Applied Computing and Intelligence

https://www.aimspress.com/journal/aci

Applied Computing and Intelligence
4(2): 164–183.
DOI: 10.3934/aci.2024010
Received: 15 October 2024
Revised: 16 October 2024
Accepted: 16 October 2024
Published: 28 October 2024

Research article

HBC: halo-based clustering using local comparative density

Le Li and Fei Wang*

School of Computing, University of Eastern Finland, Finland

* **Correspondence:** Email: feiwang@cs.uef.fi.

Academic Editor: Chih-Cheng Hung

Abstract: Clustering is a fundamental technique for revealing structure within data. Density-based clustering has the advantage of detecting structure without prior knowledge, such as the distribution or the number of clusters. However, existing density-based clustering algorithms usually suffer from limitations in latent density patterns, which hinder their applicability to complex datasets. We have showed that this may be caused by the bias of standard methods for point density estimation, such as kernel density estimation (KDE), which tends to favor compact clusters. To address the limitations of existing methods, we introduced the concept of local comparative density. We then proposed a specialized technique well-suited for capturing the nature of local comparative density. This approach allowed us to balance the density variations across clusters. Building on this foundation, we presented a halo-based clustering method that can effectively discover clusters with arbitrary shapes, heterogeneous densities, peakless distributions, and unbalanced cluster sizes. Additionally, this method is capable of identifying high-quality noise within the data. The superiority of the proposed algorithm was demonstrated by comparing it with state-of-the-art density-based clustering algorithms on artificial and real-world datasets.

Keywords: cluster analysis; density-based clustering; neighborhood analysis; absolute density estimation; relative density estimation

1. Introduction

Clustering analysis aims to gain insights into the intrinsic patterns and structures within data by grouping data objects into meaningful clusters such that objects within each cluster are highly similar to each other and dissimilar to objects in other clusters according to predefined criteria [1]. Various clustering algorithms have been proposed that can be roughly categorized as partitioning methods [2], density-based methods [3–7], hierarchical methods [8], and spectral clustering techniques [9]. Each category adopts distinct strategies for conceptualizing clusters and grouping data objects accordingly.

In general, clustering algorithms based on different principles exhibit particular strengths for specific task types or optimize different performance metrics. Meanwhile, inherent limitations exist for each clustering approach depending on its underlying assumptions and mechanisms.

Density-based clustering algorithms [10, 11] aim to recognize the various patterns from given density estimates. The density patterns can be the contiguous high-density areas [7], density ordering [12], density attractors [5], density peaks [13–15], or density subgraphs [4]. Theoretically and practically, most of the algorithms bear time complexities that lie between that of k-means [16] and hierarchical clustering such as Ward [8] and are able to handle relatively large-scale data [4,17,18]. In terms of adaptability, density-based clustering algorithms are non-parametric techniques that often make no explicit assumptions about cluster shapes and require less prior knowledge than other methods. Therefore, they have received wide attention. By incorporating the concept of density, density-based clustering algorithms take advantage of recognizing clusters of arbitrary shapes from compact regions of data space containing noise.

When handling the distributions like C2 and G3 displayed in Figure 1, common density estimate methods tend to assign much lower density values for the sparse clusters, as shown in Figure 2. The two datasets involve three challenging density distribution features: 1) significant density variation from cluster to cluster, 2) indistinguishable density valleys caused by heavily overlapping cluster boundaries, and 3) uniform or peak-less distributions. In other words, a particular density-based clustering algorithm may fail when one or more of the above distributions occur.

We propose the local comparative density estimation and the halo-based clustering algorithm. The contributions are summarized as follows: 1) We introduce a density estimation framework, called local comparative density, that allows for equal treatment of sparse and compact clusters. 2) We propose the concept of generalized nearest neighbors and show how to decompose it into three meaningful subspaces that can be seamlessly embedded into the framework, formulating an efficient and robust instance. 3) We propose a simple yet adaptable comparative density-based clustering method that works by detecting cluster halos through a global density threshold that can be empirically fixed.

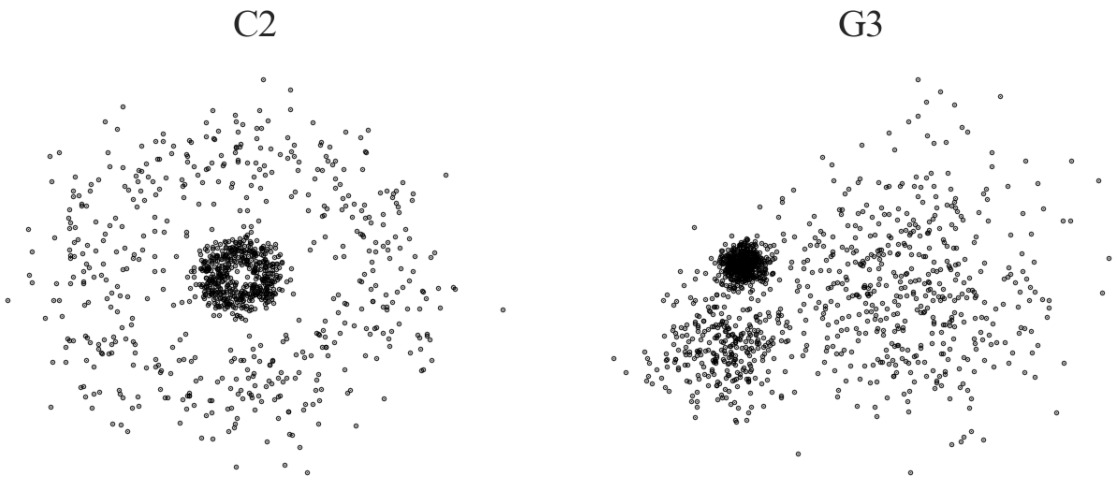

Figure 1. Two artificial datasets with varying density.

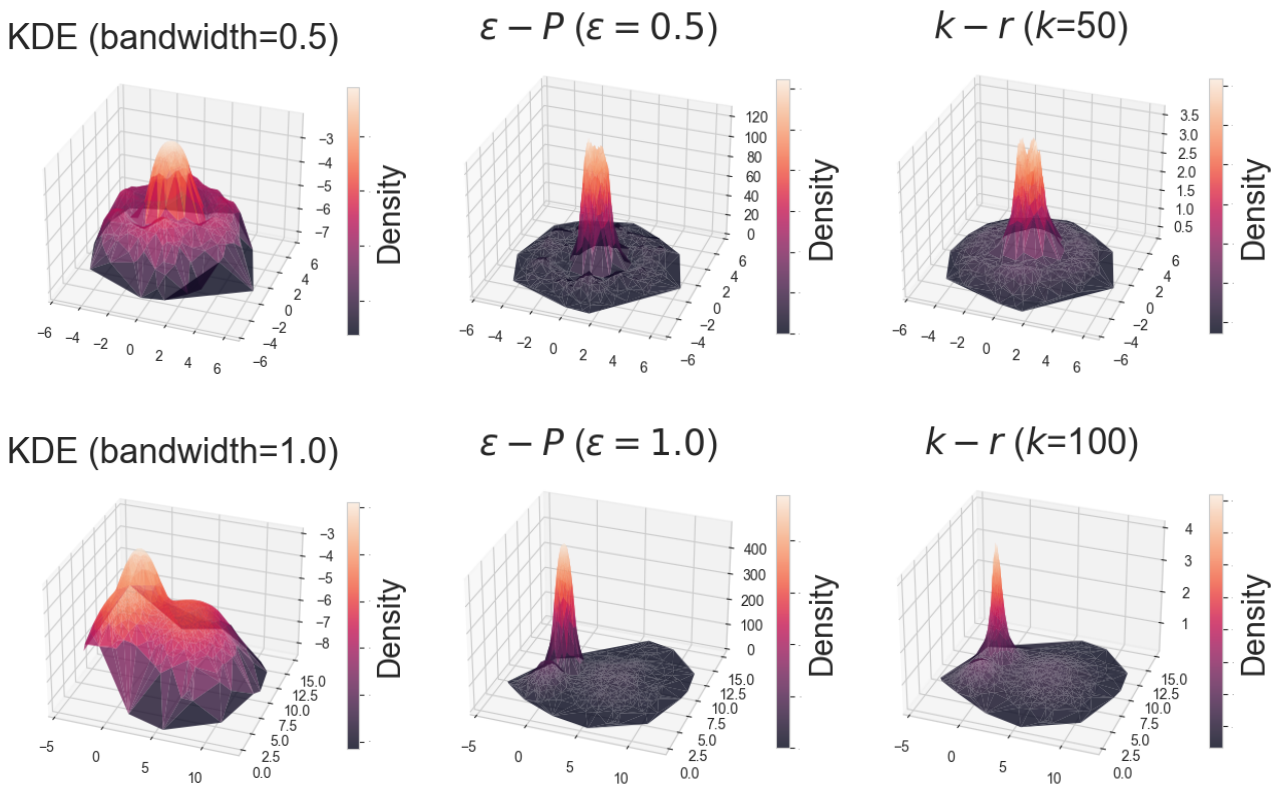

Figure 2. Density estimates by 3 common methods, ϵ-P, k-r, and Gaussian KDE for C2 and G3. They treat the sparser cluster(s) as less important.

2. Density-based clustering

In the rest of the paper, we sometimes use o, p, and q to represent some objects of a dataset $X = \{x_1, x_2, ..., x_N\} \subset R^d$, and d_{ij} represents the distance between x_i and x_j. Before introducing some existing density-based clustering algorithms, we first introduce some common density definitions.

Definition 1 (ϵ-P density)**.** *The density of x_i w.r.t. ϵ-P is defined as ϵ-$P(x_i) = \sum_j f(d_{ij} - \epsilon)$, where $f(x) = 1$ if $x > 0$ and $f(x) = 0$ otherwise. Here ϵ is a user-specified cutoff determining a neighborhood of x_i.*

Definition 2 (k-r density)**.** *k-r density of x_i is defined as k-$r(x_i) = 1/d_k(x_i)$, where $d_k(x_i)$ is the distance between x_i and its kth nearest neighbor.*

For any p, ϵ-P density fixes the radius of the neighborhood of p, depicting the local density of p as the number of neighbors within the neighborhood, while k-r density fixes the number of neighbors of p, describing the local density of p as the reciprocal of the radius of the neighborhood.

Density-based clustering algorithms using the above density definitions may fail when dealing with specific data distributions such as C2 and G3 because the resulting density estimates cannot reveal the necessary density patterns they require. DBSCAN [7, 19] uses ϵ-P density and defines a core point if it has a higher density than a given threshold. Clusters are formed by connecting dense regions of core points and their neighbors. DBSCAN cannot process significant density changes, mainly because ϵ-P

density tends to assign low-density values to sparse clusters, resulting in the density of sparse clusters falling below the given threshold.

Density peaks clustering (DPC) [6] computes two quantities for each point: density (ρ) and distance from points with a higher density (σ). Density peaks are identified from candidates with high ρ and σ, and clusters are formed as ascending density sequences pointing to the density peaks. The original DPC implementation using Gaussian KDE has a time complexity of $O(n^2)$, which can be computationally expensive for large datasets. To address this limitation, a fast density peaks algorithm was proposed in [20], which applies a fast and generic construction of an approximate k-nearest neighbor graph to accelerate density and delta calculation. In addition, when applying the mentioned density estimates, DPC fails to handle significant density variation since it is difficult for peaks to emerge from sparse areas. Moreover, DPC relies on the reliable recognition of density peaks and, therefore, suffers from peakless data distributions.

Denclue and its improved version [5, 21] model the overall density as a sum of the data points' influence functions (kernels). They apply a hill-climbing strategy to identify density attractors (local maxima of the density function) and assign points to the same cluster if they share the same attractor. These methods are sensitive to the kernel parameter as they depend on distinguishable density valleys to separate clusters.

OPTICS [12], the algorithm based on *hierarchical density estimates* (HDBSCAN) [17, 22] and the algorithm using *reverse nearest neighbor density estimates* (RNN-DBSCAN) [18] aim to address the limitations of DBSCAN. Both OPTICS and HDBSCAN achieve this by providing hierarchical representations of the clustering structure.

The key idea of OPTICS is to create an ordering of data points that reflects the density-based structure of the dataset. Unlike DBSCAN, which assigns points directly to clusters, OPTICS generates an ordering that captures the clustering structure at various density levels by introducing two main concepts: reachability distance and core distance. OPTICS further constructs a plot of these distances, allowing visualization and extraction of clusters with varying densities without requiring a fixed distance threshold.

HDBSCAN enhances the DBSCAN algorithm by introducing a hierarchical clustering structure rooted in mutual reachability distance. This method captures density-based connectivity among data points through the construction of a *minimum spanning tree* (MST), a technique explored in several studies for improving clustering performance [23, 24].

RNN-DBSCAN uses RNN counts to define core points and forms clusters by connecting core points and their RNNs. RNN-DBSCAN can automatically adapt to varying densities and does not require specifying ϵ, making it more robust to parameter settings. However, the authors did not shed light on the nature of RNN counts as a density measure.

To address density variations in clusters, DBHD [3] repeatedly finds a cluster from the current density peak based on two user-specified conditions until all points are processed. One can imagine the idea as being developed from the combination of DPC and DBSCAN since it progressively computes a density peak and then expands a cluster. RDMN [25] and RDR-DPC [26] incorporate modified density definitions to alleviate the impact caused by significant density variation. However, RDMN relies on revealing neighborhood relations through generating MST from fully connected graphs in an iterative manner, which can lead to unbearable time costs when dealing with large-scale data. While incorporating modified density helps handle density variation, it fails to improve the limitation rooted

in DPC in handling peak-less distributions.

3. Local comparative density

For the density-based clustering algorithm, it is desirable that the density estimation technique treats sparse and compact clusters equally without placing more emphasis on compact clusters. Comparing the density estimates shown in Figure 2 with those in Figure 3, the latter ones exhibit desirable properties: drastic density variations across clusters are eliminated, and density valleys between clusters are made distinguishable.

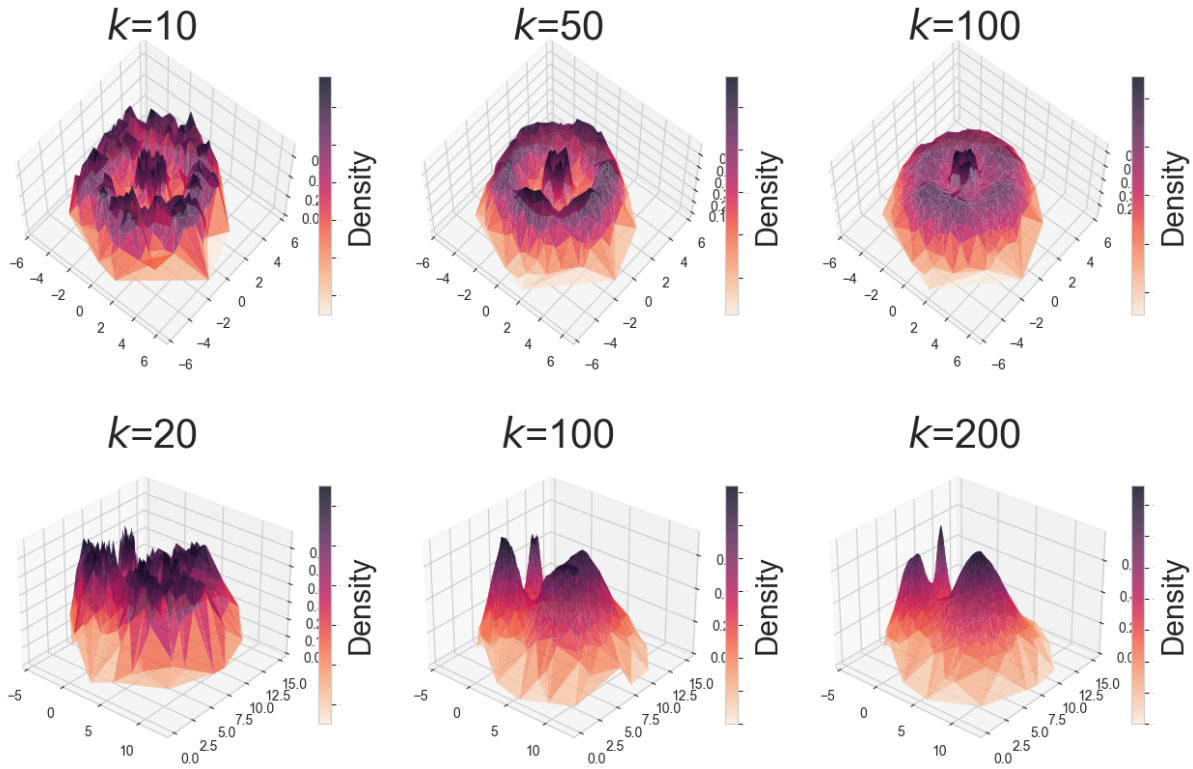

Figure 3. Density estimates with ρ_{GNN} on C2 and G3 by varying k. ρ_{GNN} provides homogeneous density estimates across clusters without being affected by the variations in data sparseness.

With this idea, we propose a new idea called *local comparative density* (LCD) estimation, which comprises three key elements:

- **Global measurement**: Obtain an arbitrary density estimate.
- **Local comparison**: Perform separate density comparisons within local neighborhoods.
- **Quantification**: Quantify densities of data points based on their relative density rankings.

Building upon the principles of local comparative density estimation, we propose a specialized method that inherently aligns with its nature and offers a more efficient approach compared to the generalized framework, as it does not necessitate an initial density estimate.

3.1. Local comparison based on generalized nearest neighbors

We first introduce the notion of generalized k nearest neighbors, which enables performing local comparison without explicitly computing absolute density.

3.1.1. Generalized k nearest neighbors

Let us recall some of the necessary concepts in advance. First, the k nearest neighbors of an object p, $NN_k(p)$, is a common concept used in machine learning, which intuitively determines the neighborhood containing the k nearest neighbors of p. On the contrary, the k reverse nearest neighbors of p, denoted as $RNN_k(p)$, is the set of objects that have p in their k nearest neighbors, and has also been widely used in cluster analysis [18]. By combining NN_k and RNN_k, we define $GNN_k(p)$, the generalized k nearest neighbors, as

Definition 3. *The generalized k nearest neighbors of p is defined as the union of $NN_k(p)$ and $RNN_k(p)$:* $GNN_k(p) = \{q|q \in NN_k(p)\} \cup \{q|q \in RNN_k(p)\}$.

The nearest-neighbor relationship between p and $GNN_k(p)$ is not always symmetric, while a symmetric nearest-neighbor space can be constructed from the intersection set of $NN_k(p)$ and $RNN_k(p)$, among which all the objects have p as their k nearest neighbors and vice versa [27, 28].

3.1.2. Implicit local comparison

We show how local comparison can be done implicitly by decomposing GNN_k without requiring explicit density estimates. Particularly, GNN_k can be decomposed into 3 special disjoint subspaces: the *density degradation space* (DDS), the *density gain space* (DGS), and the *density balance space* (DBS). Each of the 3 subspaces captures a type of relative density relationship. The details of the decomposition are given separately.

We first introduce the density degradation space, which is the subspace contained in $GNN_k(p)$, where all objects have higher density than p.

Definition 4. *For any object p and positive integer k, we define the density degradation space of p as*

$$DDS(p) = \{q|q \in GNN_k(p), p \notin NN_k(q)\}.$$

Recalling Definition 2, it can be proved that all the objects belonging to $DDS(p)$ have strict larger k-r density values than that of p.

Proposition 1. *For p and any $q \in DDS(p)$, we have $k\text{-}r(q) > k\text{-}r(p)$.*

Proof. $q \in GNN_k(p) \overset{1}{\Rightarrow} q \in NN_k(p)$ or $q \in RNN_k(p)$;
$p \notin NN_k(q) \overset{2}{\Rightarrow} q \notin RNN_k(p) \overset{3}{\Rightarrow} q \in NN_k(p)$;
$q \in NN_k(p) \overset{4}{\Rightarrow} d(p,q) \leq d_k(p)$;
$p \notin NN_k(q) \overset{5}{\Rightarrow} d(p,q) > d_k(q)$;
$d_k(q) < d(p,q) \leq d_k(p) \overset{6}{\Rightarrow} d_k(q) < d_k(p) \overset{7}{\Rightarrow} k\text{-}r(q) > k\text{-}r(p)$. □

As for the proof, implications 1 and 3 are because of Definition 3, while implications 2, 4, and 5 are because of the properties of NN_k and RNN_k. Additionally, 7 is because of Definition 2.

Let $|DDS(p)|$ be the cardinality of $DDS(p)$, and then the larger $|DDS(p)|$ is, the lower the ranking of p will be within $GNN_k(p)$. That is, a larger $|DDS(p)|$ implies p has a smaller relative density. As $|DDS(p)|$ grows, the relative density will keep decreasing. Note that $|DDS(p)|$ always lies in the range [0,k].

Next, we introduce *DGS*, which has a property completely opposite to density degraded space.

Definition 5. *For any object p, we define the density gain space as*

$$DGS(p) = \{q | q \in GNN_k(p), p \notin RNN_k(q)\}.$$

For each $q \in DGS_k(p)$, the density $k\text{-}r(q)$ is strictly smaller than that of p.

Proposition 2. *For p and any $q \in DGS_k(p)$, $k\text{-}r(q) < k\text{-}r(p)$ holds.*

Proof. $q \in GNN_k(p) \overset{1}{\Rightarrow} q \in NN_k(p)$ or $RNN_k(p)$;
$p \notin RNN_k(q) \overset{2}{\Rightarrow} q \notin NN_k(p) \overset{3}{\Rightarrow} q \in RNN_k(p)$;
$q \notin NN_k(p) \overset{4}{\Rightarrow} d(p,q) > d_k(p)$;
$q \in RNN_k(p) \overset{5}{\Rightarrow} p \in NN_k(q) \overset{6}{\Rightarrow} d(p,q) \leq d_k(q)$;
$d_k(p) < d(p,q) \leq d_k(q) \overset{7}{\Rightarrow} d_k(p) < d_k(q) \overset{8}{\Rightarrow} > k\text{-}r(q) < k\text{-}r(p)$. □

In the proof, implication 1 is because of Definition 3, while implications 2, 4, 5, and 6 are because of the properties of NN_k and RNN_k. Implication 8 is because of Definition 2.

A larger $|DGS(p)|$ implies p has a higher ranking within $GNN_k(p)$ and thus larger relative density. Therefore, we refer to this subspace as *DGS*. *DGS* actually sheds light on the nature of the density definition used in RNN-DBSCAN [18].

The last subspace decomposed from *GNN* is the density balance space.

Definition 6. *For any object p, we define the density balance space as*

$$DBS(p) = \{q \in GNN_k(p) | q \notin DGS_k(p) \cup DDS_k(p)\}.$$

Proposition 3. *For any $q \in DBS_k(p)$, it holds that $q \in NN_k(p), p \in NN_k(q)$.*

Proof. $q \in DBS(p) \Rightarrow q \in NN_k \cup RNN_k, q \notin DGS(q) \cup DDS(q)$;
$q \in NN_k \cup RNN_k, q \notin DGS(q) \cup DDS_(q) \Rightarrow q \in (NN_k(p) - DDS(p)) \Leftrightarrow q \in (RNN_k(p) - DGS(p))$;
$q \in (NN_k(p) - DDS(p)), q \in (RNN_k(p) - DGS(p)) \Rightarrow q \in NN_k(p), p \in NN_k(q)$. □

Considering Definitions 2–5, we have the following relationship:

Proposition 4. *$DDS(p)$, $DGS(p)$, and $DBS(p)$ are disjoint subspaces and the following relationship holds: $GNN_k(p) = DDS(p) \cup DGS(p) \cup DBS(p)$.*

Taking Propositions 3 and 4 together, we consider $q \in DBS(p)$ to have a balanced (relatively smooth) density relationship with p. So, we can approximately treat any q as forming an equal density relationship with p.

3.2. Quantification

With the results of local comparison given by the three decomposed subspaces, we define the local comparative density as

$$\rho_{GNN}(p) = \frac{1}{|GNN_k(p)|}\phi(GNN_k(p)), \tag{3.1}$$

where $|GNN_k(p)|$ normalizes the density and $\phi(\cdot)$ is a function quantifying the decomposition of GNN such that

$$\phi(GNN(p)) = |DGS(p)| \times \rho_g + |DBS(p)| \times \rho_b + |DDS(p)| \times \rho_d. \tag{3.2}$$

Here ρ_g, ρ_b, ρ_d are some weighting coefficients reflecting the properties of density gain, degradation, and balance. For example, we can set $(1, 0.5, 0)$ or $(1, 0, -1)$ for these weights. Figure 3 shows the density estimates from ρ_{GNN} on C2 and G3 using the first weight setting. Compared with that in Figure 2, ρ_{GNN} produces more desirable results, i.e., ρ_{GNN} treat sparse and compact clusters equally. Also, by significantly varying k, it can be observed that ρ_{GNN} is insensitive to the parameter k.

4. Clustering by cluster halo detection

In this section, we introduce our method of clustering by detecting cluster halos. The key idea is that by reliably identifying the points surrounding the clusters, which we refer to as the cluster halo, we can then obtain the isolated clusters that are defined by these cluster halos.

4.1. Cluster halo detection

We expect to effectively identify cluster halos through a density level set according to a global density threshold. This approach challenges the common density estimation methods, as they tend to yield heterogeneous density estimates across different clusters in cases where sparseness varies across clusters. However, this challenge can be easily addressed using the proposed LCD measure ρ_{GNN}, which provides homogeneous density estimates across clusters without being affected by the variations in data sparseness.

Definition 7. *Let $\rho_{max} = max\{\rho_{GNN}(p)|p \in X\}$, $\rho_{min} = min\{\rho_{GNN}(p)|p \in X\}$, and a halo threshold w.r.t. t is defined as $\gamma_t = \rho_{min} + (\rho_{max} - \rho_{min}) \times t$, where t is such that $0 < t < 1$.*

Definition 8. *Halo and Coreset are given by a γ_t-density level set: $halo = \{p \in X|\rho_{GNN}(p) < \gamma_t\}$, $Coreset = \{p \in X|\rho_{GNN}(p) \geq \gamma_t\}$, where γ_t is a density threshold.*

Here t affects the thickness of the halo, i.e., a larger t results in a thicker halo. Figure 4 shows the results of halo detection, where the detected halo can reliably isolate clusters.

By leveraging the LCD technique, we can reliably determine a global density threshold that allows us to extract the cluster halo regions surrounding the distinct cluster structures. This enables us to delineate the clusters in a more robust and consistent manner, overcoming the limitations of traditional density estimation approaches. This motivates us to propose the halo-based clustering method.

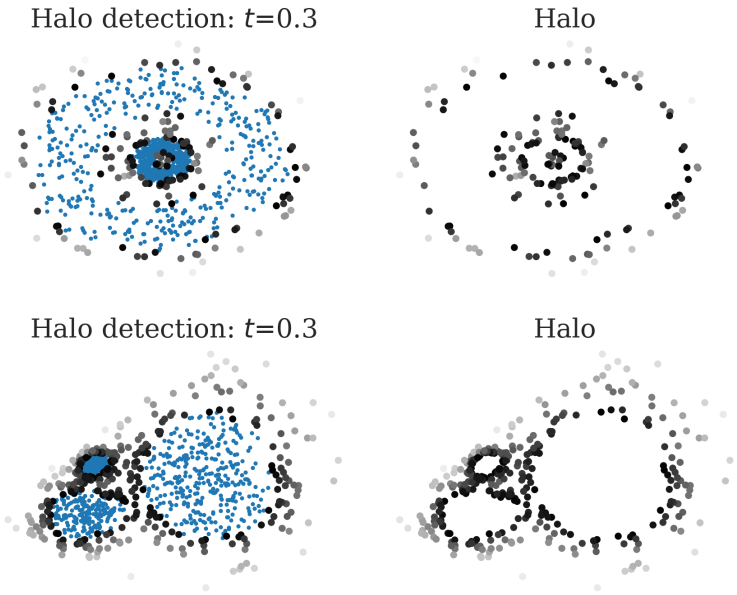

Figure 4. Halo detection with ρ_{GNN} on C2 and G3.

4.2. Halo-based clustering

The cluster halo serves as a boundary that helps delineate the distinct clusters, allowing us to identify the individual cluster structures. Specifically, we can repeatedly start from an arbitrary core point to expand a cluster within an isolated area defined by the detected halo until all the clusters are found. To facilitate a better understanding and description of how to recognize clusters from the independent spaces separated by a halo, we make some definitions subsequent to Definitions 7 and 8.

Definition 9. *For any $p \in coreset$, the adaptive radius ϵ for p is the distance between p and its nearest neighbor in halo: $\epsilon(p) = min_{d(p,q)}\{d(p,q)|q \in halo\}$.*

Definition 10. *Two core objects p, $q \in coreset$ are ϵ-reachable if $\epsilon(p) > d(p,q)$ or $\epsilon(q) > d(p,q)$.*

Definition 11. *Two core objects p and q are density-connected if they are directly or transitively ϵ-reachable.*

Definition 12. *A cluster C is a non-empty maximal subset of cores such that every pair of objects in C is density-connected.*

4.2.1. The relation and difference with DBSCAN

Through Definitions 7–12, we can relate the proposed halo-based clustering with DBSCAN. The main differences lie in 3 aspects: 1) we use ρ_{GNN} instead of ϵ-P for the density estimate, 2) coreset corresponds to the definition of the core objects of DBSCAN and halo contains the combination border objects and noise, and 3) for DBSCAN, clusters are formed as density-connected core objects with a fixed radius, while for HBC, the radius of each object is independently determined according to the detected halo so that the clusters are formed as colonies separated by the halo.

4.2.2. Pseudo code for HBC

The pseudo code shown in Algorithm 1 implements the proposed halo-based clustering. The inner while loop implements the expansion of a cluster: it starts from an unvisited core object and stops when all the ϵ-*reachable* objects are connected. Thus, a new cluster is formed after each round of the outer while loop. The outer while loop will repeatedly execute until the Coreset is empty, i.e., all clusters have been recognized. This way, a cluster grows from a series of self-adaptive expansions constrained within the enclosed spaces separated by the cluster halo. Finally, we can reassign the halo points to the clusters with the KNN algorithm according to cluster labels. In practice, we sometimes need to strip noise from the detected halo. This will be discussed and demonstrated with experiments in the following section. The source code of the algorithm is available in GitHub*.

Algorithm 1: Halo-based clustering (HBC).

Input: $X, k, t = 0.5$
Output: labels
$Halo = \{p \in X | \rho_{GNN}(p) < \gamma_t\}$
$Coreset = \{p \in X | \rho_{GNN}(p) \geq \gamma_t\}$
Initialize n-dimensional array $labels$
$labels[X] = UNVISITED$
$clusterNo = 0$
while $Coreset$ **do**
  $C = Coreset.pop()$
  while C **do**
    $o = C$.pop()
    labels$[o] = clusterNo$
    $\epsilon(o) = min\{d(o, q) | q \in Halo\}$
    $\hat{C} = \{p \in Coreset | d(p, o) < \epsilon(o)$ & $label[p] == UNVISITED\}$ $C = C \bigcup \hat{C}$
  end
  $Coreset = Coreset/C$
  $clusterNo+ = 1$
end
Strip noise from halo according to the knee point rule.
Reassign halo objects to the recognized clusters with the KNN algorithm.

4.2.3. Time complexity

Here ρ_{GNN} estimate requires a kNN query for all objects of X. Each round of the inner while loop involves a 1-NN query within Halo to obtain $\epsilon(o)$ and a $\epsilon(o)$-radius neighborhood within the Coreset for cluster expansion. It usually runs t rounds where $t << n$. Thus, the time bottleneck of HBC lies in the kNN query for all objects, which has the same time complexity as other methods based on kNN queries, such as DBSCAN and RNN-DBSCAN. Specifically, the worst-case time complexity is $O(n^2)$. However, when supported by index structures such as KDTree, R-Tree, or BallTree, the time complexity can be greatly reduced.

*https://github.com/uef-machine-learning/hbc

5. Parameter sensitivity and noise detection

This section investigates the performance of HBC on characteristics including density variations, overlapping boundaries, peakless distributions, complex shapes, and noise. The primary focus is to analyze the sensitivity of HBC to its two parameters: number of neighbors (k) and threshold for halo detection (t). Through these experiments, we aim to

1) Determine the influence of k and t on halo detection.
2) Provide an empirical reference for t.
3) Develop a heuristic search strategy for selecting the appropriate value of k.

Furthermore, we introduce a method for identifying noise within the datasets.

5.1. Parameter sensitivity analysis

As discussed in the previous section, parameter k affects the reliability of $\rho(GNN)$ density estimates, and t determines the thickness of the halo. To show the sensitivity of HBC in terms of these two parameters, we conduct experiments on C2 and G3 by varying k and t simultaneously. The results are shown in Figures 5 and 6. It can be observed that when t is too small (such as 0.1), the detected halo tends to be inadequate to isolate the true clusters accurately. In contrast, when t is large enough (0.3–0.5), the clusters can be well-isolated by the detected halo in most cases.

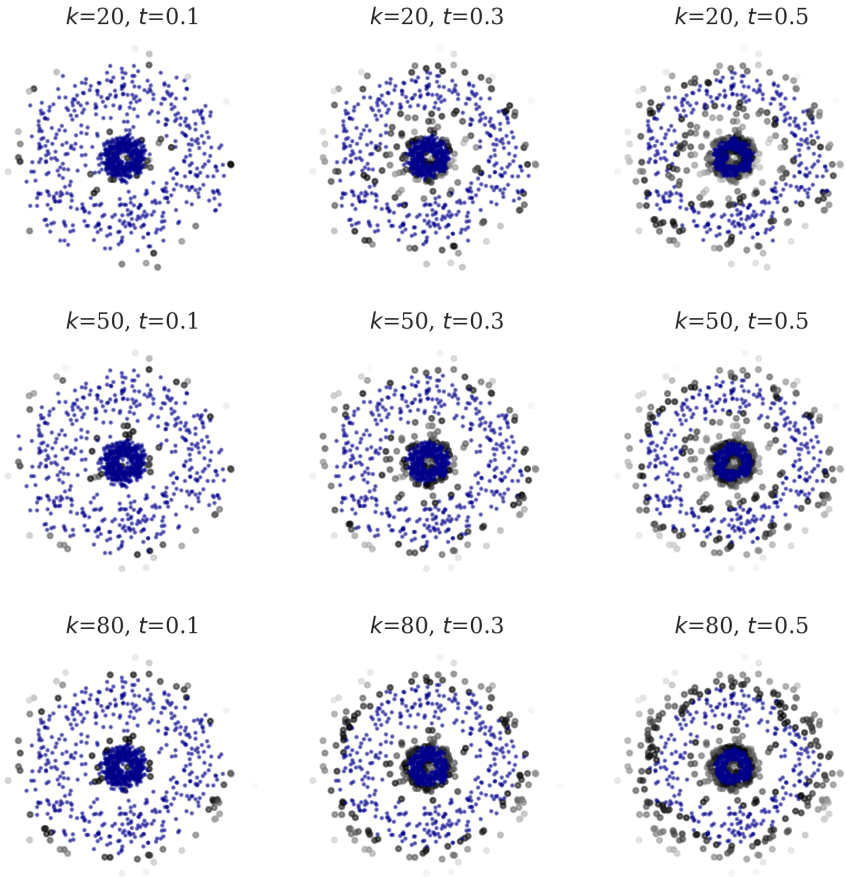

Figure 5. Sensitivity of halo detection to parameters k and t on C2. The black points represent the detected halo.

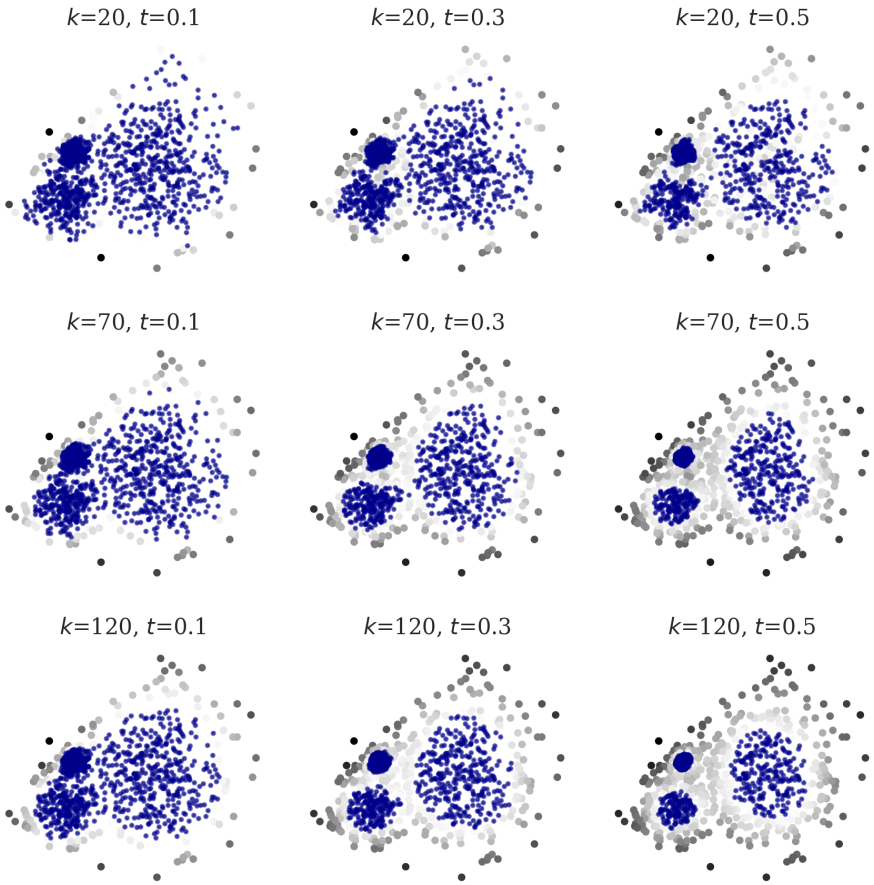

Figure 6. Sensitivity of halo detection to parameters k and t on G3. The blue points represent the cluster cores, and points with a gradient gray color correspond to the halo points with varying $\rho(GNN)$ densities. The darker the color, the smaller the $\rho(GNN)$ is.

With a large t, the reliability of the detected halo becomes less sensitive to k. In detail, despite the fact that the reliability may degrade when k is too small or too large, there are stable intervals of k yielding reliable results. As shown in Figures 5 and 6, we expect that when a large t is selected, the reliability of cluster detection tends to be less sensitive to k within a relatively stable interval (some abnormal values are ignored).

Motivated by the above discussion, we describe, in the following section, the strategy for determining the two parameters: fix t to a relatively large value (0.5) and then analyze the stable intervals of k.

5.2. Reference for parameter k

We show how to determine the candidates for the number of clusters through a pair of decision graphs constructed from a heuristic process. On this basis, we can obtain some stable intervals as a reference for determining parameter k that is used for halo detection. Figure 7 shows the heuristics of parameter selection on the artificial datasets, each of which consists of a pair of decision graphs, a line plot showing the change curve of the resulting number of clusters with respect to parameter k, and a bar plot showing the statistical frequencies of occurrence of candidate clusters. Intuitively, the relatively

high statistical values of the bar plots and the long stable ranges of the line plot most likely reflect the information about the correct number of clusters. It means that we can determine the parameter by analyzing the stable intervals of the line plot as well as the relatively higher bars of the bar plot. Particularly, in the following tests, we simply use the highest bar, the value of the maximal statistical frequency of occurrence. The parameter k is picked from the corresponding interval of the line plot where the same number of clusters was yielded. For example, we can infer from the bar plots for C2 and G3 shown in Figure 7 that the number of clusters should be 2 and 3, respectively. Then, we can set k in accordance with the corresponding stable intervals of the line plots.

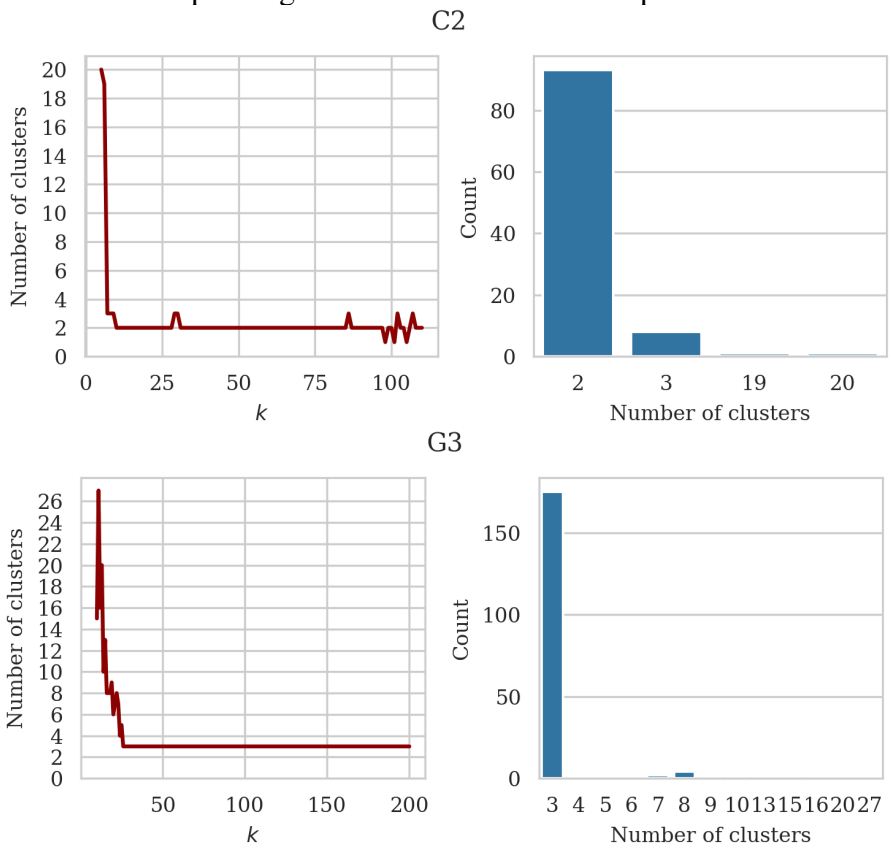

Figure 7. Decision graphs for parameter selection on C2 and G3.

5.3. Noise detection

Note that datasets C2 and G3 each contain random noise. Thus, we show how to detect noise from the detected halos by constructing a knee-shaped curve. This step should be performed after halo-based clustering and before assigning halos to the detected clusters. A knee curve is built by creating the density ascending order in terms of ρ_{GNN}. Then, we need to find the knee point of the curve around which the slope changes significantly, and noise can be identified as points with densities below the knee point. Various methods exist for identifying knee points, such as angle-based and curvature-based methods. We apply the method proposed in [29], which analyzes the knee from a difference curve built from the difference between the curve and its chord. As shown in Figure 8, the symbol 'x' marks the knee point given by the maximum of the difference curve, below which the curve reduction slows down significantly. The comparison in Figure 9 shows that the method can identify proper amounts of noise with higher quality.

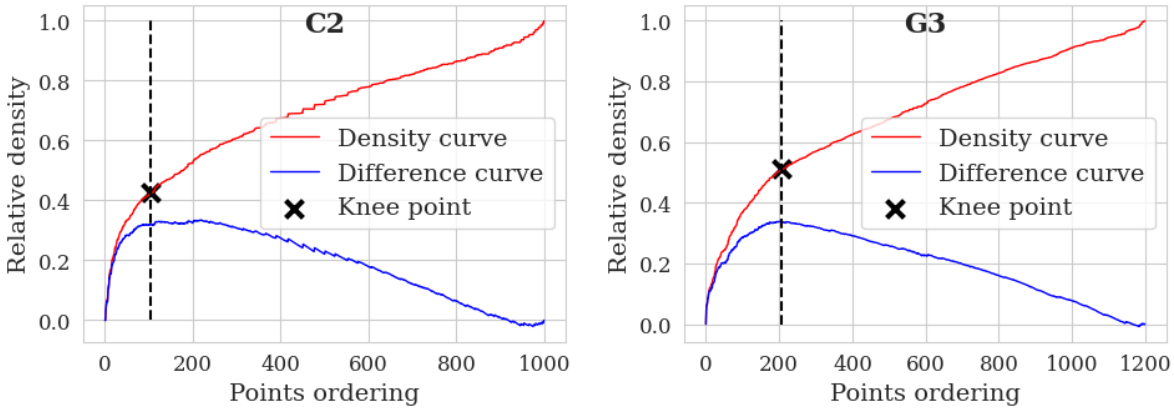

Figure 8. Identify noise according to inflection points of the knee-shaped curves. The x-axis represents the descending ordering of relative densities of the halo points. Points with relative density lower than the knee point are stripped as noise.

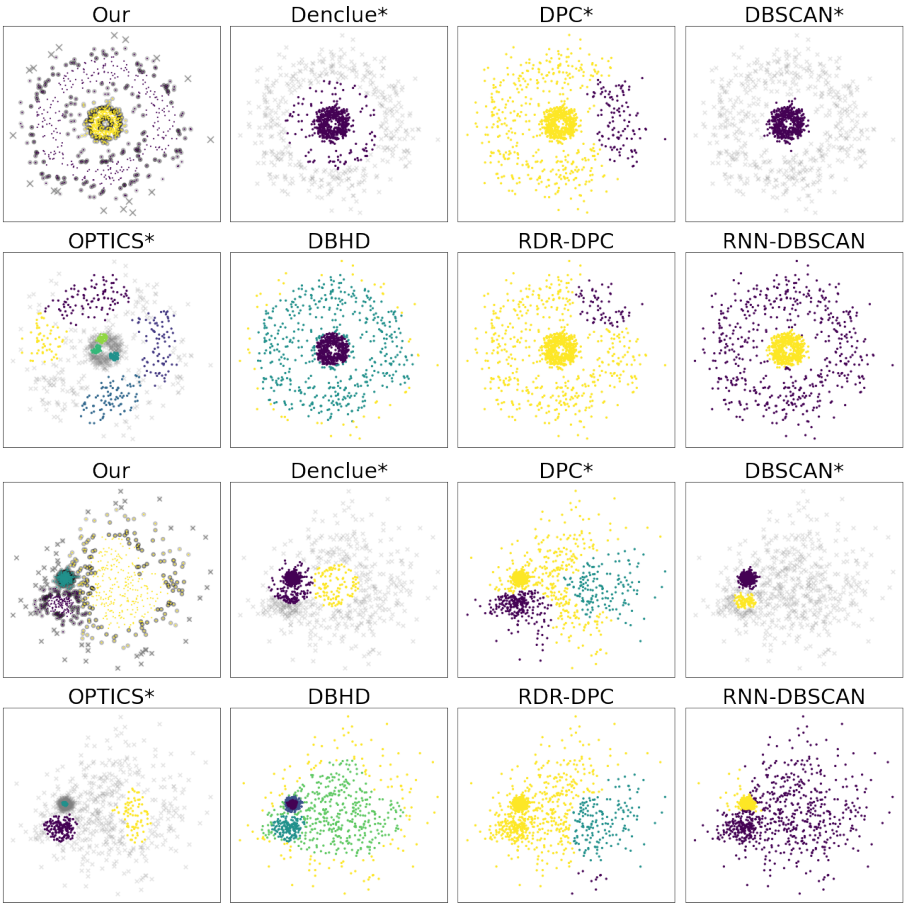

Figure 9. Results of density-based clustering algorithms on datasets C2 and G3. The solid points with different colors indicate the clusters they belong to, and the symbol 'x' represents noise. For our method, the shadowy points represent halo objects reassigned to the cluster body.

6. Experiments

6.1. Artrificial datasets

We evaluate the performance of HBC with fixed t (0.5) and k inferred from the decision graphs. The clustering quality is evaluated by comparing our method with other density-based clustering algorithms from the following aspects: the number and quality of the revealed clusters, the quantity and quality of detected noise, and the intuitive scatter distribution plot. The algorithms for comparison include Denclue 2.0 [21], DPC* [30], RDR-DPC [26], DBSCAN* [7], RNN-DBSCAN [18], OPTICS* [12], and DBHD [3]. Note that we add symbol * to stress that the algorithms are improved versions. Particularly, DPC* was implemented by combining the original DPC with Gaussian KDE and the automatic technique for determining the number of peaks. Note that DPC can also identify the noise. On the one hand, this requires additional manual intervention, and on the other hand, it does not have an impact on the cluster structure, so we simply ignore the noise identification step when running DPC.

Dataset C2 involves all three distribution characteristics that challenge the selected algorithms, which have been discussed in the first section. According to the clustering results shown in Figure 9, Denclue* and DBSCAN* identified the whole outer cluster as noise. OPTICS* recognized many more clusters than 2 real clusters. DPC* and RDR-DPC produced similar artifacts that disagree with the true clusters, although they used prior knowledge about the true number of clusters. Intuitively, the results of DBHD and RNN-DBSCAN are much better. However, DBHD recognized border points of the outer cluster as an independent cluster by mistake, while RNN-DBSCAN failed to identify the noise.

Dataset G3 involves significant density variation in clusters and indistinguishable density valleys. According to the clustering results shown in Figure 9, Denclue*, DBSCAN*, and OPTICS all involve the problem of identifying excessive noise. In addition, Denclue* and DBSCAN* ignored the sparse cluster on the right side entirely. On the other hand, DPC* and RDR-DPC still yielded undesirable results that were not inconsistent with the true cluster distribution. DBHD and RNN-DBSCAN failed to recognize the true number of clusters. Specifically, DBHD recognized two more clusters incorrectly, and RNN-DBSCAN merged two less compact clusters. In contrast, the HBC results are highly consistent with the true cluster distributions, and intuitively, the recognized noise is of high quality.

6.2. Real-life datasets

We also performed experiments on 8 UCI datasets [31] with different characteristics, including sizes, categories, and dimensions. The Olivetti face dataset [32] is a space-sparse dataset since the dimension of the embedded space is large and the number of objects belonging to a cluster is less than the number of clusters. See Table 1 for more details. The 7 density-based clustering algorithms used in the above subsection were selected to make a comprehensive comparison.

As for performance evaluation, the *Normalized Mutual Information Index* (NMI) [33] and *Clustering Accuracy* (ACC) [34] were used to evaluate the clustering performance. Scores close to 1 (0) indicate perfect (bad) results.

DBSCAN* uses two parameters to estimate the point density, which was tuned through a grid search from broad ranges. In addition to specifying parameter *minPts* for constructing the density ordering, OPTICS* also requires inferring ξ*-steep* to identify valleys from the reachability plot.

As recommended, OPTICS* sets the radius as infinity. The bandwidth for Gaussian KDE and the smoothing parameter for Denclue 2.0 were manually tuned. For DPC* and RDR-DPC, density peaks were picked automatically according to the true number of clusters. As for RNN-DBSCAN, the only used parameter, the number of neighbors for finding the reversed nearest neighbors, was set following the heuristics given in the original work. DBHD requires specifying three parameters: the minimum cluster size, the cluster member ratio, and the density deviation threshold. The last two parameters range from 0 to 1. These parameters were adjusted through a grid search from proper ranges to ensure good performances. For our method HBC, we used $t = 0.5$ for halo detection in most cases. Another parameter k was set following the procedure described in Section 5.

Table 1. Real-life datasets.

Dataset	objects	clusters	dimensions
Banknote authentication	1372	2	4
Segment	2310	7	19
Digits	1797	10	64
MiceProtein	1080	8	80
ReligiousBooks	590	8	8268
TCGA	801	5	20,531
Mnist-784	70k	10	784
Mfeat-zernike	1M	10	47
Olivetti-faces	400	40	4096

According to the results shown in Tables 2 and 3, the competitors fluctuate dramatically in performance when dealing with different datasets. On datasets Mnist-784 and Religiousbooks, the competitors generally achieve poor scores in terms of ACC and NMI. Such failures may be caused by the distribution assumption implicitly or explicitly made by the methods. For example, DBSCAN assumes homogeneous densities across clusters, OPTICS relies on the existence of distinguishable density valleys between clusters, and DPC, RDR-DPC, and Denclue suffer from peakless cluster distribution. As long as the true data distribution does not meet the assumption of the methods, they will inevitably produce unsatisfactory clustering assignations.

Table 2. Experimental results on 8 real-life datasets (NMI).

Alg/datasets	Banknote	Segment	Digits	Protein	Mfeat-Z	Mnist-784	Books	TCGA
DBSCAN*	0.52	0.68	0.88	0.70	0.65	0.42	0.46	0.87
RNN-DBSCAN	0.56	0.62	0.85	0.63	0.63	0.40	0.77	0.98
OPTICS*	0.36	0.63	**0.90**	0.68	0.88	0.69	0.68	0.90
Denclue2.0	0.46	0.60	0.87	0.47	0.68	0.44	0.31	0.87
DPC*	0.60	0.64	0.86	0.37	0.60	0.40	0.71	0.99
RDR-DPC*	0.22	0.66	0.87	0.35	0.60	0.45	0.81	0.99
DBHD	0.53	0.52	0.72	0.43	0.33	0.35	0.17	0.72
HBC	**0.93**	**0.71**	0.87	**0.86**	**0.89**	**0.85**	**0.97**	**1.00**

Table 3. Experimental results on 8 real-life datasets (ACC).

Alg/datasets	Banknote	Segment	Digits	Protein	Mfeat-Z	Mnist-784	Books	TCGA
DBSCAN*	0.71	0.57	0.86	0.32	0.60	0.45	0.28	0.79
RNN-DBSCAN	0.61	0.58	0.84	0.28	0.60	0.39	0.79	0.99
OPTICS*	0.18	0.27	0.87	0.25	**0.86**	0.28	0.25	0.79
Denclue2.0	0.65	0.27	0.88	0.02	0.63	0.01	0.01	0.88
DPC*	0.89	0.62	0.84	0.37	0.60	0.48	0.84	0.99
RDR-DPC*	0.68	0.58	0.85	0.34	0.60	0.50	0.89	0.99
DBHD	0.70	0.58	0.69	0.41	0.27	0.34	0.45	0.69
HBC	**0.99**	**0.73**	**0.90**	**0.57**	0.85	**0.71**	**0.99**	**1.00**

In comparison, our method shows desirable adaptability and achieves competitive or better performance over the best-performed competitors. In addition, the Olivetti-Face dataset is a space-sparse dataset that contains 64×64 gray-scale face images from 40 subjects, each of which contains only 10 samples. As shown in Figure 10, of the 10 subjects, HBC successfully recognized 9 clusters. After halo reallocation with KNN, the detected clusters are more consistent with true labels, resulting in better scores.

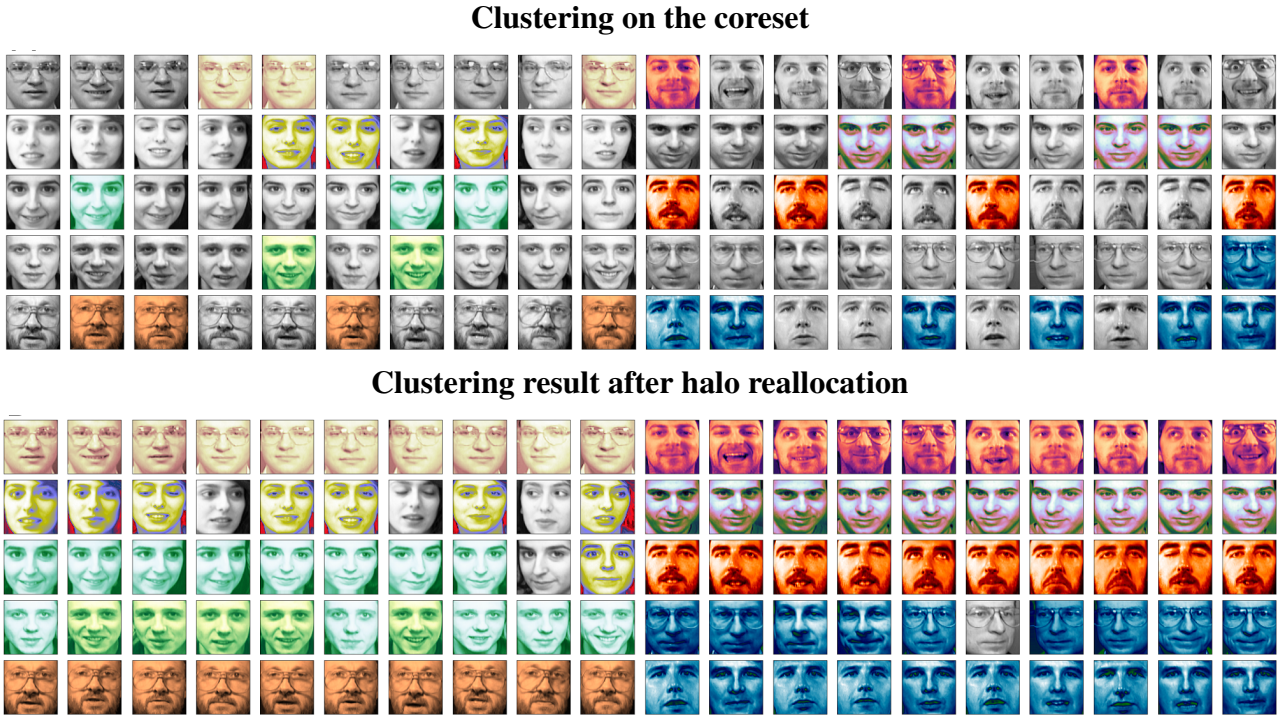

Figure 10. Halo-based clustering process on the Olivetti-Face dataset. The faces are colored according to the detected clusters. The gray-scale faces are halo objects in the upper graph and noise in the bottom graph.

Specifically, HBC achieves 14 out of the 16 best scores. This may be attributed to the fact that local comparative density and halo-based clustering extend the advantages of algorithms in detecting arbitrarily shaped clusters and overcome the distribution characteristics that challenge these density-

based clustering algorithms.

7. Conclusions

We proposed a simple yet effective halo-based clustering method based on the local comparative density, enabling the tight isolation of clusters by cluster halos through a global threshold. Experiments on artificial synthetic datasets that challenge density-based clustering methods, as well as on 9 real-world datasets with different characteristics, confirm the effectiveness of HBC. Moreover, the proposed local comparative density exhibits potential for improving other density-based clustering methods.

Conflict of interest

The authors declare that there is no conflict of interest.

References

1. A. Saxena, M. Prasad, A. Gupta, N. Bharill, O. P. Patel, A. Tiwari, et al., A review of clustering techniques and developments, *Neurocomputing*, **267** (2017), 664–681. http://dx.doi.org/10.1016/j.neucom.2017.06.053
2. A. K. Jain, Data clustering: 50 years beyond K-means, *Pattern Recogn. Lett.*, **31** (2010), 651–666. http://dx.doi.org/10.1016/j.patrec.2009.09.011
3. W. Durani, D. Mautz, C. Plant, C. Böhm, DBHD: density-based clustering for highly varying density, *Proceedings of IEEE International Conference on Data Mining (ICDM)*, 2022, 921–926. http://dx.doi.org/10.1109/ICDM54844.2022.00108
4. X. Zheng, C. Ren, Y. Yang, Z. Gong, X. Chen, Z. Hao, Quickdsc: clustering by quick density subgraph estimation, *Inform. Sciences*, **581** (2021), 403–427. http://dx.doi.org/10.1016/j.ins.2021.09.048
5. M. Khader, G. Al-Naymat, An overview of various enhancements of denclue algorithm, *Proceedings of the Second International Conference on Data Science, E-Learning and Information Systems*, 2019, 1–7. http://dx.doi.org/10.1145/3368691.3368724
6. A. Rodriguez, A. Laio, Clustering by fast search and find of density peaks, *Science*, **344** (2014), 1492–1496. http://dx.doi.org/10.1126/science.1242072
7. E. Schubert, J. Sander, M. Ester, H. P. Kriegel, X. Xu, DBSCAN revisited, revisited: why and how you should (still) use DBSCAN, *ACM T. Database Syst.*, **42** (2017), 19. http://dx.doi.org/10.1145/3068335
8. F. Murtagh, P. Legendre, Ward's hierarchical agglomerative clustering method: which algorithms implement Ward's criterion? *J. Classif.*, **31** (2014), 274–295. http://dx.doi.org/10.1007/s00357-014-9161-z
9. A. Damle, V. Minden, L. Ying, Simple, direct and efficient multi-way spectral clustering, *Inf. Inference*, **8** (2019), 181–203. http://dx.doi.org/10.1093/imaiai/iay008
10. P. Bhattacharjee, P. Mitra, A survey of density based clustering algorithms, *Front. Comput. Sci.*, **15** (2021), 151308. http://dx.doi.org/10.1007/s11704-019-9059-3

11. W. K. Loh, Y. H. Park, A survey on density-based clustering algorithms, In: *Ubiquitous information technologies and applications*, Berlin: Springer, 2014, 775–780. http://dx.doi.org/10.1007/978-3-642-41671-2_98

12. E. Schubert, M. Gertz, Improving the cluster structure extracted from optics plots, *Proceedings of the Conference "Lernen, Wissen, Daten, Analysen"*, 2018, 318–329.

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. 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. http://dx.doi.org/10.1016/j.neucom.2016.01.102

15. M. Parmar, D. Wang, X. Zhang, A. H. Tan, C. Miao, J. Jiang et al., Redpc: a residual error-based density peak clustering algorithm, *Neurocomputing*, **348** (2019), 82–96. http://dx.doi.org/10.1016/j.neucom.2018.06.087

16. D. Arthur, S. Vassilvitskii, *k-means++: the advantages of careful seeding*, Technical report, 2006.

17. L. McInnes, J. Healy, Accelerated hierarchical density based clustering, *Proceedings of IEEE International Conference on Data Mining Workshops (ICDMW)*, 2017, 33–42. http://dx.doi.org/10.1109/ICDMW.2017.12

18. A. Bryant, K. Cios, Rnn-dbscan: a density-based clustering algorithm using reverse nearest neighbor density estimates, *IEEE T. Knowl. Data Eng.*, **30** (2018), 1109–1121. http://dx.doi.org/10.1109/TKDE.2017.2787640

19. M. Ester, H. P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, *Proceedings of KDD-96*, 1996, 226–231.

20. S. Sieranoja, P. Fränti, Fast and general density peaks clustering, *Pattern Recogn. Lett.*, **128** (2019), 551–558. http://dx.doi.org/10.1016/j.patrec.2019.10.019

21. A. Hinneburg, H. H. Gabriel, DENCLUE 2.0: fast clustering based on kernel density estimation, In: *Advances in intelligent data analysis VII*, Berlin: Spring, 2007, 70–80. http://dx.doi.org/10.1007/978-3-540-74825-0_7

22. R. J. Campello, D. Moulavi, J. Sander, Density-based clustering based on hierarchical density estimates, In: *Advances in knowledge discovery and data mining*, Berlin: Spring, 2013, 160–172. http://dx.doi.org/10.1007/978-3-642-37456-2_14

23. C. Zhong, D. Miao, R. Wang, A graph-theoretical clustering method based on two rounds of minimum spanning trees, *Pattern Recogn.*, **43** (2010), 752–766. http://dx.doi.org/10.1016/j.patcog.2009.07.010

24. C. Zhong, D. Miao, P. Fränti, Minimum spanning tree based split-and-merge: a hierarchical clustering method, *Inform. Sciences*, **181** (2011), 3397–3410. http://dx.doi.org/10.1016/j.ins.2011.04.013

25. G. Mishra, S. Mohanty, Rdmn: a relative density measure based on mst neighborhood for clustering multi-scale datasets, *IEEE T. Knowl. Data Eng.*, **34** (2022), 419–432. http://dx.doi.org/10.1109/TKDE.2020.2982400

26. J. Hou, A. Zhang, N. Qi, Density peak clustering based on relative density relationship, *Pattern Recogn.*, **108** (2020), 107554. http://dx.doi.org/10.1016/j.patcog.2020.107554

27. C. Cassisi, A. Ferro, R. Giugno, G. Pigola, A. Pulvirenti, Enhancing density-based clustering: parameter reduction and outlier detection, *Inform. Syst.*, **38** (2013), 317–330. http://dx.doi.org/10.1016/j.is.2012.09.001

28. Y. Lv, T. Ma, M. Tang, J. Cao, Y. Tian, A. Al-Dhelaan, et al., An efficient and scalable density-based clustering algorithm for datasets with complex structures, *Neurocomputing*, **171** (2016), 9–22. http://dx.doi.org/10.1016/j.neucom.2015.05.109

29. V. Satopaa, J. Albrecht, D. Irwin, B. Raghavan, Finding a "kneedle" in a haystack: detecting knee points in system behavior, *Proceedings of 31st International Conference on Distributed Computing Systems Workshops*, 2011, 166–171. http://dx.doi.org/10.1109/ICDCSW.2011.20

30. 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. Sciences*, **354** (2016), 19–40. http://dx.doi.org/10.1016/j.ins.2016.03.011

31. *M. Kelly, R. Longjohn, K. Nottingham, The UCI machine learning repository*, University of California at Irvine, 2024. Available from: `https://archive.ics.uci.edu`.

32. F. S. Samaria, A. C. Harter, Parameterisation of a stochastic model for human face identification, *Proceedings of 1994 IEEE workshop on applications of computer vision*, 1994, 138–142. http://dx.doi.org/10.1109/ACV.1994.341300

33. A. Strehl, J. Ghosh, Cluster ensembles—a knowledge reuse framework for combining multiple partitions, *J. Mach. Learn. Res.*, **3** (2002), 583–617. http://dx.doi.org/10.1162/153244303321897735

34. P. Fränti, S. Sieranoja, Clustering accuracy, *Applied Computing and Intelligence*, **4** (2024), 24–44. http://dx.doi.org/10.3934/aci.2024003

AIMS Press