625	4	3
S1	5000	2	15	Dim512	1024	512	16	Yeast	1484	8	10	Appendicitis	106	7	2
S2	5000	2	15	Dim1024	1024	1024	16	Breast	699	9	2	Arcene	200	10 000	2
A1	3000	2	20	Spread1	1000	2	10	Leaves	1600	64	100	Optdigits	5620	64	10
A2	5250	2	35	Spread2	2000	10	20	Seeds	210	7	3	Robot-Navigation	5456	24	4
A3	7500	2	50	Spread3	3500	20	35	Segmentation	2310	19	7	SCC	600	60	6
G2-2-10	2048	2	2	Spread4	200	35	2	Libras	360	90	15	Pendigits	10 992	16	10
G2-2-30	2048	2	2	Spread5	5000	50	50	Ionosphere	351	34	2	USPS	11 000	256	10
G2-128-30	2048	2	2					Waveform	5000	21	3				



(a) With synthetic datasets (b) With real datasets

Fig. 4. Clustering results of our algorithm with different ϵ's.



(a) With synthetic datasets (b) With real datasets

Fig. 5. Clustering results of our algorithm with different κ's.

Our observations from Fig. 5(a) are as follows. With the majority of synthetic datasets, the increase of κ firstly improves the clustering results, and then larger κ's keep unchanged or degrade the cluster results slightly. While the performance peaks of different datasets are obtained at different κ's, it is shown that the best overall result is obtained at around $\kappa = 11$. These observations indicate that with clusters of Gaussian distribution, our algorithm is able to generate the near-best results with a relatively fixed κ.

As the clusters in many real datasets only follow the Gaussian distribution approximately, it can be observed from Fig. 5(b) that the variances of clustering results with respect to κ on real datasets are much more complex than on synthetic datasets. Some real datasets, e.g., Optdigits, Dermatology, Pendigits, SCC, USPS and Iris, have similar variances as with synthetic datasets. Meanwhile, we observe that the clustering results on these datasets are the best ones in all the real datasets. These two observations together indicate that clusters in these datasets do follow the Gaussian distribution approximately. In contrast, some other real datasets, e.g., Olivertti, Seeds, Thyroid, Ionosphere, Wine and Libras, experience significant performance degradation with $\kappa > 11$. One possible expansion is that clusters in these datasets deviate from the Gaussian distribution evidently. Nonetheless, $\kappa = 11$ is still among the best κ's for these datasets. Based on the clustering result variances with respect to κ on both synthetic and real datasets, we

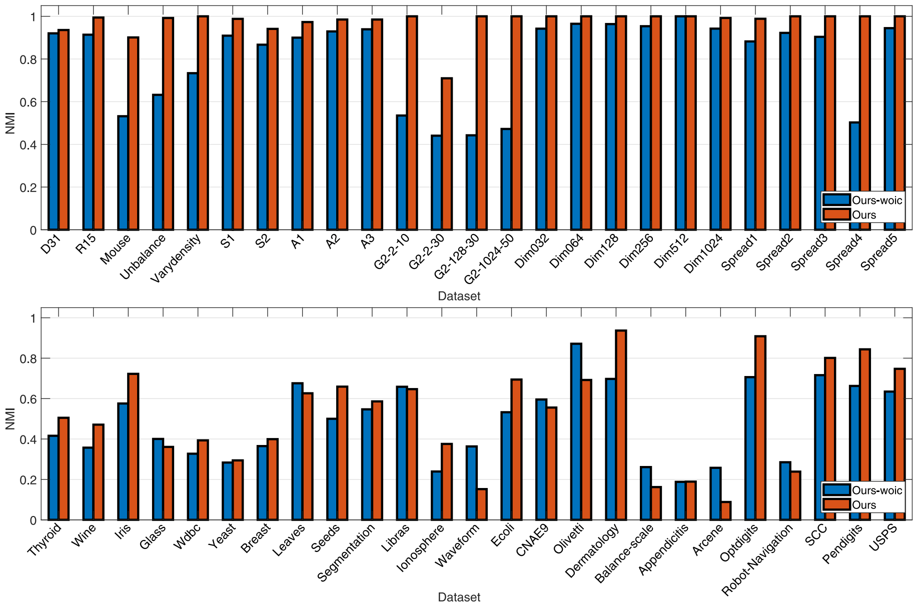


Fig. 6. Comparison between our algorithm and the version without initial cluster extraction.

select $\kappa = 11$ in our algorithm, consistent with our previous argument that the best κ is not much larger than 4.

4.4. Effect of initial cluster extraction

In extracting each cluster, we firstly detect the cluster center, followed by extracting the initial cluster and expanding the initial cluster. As the cluster center itself can be regarded as a special initial cluster (with only one data point), we can also expand from the cluster center directly, and remove the initial cluster extraction step. Here we compare these two versions of our algorithm in Fig. 6, where “Ours” denotes our algorithm with initial cluster extraction, and “Ours-woic” denotes our algorithm without initial cluster extraction.

In Fig. 6(a), our algorithm outperforms the version without initial cluster extraction on 24 out of 25 synthetic datasets, and performs the same as the latter on the remaining 1 dataset. This means that expanding from the initial cluster generates better results than expanding from the cluster center directly on clusters of Gaussian distribution. In Fig. 6(b), our algorithm performs better than the version without initial cluster extraction on 16 out of 25 real datasets, supporting the basis of our algorithm that clusters in many real datasets follow the Gaussian distribution approximately. In the remaining 9 real datasets, Olivertti and Leaves have very small clusters (10 data points in each cluster for Olivertti and 16 for Leaves). In this case, the obtained initial clusters based on DBSCAN are likely to be larger than the real clusters, and cluster expansion degrades the clustering results further. Instead, expanding from the cluster center is able to avoid this problem, and generate better results on these two datasets. As for the remaining 7 real datasets, we attribute the bad performance of our algorithm to the non-Gaussian distribution of clusters and cluster overlap. In conclusion, we choose to extract an initial cluster before cluster expansion, which is a better solution for clusters of Gaussian distribution in many synthetic and real datasets.

4.5. Effect of candidate set determination

In expanding the initial cluster, we present an adaptive method to determine the candidate data for inclusion into the initial cluster. Meanwhile, it is also possible to use a fixed number n_p of nearest unclustered data to the initial cluster as candidate data. Here we compare these two methods of obtaining the candidate set. We test n_p from 5 to 50 in the step of 5, and find that $n_p = 10$ generates the best average result. With $n_p = 10$, the comparison of these two methods is reported in Fig. 7, where “Fixed” denote the method of using a fixed number of nearest unclustered data.

In Fig. 7 our adaptive method performs better than the fixed method on 25 datasets, the same as the latter on 20 datasets, and inferior to the latter slightly on only 5 datasets (Dim1024, Spread1, Breast, Segmentation and Pendigits). This comparison demonstrates the advantage of our adaptive candidate set determination method.

4.6. Comparison with DSet-DPC

As our algorithm is proposed based on DSet-DPC [18], we make a comparison between these two algorithms. The comparisons of clustering results and running time are reported in Figs. 8 and 9, respectively.

In Fig. 8, our algorithm performs better than DSet-DPC on 24 datasets, the same as the latter on 12 datasets, inferior to the latter on 14 datasets. This comparison shows that our algorithm is better than DSet-DPC in overall clustering results. In fact, the average NMI of all datasets from our algorithm is 0.75, in contrast to 0.72 of DSet-DPC. In Fig. 9, the running time of our algorithm is less than that of DSet-DPC on all the 50 datasets, indicating that our algorithm is computationally more efficient than DSet-DPC. In our experiments, the average running time of our algorithm is 1.88 s, in contrast to the 24.26 s of DSet-DPC. In other words, while our algorithm improves the clustering results compared with DSet-DPC, it does not take more running time. Instead, it reduces the computational load significantly in comparison with DSet-DPC. In summary, our algorithm performs better than DSet-DPC in both clustering quality and running time, showing as a better clustering approach.

4.7. Comparison with other algorithms

Finally, we compare our algorithm with existing algorithms, including k-means, X-means [33], CBKM [34], CE3-kmeans [35], EM (Expectation Maximization), SPRG [36], and DPC-based algorithms, including FKNN-DPC [37], DPC-KNN [14], SNN-DPC [15], 3W-DPET [38], DenMune [31], DPC-FSC [39] and MDPC+[40]. With the parameter settings listed in Table 2, the comparison results are reported in Tables 3 and 4. In Table 2 the fixed parameters are selected to generate the best average result on all 50 datasets in our experiments.

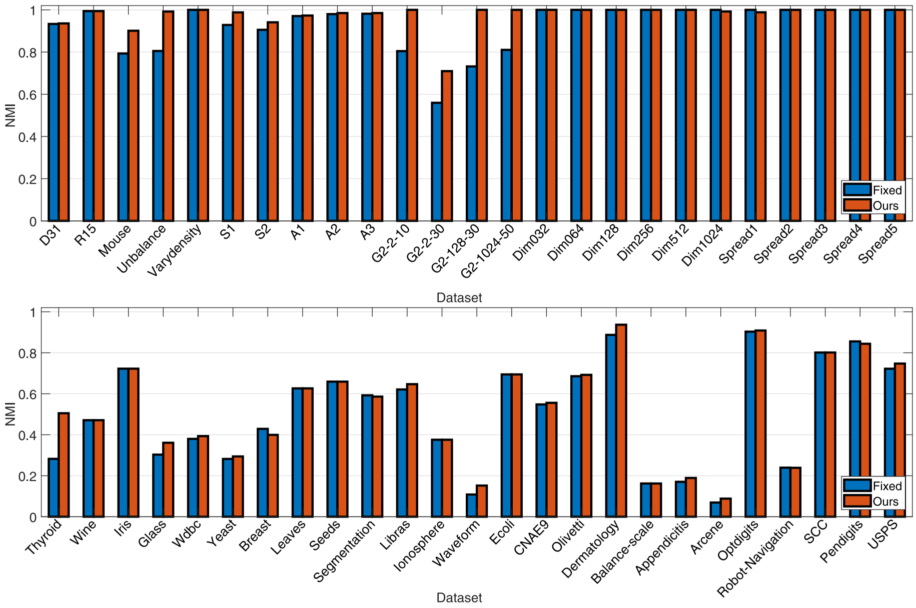


Fig. 7. Comparison between our algorithm with adaptive candidate data and the version with a fixed number of candidate data.

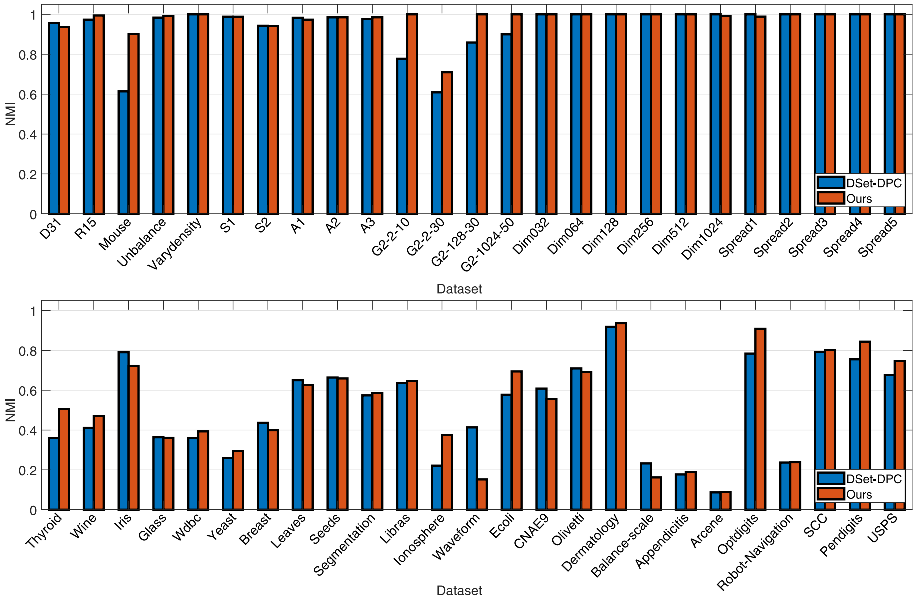


Fig. 8. Comparison between our algorithm and DSet-DPC.

Table 2
Parameter settings of baseline algorithms.

Algorithm	Parameter	Value
k-means	k: number of clusters	k = ground truth
CBKM	k: number of clusters	k = ground truth
EM	k: number of clusters	k = ground truth
SPRG-gt	k: number of clusters	k = ground truth
SPRG-egap	k: number of clusters	k determined with eigen-gap [17]
SPRG-LL	k: number of clusters	k determined with LastLeap [16]
CE3-kmeans	q: number of nearest neighbors, ρ: distance ratio	$q = 15$, $\rho = 1.5$ following [35]
FKNN-DPC	k: number of nearest neighbors	k = 5 selected from 2 to 50
DPC-KNN	k: number of nearest neighbors	k = 2 selected from 2 to 50
SNN-DPC	k: number of nearest neighbors	k = 8 selected from 2 to 50
3W-DPET	k: number of nearest neighbors	k = 6 selected from 2 to 50
DenMune	K: number of mutual nearest neighbors	K = 37 selected from 1 to 200
DPC-FSC	a: scale parameter of local density kernel	a = 0.9 selected from 0.01 to 0.99
MDPC+	λ: attention coefficient	λ = 1.2 selected from 1.0 to 5.0

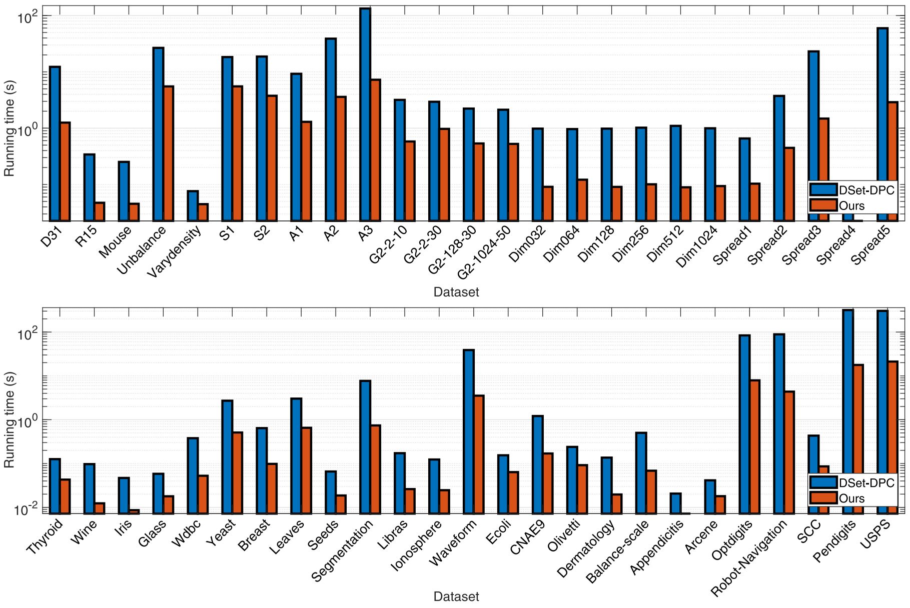


Fig. 9. Running time comparison between our algorithm and DSet-DPC.

Table 3
Clustering results (NMI) on 25 synthetic datasets.

	k-means	X-means	CBKM	CE3-kmeans	EM	SPRG	SPRG	SPRG	FKNN	DPC	SNN	3W-DPET	DenMune	DPC	MDPC+	Ours
						-gt	-egap	-LL	-DPC	-KNN	-DPC			-FSC		
D31	0.92	0.00	0.94	0.94	0.93	0.89	0.65	0.89	0.94	0.95	0.92	0.95	0.95	0.83	0.89	0.94
R15	0.94	0.00	0.99	1.00	0.95	0.98	0.88	0.98	0.97	0.99	0.99	0.99	0.77	0.98	0.97	0.99
Mouse	0.62	0.00	0.62	0.62	0.96	0.87	0.87	0.53	0.75	0.94	0.59	0.67	0.93	0.78	0.71	0.90
Unbalance	0.80	0.96	0.80	0.92	0.80	0.86	0.85	0.85	1.00	1.00	1.00	0.99	0.98	0.62	0.77	0.99
Varydensity	0.81	0.74	0.69	1.00	0.93	0.97	0.69	0.68	1.00	1.00	0.77	0.82	1.00	0.88	0.76	1.00
S1	0.92	0.00	0.95	0.97	0.93	0.93	0.53	0.93	0.98	0.99	0.98	0.99	0.97	0.08	0.92	0.99
S2	0.90	0.00	0.92	0.95	0.91	0.89	0.58	0.89	0.89	0.94	0.81	0.95	0.89	0.07	0.83	0.94
A1	0.92	0.98	0.95	0.96	0.93	0.90	0.70	0.90	0.91	0.95	0.94	0.94	0.97	0.20	0.92	0.97
A2	0.94	0.99	0.97	0.93	0.95	0.92	0.71	0.92	0.95	0.98	0.96	0.97	0.98	0.26	0.94	0.98
A3	0.95	0.00	0.97	0.97	0.96	0.94	0.75	0.94	0.95	0.98	0.95	0.98	0.99	0.31	0.96	0.98
G2-2-10	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.98	1.00	0.99	1.00
G2-2-30	0.92	0.92	0.92	0.98	0.92	0.93	0.93	0.93	0.88	0.92	0.88	0.93	0.92	0.94	0.51	0.71
G2-128-30	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
G2-1024-50	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.10	1.00	1.00	1.00	1.00	0.82	1.00
Dim032	0.93	0.96	1.00	0.98	0.93	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.98	1.00
Dim064	0.93	0.97	1.00	1.00	0.93	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.98	0.98	1.00
Dim128	0.92	0.97	1.00	1.00	0.92	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.99	0.85	0.79	1.00
Dim256	0.91	0.97	1.00	1.00	0.91	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Dim512	0.90	0.97	1.00	1.00	0.90	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Dim1024	0.91	0.97	1.00	1.00	0.91	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.99
Spread-2-10	0.88	0.00	1.00	0.97	0.89	0.99	0.67	0.99	1.00	1.00	1.00	0.97	1.00	1.00	1.00	0.99
Spread-10-20	0.91	1.00	1.00	0.99	0.91	1.00	0.58	1.00	1.00	1.00	1.00	0.99	1.00	1.00	1.00	1.00
Spread-20-35	0.92	1.00	1.00	0.99	0.92	1.00	1.00	1.00	1.00	1.00	1.00	0.99	1.00	1.00	1.00	1.00
Spread-35-2	1.00	1.00	1.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.00	1.00	1.00	1.00	1.00
Spread-50-50	0.93	0.98	1.00	1.00	0.93	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Mean	0.91	0.69	0.95	0.93	0.93	0.96	0.86	0.94	0.97	0.95	0.95	0.92	0.97	0.79	0.91	0.98

In Table 3, our algorithm generates the best average result on 25 synthetic datasets in all the 16 algorithms. In Table 4, our average result on real datasets is inferior only to the SPRG algorithm with the ground truth number of clusters. The SPRG algorithm learns the similarity between data and therefore performs well with real datasets, when the number of clusters is accurate. However, if we determine the number of clusters with the eigen-gap method or the LastLeap method, the SPRG results are degraded evidently and outperformed by our algorithm. We also compare the average running time on all the datasets of different algorithms, and find that our algorithm takes more running time than only k-means and MDPC+. Considering that the k-means algorithm relies on the number of clusters and the MDPC+ algorithm is outperformed by our algorithm significantly, we believe our algorithm with fixed parameters is shown to be effective in comparison with some other algorithms.

Considering that some algorithms use different parameters for different datasets to obtain the best possible result, we also conduct experiments to compare the best results of algorithms. It is shown that our algorithm is also competitive in comparison with other algorithms in this aspect. Due to limited space, the detailed comparison is presented in the supplemental material.[4]

5. Conclusion

In this paper we presented a sequential density peak clustering algorithm for real-world data. All clusters are obtained one by one in

[4] https://github.com/dr-houjian/DBSCAN-DPC.

Table 4
Clustering results (NMI) on 25 real datasets.

	k-means	X-means	CBKM	CE3-kmeans	EM	SPRG -gt	SPRG -egap	SPRG -LL	FKNN -DPC	DPC -KNN	SNN -DPC	3W-DPET	DenMune	DPC -FSC	MDPC+	Ours
Thyroid	0.60	0.42	0.28	0.67	0.79	0.77	0.52	0.49	0.46	0.37	0.51	0.45	0.00	0.50	0.17	0.51
Wine	0.76	0.35	0.43	0.94	0.82	0.82	0.82	0.48	0.76	0.54	0.84	0.81	0.47	0.89	0.60	0.47
Iris	0.69	0.64	0.75	0.67	0.84	0.72	0.72	0.63	0.74	0.71	0.80	0.67	0.76	0.78	0.72	0.72
Glass	0.34	0.47	0.42	0.39	0.32	0.41	0.40	0.36	0.37	0.27	0.26	0.25	0.32	0.33	0.31	0.36
Wdbc	0.62	0.33	0.47	0.60	0.67	0.57	0.57	0.41	0.58	0.43	0.61	0.63	0.42	0.62	0.53	0.39
Yeast	0.27	0.27	0.25	0.31	0.14	0.24	0.17	0.10	0.10	0.23	0.13	0.25	0.22	0.23	0.03	0.29
Breast	0.74	0.42	0.74	0.83	0.55	0.77	0.77	0.63	0.11	0.68	0.00	0.32	0.61	0.05	0.07	0.40
Leaves	0.72	0.63	0.67	0.72	0.72	0.72	0.49	0.21	0.68	0.72	0.68	0.70	0.54	0.64	0.47	0.63
Seeds	0.67	0.48	0.70	0.73	0.66	0.64	0.64	0.54	0.76	0.57	0.69	0.57	0.70	0.69	0.66	0.66
Segment	0.60	0.55	0.11	0.64	0.62	0.68	0.39	0.39	0.66	0.02	0.61	0.00	0.61	0.67	0.66	0.59
Libras	0.58	0.67	0.57	0.60	0.58	0.62	0.33	0.19	0.64	0.58	0.62	0.49	0.37	0.57	0.43	0.65
Ionosphere	0.13	0.29	0.13	0.15	0.31	0.08	0.08	0.08	0.07	0.03	0.32	0.00	0.06	0.10	0.11	0.38
Waveform	0.36	0.47	0.36	0.37	0.51	0.36	0.36	0.33	0.35	0.25	0.30	0.31	0.00	0.37	0.24	0.15
Ecoli	0.59	0.59	0.62	0.60	0.61	0.59	0.57	0.57	0.61	0.60	0.62	0.58	0.66	0.64	0.63	0.69
CANE9	0.38	0.52	0.22	0.05	0.42	0.51	0.21	0.18	0.45	0.34	0.38	0.36	0.57	0.28	0.25	0.56
Olivertti	0.83	0.87	0.83	0.82	0.83	0.86	0.80	0.16	0.88	0.72	0.85	0.80	0.29	0.88	0.48	0.69
Dermatology	0.86	0.70	0.80	0.77	0.80	0.90	0.77	0.93	0.83	0.63	0.89	0.74	0.94	0.93	0.71	0.94
Balance-scale	0.14	0.00	0.16	0.12	0.12	0.10	0.14	0.09	0.06	0.09	0.00	0.00	0.00	0.04	0.28	0.16
Appendicitis	0.19	0.18	0.21	0.00	0.16	0.16	0.16	0.18	0.14	0.26	0.23	0.00	0.00	0.33	0.23	0.19
Arcene	0.05	0.22	0.08	0.00	0.05	0.08	0.08	0.11	0.02	0.02	0.07	0.00	0.08	0.00	0.08	0.09
Optdigits	0.73	0.74	0.71	0.60	0.66	0.70	0.44	0.32	0.85	0.02	0.74	0.01	0.92	0.84	0.79	0.91
Robotnavi	0.11	0.20	0.11	0.11	0.08	0.11	0.11	0.12	0.08	0.05	0.02	0.06	0.24	0.09	0.05	0.24
SCC	0.74	0.72	0.74	0.77	0.74	0.75	0.54	0.75	0.81	0.60	0.68	0.65	0.85	0.79	0.77	0.80
Pendigits	0.67	0.72	0.67	0.70	0.69	0.69	0.43	0.31	0.75	0.74	0.71	0.60	0.86	0.70	0.71	0.84
USPS	0.44	0.57	0.45	0.43	0.41	0.51	0.25	0.15	0.61	0.37	0.55	0.26	0.78	0.41	0.41	0.75
Mean	0.51	0.48	0.46	0.50	0.52	0.53	0.43	0.35	0.49	0.39	0.49	0.38	0.45	0.49	0.41	0.52

a peeling-off mode, and the number of clusters is determined automatically. The detection of each cluster consists of two steps, i.e., initial cluster extraction and cluster expansion. Firstly, we use the largest-density one in the unclustered data as the cluster center, and then obtain a large-density area surrounding the cluster center as the initial cluster, based on DBSCAN. In expanding the initial cluster, we use a non-parametric method to determine the set of candidate data for inclusion into the cluster adaptively. Together with a termination criterion, the cluster expansion is terminated at the boundary between clusters, avoiding under-expansion and over-expansion simultaneously. In experiments with 50 synthetic and real datasets, our algorithm is shown to perform better than the previous DSet-DPC algorithm in both clustering quality and computation efficiency. It also compares favorably with some other DPC-based and non-DPC algorithms.

Our algorithm solves some major difficulties of the density peak clustering algorithm. First, the number of clusters is determined automatically, avoiding the difficulty of determining the number of clusters accurately. Second, the largest-density unclustered data is used as the cluster center in extracting each cluster, making it unnecessary to identify cluster centers with some criteria manually or automatically. Third, each cluster is obtained separately by expanding the cluster from the dense area to less dense area, and the density difference among different clusters has little influence on the clustering results. In contrast, in the DPC algorithm significant density differences across clusters have evident influence on the identification of cluster centers.

In experiments we also observe that our algorithm performs less satisfactorily in some cases, including very small cluster sizes, significant cluster overlap and clusters of non-Gaussian distribution. In addition, we simply use t-sne to reduce data dimension to 2 in dealing with high-dimensional data, which may result in significant information loss. In the future we will work on solutions to these drawbacks. With very small cluster sizes, we will consider avoiding too large initial clusters with new adaptive methods to extract the initial cluster. In order to deal with significant cluster overlap, a possible solution is to borrow ideas from Gaussian mixture models. As to clusters of non-Gaussian distribution, we plan to make more use of the DBSCAN algorithm, where the density threshold can be utilized to overcome the irregular distribution of data density. Finally, to obtain better results with high-dimensional data, we will explore better dimension reduction methods, and also similarity learning methods following. e.g., SPRG.

CRediT authorship contribution statement

Jian Hou: Conceptualization, Funding acquisition, Methodology, Software, Writing – original draft. **Houshen Lin:** Data curation, Software. **Huaqiang Yuan:** Funding acquisition, Project administration. **Marcello Pelillo:** Writing – review & editing.

Declaration of competing interest

None.

Data availability

Data will be made available on request.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grants No. 62176057 and No. 61972090.

Appendix A. Difficulty in identifying cluster centers in DPC

In identifying cluster centers, the original DPC algorithm distinguishes cluster centers from non-center data manually based on their difference in ρ and δ values. This practice leads to some problems in dealing with datasets of complex distributions, and many algorithms have been proposed to deal with these problems. However, accurate identification of all the cluster centers is still quite difficult, especially with real datasets where challenges including clusters of irregular shapes, cluster overlap and data of high dimension are frequent. In the case that the number of clusters is given, one cluster may have multiple data being identified as cluster centers, whereas another cluster has none. In the case that number of clusters is estimated automatically, the estimated number of clusters may be far from accurate. One important reason behind these problems lies in the assumption that cluster centers have both large ρ and large δ, whereas non-center data have either small ρ and/or small δ. While this assumption seems reasonable based on the description in Section 2.1, we notice that the *large* and *small* ρ and δ are fuzzy descriptions. In other words, no matter how researchers try to enlarge the difference between cluster centers and non-center data with respect to ρ and δ, there is no clear boundary between *large* and *small* values of ρ and δ. As a result, it is usually difficult to determine which and how many data points should be identified as cluster centers.

Appendix B. Expected properties of the initial cluster

We expect the initial cluster to possess the following two properties. First, data in the initial cluster have larger densities than other data in the same cluster, so that cluster expansion is able to include the outside smaller-density data into the cluster based on DPC. This property propels us to select a density threshold based method to differentiate between the data inside and outside the initial cluster. Second, the initial cluster should be large, to facilitate the subsequent cluster expansion, and in the meanwhile not too large, to avoid including data of other clusters. Evidently, a large density threshold leads to a small initial cluster, and a small threshold results in a large one. Considering that different clusters may have significantly different densities, a fixed density threshold is unlikely to be suitable for all clusters. Therefore the density threshold should be determined adaptively for each cluster.

Appendix C. Difficulty in expanding the initial cluster

The DPC algorithm assumes that one data is in the same cluster as its superordinate. Based on this assumption, a simple method to expand the initial cluster is as follows. For each data p_i outside the initial cluster, we find its superordinate p_i^*. If p_i^* is in the initial cluster, p_i is included into the initial cluster. One major problem with this simple method is that cluster centers of other clusters may be included. As illustrated in Fig. 2(a), C2 has the largest density in the right cluster, and its superordinate is in the left cluster. In this case, if we expand an initial cluster in the left cluster, C2 will be included into the left cluster. As C2 has the largest density in the right cluster, all other data in the right cluster are in the same cluster as C2. Therefore, if C2 is included into the left cluster, the whole right cluster will be merged into the left one.

In the DPC algorithm, this problem is solved by detecting all the cluster centers simultaneously. For example, in Fig. 2(b) the data C2 will be identified as the cluster center of the right cluster, and its label is not determined by finding its superordinate. As a result, the right cluster will not be merged into the left one. However, in our sequential algorithm the cluster centers and clusters are detected sequentially. With the dataset of Fig. 2, in expanding the initial cluster of the left cluster, C2 has not been identified as a cluster center yet, and therefore C2 and the whole right cluster will be merged into the left cluster. A straightforward approach to avoid this problem is to set a threshold on the distance δ. As cluster centers usually have larger δ than non-center data, we can reject to add C2 into the left cluster if its δ is greater than a threshold. Unfortunately, as mentioned in discussing the problems of the DPC algorithm, it is difficult to determine an appropriate threshold of δ to differentiate between cluster centers and non-center data.

Appendix D. Expected properties of the candidate set

We use Fig. 3 to illustrate how to determine S_{candi} in our algorithm. In cluster expansion, our aim is to include all the data from the left cluster and none of the data from the right cluster. In other words, before the cluster is expanded to the boundary between two clusters, the cluster expansion should not be terminated. To achieve this aim, S_{candi} cannot be too small. It should contain as many neighboring data of S_{ic} as possible, so that at least one candidate data can be included into S_{ic} and the cluster expansion is not terminated prematurely. In the meanwhile, S_{candi} cannot be too large, to avoid including the cluster centers of other clusters, as illustrated in Fig. 2(a). In summary, S_{candi} should contain only the closest data to S_{ic} (to avoid including cluster centers of other clusters), and all the closet data to S_{ic} (to ensure that at least one candidate data can be included into S_{ic}). Due to the complex data distributions in real datasets, it is reasonable to infer that the appropriate size of S_{candi} varies with different clusters and different datasets. This implies that S_{candi} should be determined adaptively for each cluster.

References

[1] M. Ester, H.P. Kriegel, J. Sander, X.W. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in: International Conference on Knowledge Discovery and Data Mining, 1996, pp. 226–231.

[2] J. Shi, J. Malik, Normalized cuts and image segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 22 (8) (2000) 167–172.

[3] X. Chen, W. Hong, F. Nie, J.Z. Huang, L. Shen, Enhanced balanced min cut, Int. J. Comput. Vis. 128 (7) (2020) 1982–1995.

[4] L. Bai, X. Cheng, J. Liang, H. Shen, Y. Guo, Fast density clustering strategies based on the k-means algorithm, Pattern Recognit. 71 (2017) 375–386.

[5] L. Wang, J. Huang, M. Yin, R. Cai, Z. Hao, Block diagonal representation learning for robust subspace clustering, Inform. Sci. 526 (2020) 54–67.

[6] J. Ma, Y. Zhang, L. Zhang, Discriminative subspace matrix factorization for multiview data clustering, Pattern Recognit. 111 (2021) 107676.

[7] M. Chen, C.-D. Wang, J.-H. Lai, Low-rank tensor based proximity learning for multi-view clustering, IEEE Trans. Knowl. Data Eng. 35 (5) (2023) 5076–5090.

[8] D. Huang, C.-D. Wang, J.-S. Wu, J.-H. Lai, C.-K. Kwoh, Ultra-scalable spectral clustering and ensemble clustering, IEEE Trans. Knowl. Data Eng. 32 (6) (2020) 1212–1226.

[9] Z. Yu, Z. Zhang, W. Cao, C.L.P. Chen, C. Liu, H.-S. Wong, GAN-based enhanced deep subspace clustering networks, IEEE Trans. Knowl. Data Eng. 34 (7) (2022) 3267–3281.

[10] M. Pavan, M. Pelillo, Dominant sets and pairwise clustering, IEEE Trans. Pattern Anal. Mach. Intell. 29 (1) (2007) 167–172.

[11] J.F. Brendan, D. Delbert, Clustering by passing messages between data points, Science 315 (2007) 972–976.

[12] A. Rodriguez, A. Laio, Clustering by fast search and find of density peaks, Science 344 (2014) 1492–1496.

[13] R. Mehmood, G. Zhang, R. Bie, H. Dawood, H. Ahmad, Clustering by fast search and find of density peaks via heat diffusion, Neurocomputing 208 (2016) 210–217.

[14] 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.

[15] 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.

[16] A. Gupta, S. Datta, S. Das, Fast automatic estimation of the number of clusters from the minimum inter-center distance for k-means clustering, Pattern Recognit. Lett. 116 (2018) 72–79.

[17] R. Heckel, H. Bölcskei, Robust subspace clustering via thresholding, IEEE Trans. Inform. Theory 61 (11) (2015) 6320–6342.

[18] J. Hou, H. Yuan, M. Pelillo, Towards parameter-free clustering for real-world data, Pattern Recognit. 134 (2023) 109062.

[19] A. Seyedi, A. Lotfi, P. Moradi, N.N. Qader, Dynamic graph-based label propagation for density peaks clustering, Expert Syst. Appl. 115 (2019) 314–328.

[20] J. Zhao, G. Wang, J.-S. Pan, T. Fan, I. Lee, Density peaks clustering algorithm based on fuzzy and weighted shared neighbor for uneven density datasets, Pattern Recognit. 139 (2023) 109406.

[21] Z. Guo, T. Huang, Z. Cai, A new local density for density peak clustering, in: Pacific-Asia Conference on Knowledge Discovery and Data Mining, 2018, pp. 426–438.

[22] G. Wang, Q. Song, Automatic clustering via outward statistical testing on density metrics, IEEE Trans. Knowl. Data Eng. 28 (2016) 1971–1985.

[23] R. Bie, R. Mehmood, S. Ruan, Y. Sun, H. Dawood, Adaptive fuzzy clustering by fast search and find of density peaks, Pers. Ubiquitous Comput. 20 (5) (2016) 785–793.

[24] X. Xu, S. Ding, H. Xu, H. Liao, Y. Xue, A feasible density peaks clustering algorithm with a merging strategy, Soft Comput. 23 (2019) 5171–5183.

[25] R. Liu, W. Huang, Z. Fei, K. Wang, J. Liang, Constraint-based clustering by fast search and find of density peaks, Neurocomputing 330 (2019) 223–237.

[26] F. Fang, L. Qiu, S. Yuan, Adaptive core fusion-based density peak clustering for complex data with arbitrary shapes and densities, Pattern Recognit. 107 (2020) 107452.

[27] Z. Liang, P. Chen, Delta-density based clustering with a divide-and-conquer strategy: 3DC clustering, Pattern Recognit. Lett. 73 (2016) 52–59.

[28] 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.

[29] Y. Liu, Z. Ma, F. Yu, Adaptive density peak clustering based on K-nearest neighbors with aggregating strategy, Knowl.-Based Syst. 133 (2017) 208–220.

[30] J. Hou, A. Zhang, N. Qi, Density peak clustering based on relative density relationship, Pattern Recognit. 108 (2020) 107554.

[31] M. Abbas, A. El-Zoghabi, A. Shoukry, DenMune: Density peak based clustering using mutual nearest neighbors, Pattern Recognit. 109 (2021) 107589.

[32] A. Lyon, Why are normal distributions normal? Br. J. Phil. Sci. 65 (3) (2014) 621–649.

[33] D. Pelleg, A. Moore, X-means: Extending k-means with efficient estimation of the number of clusters, in: International Conference on Machine Learning, Vol. 1, 2000, pp. 727–734.

[34] P. Fränti, S. Sieranoja, How much can k-means be improved by using better initialization and repeats, Pattern Recognit. 93 (2019) 95–112.

[35] P. Wang, Y. Yao, CE3: A three-way clustering method based on mathematical morphology, Knowl.-Based Syst. 155 (2018) 54–65.

[36] X. Zhu, C.C. Loy, S. Gong, Constructing robust affinity graphs for spectral clustering, in: IEEE International Conference on Computer Vision and Pattern Recognition, 2014, pp. 1450–1457.

[37] 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.

[38] H. Yu, L. Chen, J. Yao, A three-way density peak clustering method based on evidence theory, Knowl.-Based Syst. 211 (2021) 106532.

[39] Y. Li, L. Sun, Y. Tang, DPC-FSC: An approach of fuzzy semantic cells to density peaks clustering, Inform. Sci. 616 (2022) 88–107.

[40] J. Guan, S. Li, X. He, J. Chen, Clustering by fast detection of main density peaks within a peak digraph, Inform. Sci. 628 (2023) 504–521.

Jian Hou is a professor with the School of Computer Science and Technology, Dongguan University of Technology, Dongguan, China. His research interests include pattern recognition, machine learning, computer vision and image processing.

Houshen Lin is a master student with the School of Computer Science and Technology, Dongguan University of Technology, Dongguan, China. His research interests include data clustering and the application in pattern recognition.

Huaqiang Yuan is a professor with the School of Computer Science and Technology, Dongguan University of Technology, Dongguan, China. His research interests include intelligent computation and virtual reality.

Marcello Pelillo is a Professor of Computer Science at the University of Venice, Italy, where he directs the European Centre for Living Technology and leads the Computer Vision and Pattern Recognition group, which he founded in 1995. He held visiting research positions at Yale University (USA), McGill University (Canada), the University of Vienna (Austria), York University (UK), the University College London (UK), and the National ICT Australia (NICTA) (Australia). He serves (or has served) on the editorial boards of IEEE Transactions on Pattern Analysis and Machine Intelligence, IET Computer Vision, Pattern Recognition, Brain Informatics, and is on the advisory board of the International Journal of Machine Learning and Cybernetics. He has initiated several conferences series as Program Chair (EMMCVPR, IWCV, SIMBAD) and will serve as a General Chair for ICCV 2017. He is (or has been) scientific coordinator of several research projects, including SIMBAD, a highly successful EU-FP7 project devoted to similarity-based pattern analysis and recognition. He is a Fellow of the IEEE and a Fellow of the IAPR, and has been appointed IEEE Distinguished Lecturer (2016–2017 term).