Multimedia Tools and Applications (2021) 80:32379–32407
https://doi.org/10.1007/s11042-021-11171-w

A neighborhood-based three-stage hierarchical clustering algorithm

Yan Wang [1] · Yan Ma [1] · Hui Huang [1]

Received: 22 October 2020 / Revised: 29 May 2021 / Accepted: 22 June 2021 /
Published online: 29 July 2021

Abstract
Many neighborhood-based clustering algorithms have been proposed to measure the similarity between data points or subclusters with their neighborhood information. However, most of them are vulnerable to the different cluster sizes, shapes and densities. In this paper, we propose a neighborhood-based three-stage hierarchical clustering algorithm (NTHC) which is robust to the difference. Three concepts, i.e., the stability of data point pair, the linked representatives, and the expanded representatives, are defined. Furthermore, a new measure of intercluster distance based on representatives is designed. In Stage 1, the outliers are detected and removed from the data set using reverse nearest neighbors. In Stage 2, small clusters are formed by merging the data points with stable connection on 1-nearest neighbor graph. In Stage 3, the final partitions are obtained by iteratively merging the closest pair of clusters based on the new measure of intercluster distance. Tests are carried out to compare the proposal with 15 other clustering algorithms. The experimental results on synthetic and real data sets demonstrate the proposed method is effective. In addition, we test the statistically significant differences among the sixteen clustering algorithms using the Friedman test. And the average rank value of the proposed algorithm is 4.19, which is superior to the other algorithms.

Keywords Similarity measure · 1-nearest neighbor · Shared nearest neighbors · Intercluster distance · Outlier detection · Hierarchical clustering

✉ Yan Ma
ma–yan@shnu.edu.cn

Yan Wang
yian1585@163.com

Hui Huang
Huanghui@shnu.edu.cn

[1] College of Information, Mechanical and Electrical Engineering, Shanghai Normal University, Shanghai, China

1 Introduction

Cluster analysis is an unsupervised learning method and aims to group a set of unlabeled objects into clusters, such that each cluster contains similar objects and different objects are in different clusters [58]. Clustering can help people analyze data and solve practical problems, and thus it has been widely used in disparate fields, including statistical analysis [2, 3, 16, 23], pattern recognition [14, 42, 64], information retrieval [15, 26], and bioinformatics [13, 17]. The traditional clustering algorithms can be divided into different categories, such as partition methods [7, 24], hierarchical algorithms [50], density-based algorithms [4, 29], and graph-based methods [11], and so on.

The density-based algorithms, such as DBSCAN (density-based spatial clustering of applications with noise) [29], OPTICS (ordering points to identify the clustering structure) [4], GDD (Gaussian Density Distance) [33] attempt to discover high density clusters separated by low density regions. The data points located in low-density regions are typically considered as outliers. Removing the outliers before executing a clustering task can potentially improve the clustering performance.

By introducing the concept of the k-reverse nearest neighbors (kRNNs), Vadapalli et al. [56] proposed reverse nearest neighbor based clustering and outlier detection algorithm (RECORD). RECORD counts the number of k-reverse nearest neighbors of data points, and takes a point that has less than k number of kRNNs as an outlier point. After removing all outlier points, the remaining points are clustered. Similarly, many algorithms exploit the kRNNs to further optimize the DBSCAN algorithm, such as IS-DBSCAN (enhancing density-based clustering algorithm) [10], ISB-DBSCAN (efficient and scalable density-based clustering algorithm) [46], and RNN-DBSCAN (density-based clustering algorithm using reverse nearest neighbor density estimates algorithm) [9], etc. These variants of DBSCAN use reverse nearest neighbor counts as the density of data point. However, they have not considered how to develop the clustering algorithm with the organic combination of multiple neighbor relations.

As one of the classical clustering algorithms, the core idea of hierarchical clustering is to produce a set of nested clusters organized as a hierarchical tree by investigating the similarity between subclusters. The hierarchical clustering algorithms can be divided into agglomerative algorithms and divisive algorithms. The advantage of hierarchical clustering is that it produces a hierarchical partition of the data at different levels of granularity. However, the hierarchical clustering suffers from the disadvantage that it cannot repair the previous merging decision. In addition, hierarchical clustering algorithm has high computational complexity of $O(n^3)$. Recently, several variants of the hierarchical clustering algorithm have been studied. Yu et al. proposed a minimum spanning tree (MST)-based agglomerative hierarchical clustering (TAHC) method [61] to detect clusters in artificial trees. Jeon et al. [38] proposed a new linkage method (NC-Link) for efficient hierarchical clustering of large-scale data. Bouguettaya et al. [8] presented a methodology (KnA) of combining agglomerative hierarchical clustering and partitional clustering to reduce the running cost. In addition, many hierarchical algorithms based on neighbor relation were proposed. Chameleon clustering algorithm [39] models the data points with a k-nearest neighbor graph which are further partitioned into a number of small sub-clusters. Lai et al. [31, 40] proposed an agglomerative clustering algorithm using a dynamic k-nearest-neighbor list to reduce the computational complexity of Ward's method. Ma et al. [48] proposed a three-stage MST-based hierarchical clustering algorithm, in which the neighbor relationship between the points is modeled with minimum spanning tree. The

reciprocal-nearest-neighbors supported hierarchical clustering algorithm (RSC) [57] is based on the idea that two reciprocal nearest data points should be grouped in one cluster. One specific neighbor relation has been used in the above algorithms. However, to our knowledge, how to organically combine various neighbor relations in the hierarchical clustering algorithm has not been explored in the literature.

In clustering analysis, similarity measure between data points or subclusters plays an important role in the process of clustering [18]. Many clustering algorithms use distance-based methods to measure the similarity, including Euclidean distance or cosine distance between the data points, and the average distance or maximum distance between subclusters, and so on. In recent years, the neighbor relation [51, 60] has been applied to measure the similarity. The shared nearest neighbor (SNN) algorithm [28, 37] developed a similarity measure between the points based on the number of the nearest neighbors they share. The reverse nearest neighbor based clustering (RECORD) algorithm [56] uses a reverse nearest neighborhood set to detect the outliers. Sarfraz et al. [55] proposed a parameter-free clustering algorithm (FINCH) using first neighbor relation. Qin et al. [51] proposed a clustering method based on hybrid *k*-nearest-neighbor (CHKNN) to find and merge the high-density regions. The key idea of the above similarity measures is to take the data point together with its neighbors as a whole and evaluate the similarity of two points by their neighbors' similarities instead of the distance between two points. Although the above algorithms proposed various neighborhood-based similarity measures, most of them have difficulty in processing complex data sets. For SNN, RECORD and CHKNN, it is difficult for a user to tune the parameters in realistic setting. Moreover, most neighborhood-based algorithms are limited to work with at most one or two neighbor relations.

In this study we propose a three-stage hierarchical clustering algorithm based on the nearest neighbor relation. Step 1: Outlier detection and removal. To alleviate the impact of outliers on the cluster structure, the reverse nearest neighbor method is used to detect and remove the outliers in the data set; Step 2: Preliminary clustering. The data points with stable connection on 1-nearest neighbor graph are merged to form small clusters; Step 3: Subcluster merging. By introducing the concepts of linked representatives and expanded representatives of two clusters, we propose a new measure of intercluster distance based on the representative points of two clusters. Based on this measure, we keep merging the closest pair of clusters until K clusters left. According to the extensive experiments on several synthetic and real data sets, the NTHC algorithm provides more accurate results than the state-of-the-art methods.

A valuable feature of the proposed method consists in its organic combination of various types of neighborhoods, which improves both the effectiveness and efficiency of the algorithm. Compared to the existing clustering methods, the proposed method can find clusters with different sizes, shapes and densities.

The main contributions of this paper are highlighted as follows.

(1) Based on the idea of various types of neighborhoods, we define three concepts: the stability of data point pair, the linked representatives, and the expanded representatives.
(2) We design a new measure of intercluster distance based on the linked representatives and extended representatives.
(3) A clustering algorithm termed NTHC is developed based on the new measure of intercluster distance. The NTHC method can deal with clusters with different densities and shapes.

(4) All the values of the three parameters (including k, th_r, and th_s) involved in the algorithm are fixed, thus the parameter tuning is not required.

The rest of the paper is outlined as follows. In Section 2, we discuss the outlier detection and neighborhood-based clustering algorithms. In Section 3, we present the proposed algorithm. In Section 4, experimental results are demonstrated to show the effectiveness of the proposed method. Some concluding remarks are given in Section 5.

2 Related work

2.1 Outlier detection

The concept of "outlier" was first proposed by Grubbs [43]: "An outlying observation, or outlier, is one that appears to deviate markedly from other members of the sample in which it occurs". Although there is no strict definition for outliers, it is accepted by researchers that outliers exhibit "deviating characteristics". Many researchers use such characteristics to improve clustering results by means of outlier detection.

In the research of clustering, outlier detection is commonly used in density-based clustering algorithms. Some clustering algorithms, such as DBSCAN and SNN, detect the outliers based on the densities of data points. In DBSCAN algorithm, the density of the data point x_i is obtained by counting the number of data points within its ε-neighborhood. The point with a density under a specified threshold, *MinPts*, is considered as an outlier and discarded later as noise. On the basis of DBSCAN, SNN redefines the density of data point using the shared nearest neighbors and further improves the DBSCAN algorithm. However, the two algorithms cannot work properly for multi density clusters.

RECORD algorithm regards the point with fewer kRNNs as an outlier. Similar algorithms, such as IS-DBSCAN algorithm, ISB-DBSCAN algorithm, and RNN-DBSCAN algorithm, also use kRNNs concept to detect the outliers. This kind of outlier detection method depends on the reverse nearest neighbors and is more flexible in dealing with clusters with different densities.

The outliers are kept far away from the cluster center, and lie on the boundary of the cluster. The outliers can have some effects on the accuracy of the clustering results, so it is beneficial to improve the clustering performance by removing the outliers. Figure 1(a) represents the outliers (denoted in red) identified by the proposed algorithm. Later in Section 3, we will discuss in detail about our algorithm. Figure 1(b) represents the clusters after eliminating outliers. It can be seen from Fig. 1 that eliminating outliers is a necessary preprocessing step in clustering, which can make the distribution of data points in the same cluster much closer.

2.2 Neighborhood-based clustering

Similarity measure based on shared nearest neighbors has been used to improve the performance of various types of clustering algorithms, including spectral clustering [21, 25], density peaks clustering [44, 47], k-means [20] and so on. As for hierarchical clustering, k-nearest-neighbor list is incorporated to reduce the computational complexity of Ward's method. In addition, the idea of k nearest neighbors has been introduced into density peaks clustering for the local density computation [27]. Recently, the concept of reciprocal nearest neighbors was

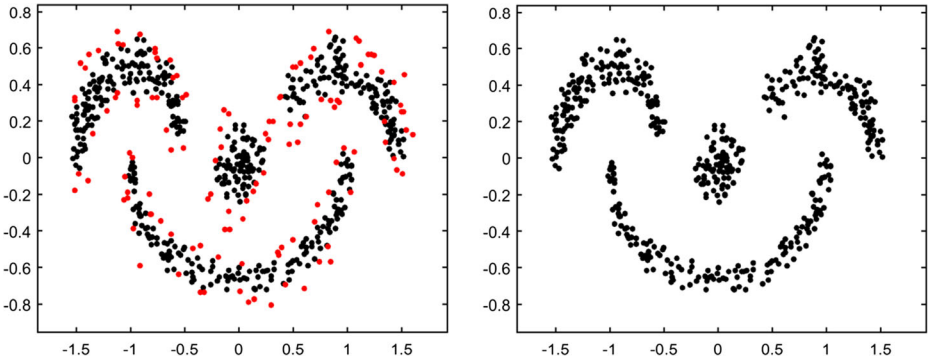


Fig. 1 Illustration of outlier detection and removal. (a) The outliers (denoted in red) identified by the proposed algorithm. (b) The clusters after outlier removal

investigated for the development of clustering algorithm. This concept is used to identify dense regions [1], discover the potential core clusters [54], distinguish the cluster membership of each point [30], and so on. Besides, to improve the performance of DBSCAN, reverse nearest neighbor methods such as RECORD, IS-DBSCAN, ISB-DBSCAN, and RNN-DBSCAN, define the density of point using reverse nearest neighbors. These methods require one single parameter k, the number of nearest neighbors.

The definition of a neighborhood can be categorized into two major types. The first one counts the number of points in a hypersphere of radius r. The second one is k-nearest neighbors, which has a wide range of applications in classification, clustering and outlier detection [9, 12, 51, 57]. Its main idea is to find the k nearest neighbors for all points in the data set. Assume that $X_{N\times d} = \{x_1, x_2, \Lambda, x_N\}$ is the data set X with N data points and d dimensions. The distance matrix D of data set X is a $N \times N$ two-dimensional matrix, where $D(x_i, x_j)$ is the Euclidean distance between points x_i and x_j:

$$D(x_i, x_j) = \sqrt{\sum_{m=1}^{d} (x_{im} - x_{jm})^2} \tag{1}$$

Starting with the distance matrix D, we can find the k nearest neighbors of x_i, denoted by $kNN(x_i)$:

$$kNN(x_i) = \{x_j | D(x_i, x_j) \leq D(x_i, NN_k(x_i))\} \tag{2}$$

where $NN_k(x_i)$ is the kth nearest neighbor of x_i.

After determining the k nearest neighbors of data point, the number of shared neighbors between two points can be used to define the similarity between points:

$$similarity(x_i, x_j) = size(kNN(x_i) \cap kNN(x_j)) \tag{3}$$

According to the clustering algorithm based on the shared nearest neighbors (SNN), the more neighbors the two points share, the more similar they are. SNN algorithm defines the similarity between a pair of points in terms of how many nearest neighbors the two points share. Based on the definition of similarity, SNN algorithm identifies core points and then builds clusters around the core points. And all non-core points that are not within a radius of *Eps* of a core

point are discarded. Ye et al. [60] improved the SNN algorithm by considering both the number of shared nearest neighbors and the distance between data points.

In general, the number of shared nearest neighbors is compared to a pre-specified threshold in the practical application, from which we can evaluate the similarity between two points. Here, the threshold value plays a vital role in the similarity comparison. And too large or too small values will affect the performance of final clustering result.

For data point x_i, its k-reverse nearest neighbors can be defined as:

$$kRNNs(x_i) = \left\{x_j | D(x_i, x_j) \leq D\left(x_j, NN_k(x_j)\right)\right\} \tag{4}$$

where $NN_k(x_j)$ is the kth nearest neighbor of x_j. Obviously, if x_i is one of k nearest neighbors of x_j, x_j is one of k-reverse nearest neighbors of x_i, that is, $x_i \in kNNx_j \Leftrightarrow x_j \in kRNNs(x_i)$.

As discussed in [51], the reciprocal neighbor relation is defined as:

$$NN_1(x_i) = x_j \wedge NN_1\left(x_j\right) = x_i \tag{5}$$

where $NN_1(x_i)$ and $NN_1(x_j)$ denote the first nearest neighbor of x_i and x_j, respectively.

Table 1 shows the main strengths and shortcomings of the various neighborhood based clustering algorithms. This table evaluates the methods using 11 properties, including *K-nearest neighbor*, *Shared nearest neighbors*, *K-reverse nearest neighbors*, *Reciprocal-nearest-neighbors*, *Detection of outliers*, *Number of adjustable parameters*, *Computational complexity*, *Varying densities*, *Irregular shapes*, *Varying sizes*, and *Parameter sensitivity*. *K-nearest neighbor*, *Shared nearest neighbors*, *K-reverse nearest neighbors* and *Reciprocal-nearest-neighbors* are used to show whether the type of neighbor structure is used in the algorithms ("√" denotes used and "×" denotes not used). *Detection of outliers* includes three values: able(√), unable(×), and partial(△) to show the outlier detection capability of the clustering algorithm. △ represents that the clustering algorithm can detect outliers for partial data sets. *Number of adjustable parameters* shows the number of user-defined parameters in the clustering algorithm. *Computational complexity* shows the amount of resources needed to perform certain computations. *Varying densities*, *Irregular shapes* and *Varying sizes* refer to the ability of the algorithm to discover clusters of various densities, irregular shapes and various sizes, respectively. They all include three values: able(√), unable(×), and partial(△). *Parameter sensitivity* evaluates whether the algorithm is sensitive to the present parameters.

From Table 1, we can see that several methods such as SNN_0, HC-MNN, Chameleon, SNN, and RECORD are sensitive to the parameters. In contrast, for NC, DAN and FINCH, there are no parameters required to tune. All of the three methods use reciprocal nearest neighbors. But both of NC and DAN require $O(n^3)$ computations. Four variants of DBSCAN (IS-DBSCAN, ISB-DBSCAN, RNN-DBSCAN, and ADBSCAN) can identify the clusters of varying sizes and irregular shapes. And three of them use k-reverse nearest neighbors. Six methods (HC-MNN, Chameleon, DLA, DKNNA, MUNEC, and RSC) are hierarchical clustering algorithms. To reduce the computational complexity of traditional hierarchical clustering method, various neighbor relations (k-reverse nearest neighbors, reciprocal nearest neighbors) are used in the DLA, DKNNA, and RSC methods. Although these methods reduce significantly the computational complexity of hierarchical clustering, there are still some problems among these methods. For example, the performance of these methods will be degraded if there are outliers in the clusters. Furthermore, most of them cannot guarantee the

Table 1 Comparison of clustering algorithms involving various neighbor relations. Note that N is the number of data points, k is the number of nearest neighbors, where'x' refers to unable,'√' means able and'△' is partial

Method	Publication year	neighbor relations				Outlier detection	Varying densities	Varying sizes	Irregular shapes	Number of adjustable parameters	Parameter sensitivity	Computational complexity
		K-nearest neighbor	K-reverse nearest neighbors	Reciprocal nearest neighbors	Shared nearest neighbors							
SNN_0 [37]	1973	√	x	x	√	x	△	△	√	2	High	$O(N^2)$
HC-MNN [32]	1978	√	x	√	x	x	△	△	√	1	High	$O(N^2)$
Chameleon [39]	1999	√	x	x	x	x	√	√	√	3	High	$O(N^2)$
SNN [28]	2003	√	x	x	√	√	x	√	√	3	High	$O(N^2)$
RECORD [56]	2006	√	√	x	x	√	x	√	√	1	High	$O(N^2)$
DLA [31]	2006	√	√	x	x	x	△	△	√	1	Low	$O(N\log(N))$
KNN-NC [49]	2009	√	x	√	x	△	x	√	√	1	Medium	$O(N^2)$
DKNNA [40]	2011	√	√	x	x	x	△	△	√	1	Low	$O(N\log(N))$
IS-DBSCAN [10]	2013	√	√	x	x	√	△	√	√	1	Medium	$O(N\log(N))$
NC [36]	2015	√	x	√	x	△	√	△	√	0	Low	$O(N^3)$
DAN [35]	2015	√	x	√	x	△	√	△	√	0	Low	$O(N^3)$
ISB-DBSCAN [46]	2016	√	√	x	x	√	△	√	√	1	Medium	$O(N\log(N))$
DPC-KNN [27]	2016	√	x	x	x	△	△	√	△	1	Medium	$O(N^2)$
APDC-KNN [59]	2017	√	x	x	x	△	△	√	△	1	Medium	$O(N^2)$
CHKNN [51]	2018	√	x	√	x	√	x	√	√	3	High	$O(kN^2)$
RNN-DBSCAN [9]	2018	√	√	x	x	√	△	√	√	1	Medium	$O(N\log(N))$
SNN-DPC [44]	2018	√	x	x	√	△	△	√	△	1	Medium	$O(N^2)$
FINCH [55]	2019	√	x	√	x	x	x	√	√	0	Low	$O(N\log(N))$
MUNEC [53]	2019	√	x	√	x	△	x	√	√	1	Medium	$O(N^2)$
RSC [57]	2020	√	x	√	x	x	x	√	√	1	Medium	$O(N\log(N))$
ADBSCAN [41]	2020	√	x	√	x	√	△	√	√	2	Medium	$O(N\log(N))$

results when the clusters have large variations of densities. Therefore, the goal of this paper is to incorporate different neighbor relations into hierarchical clustering while guaranteeing efficiency and effectiveness.

3 The proposed algorithm

In this section, we present a detailed description of the NTHC algorithm. First, we define the stability of the data point pair and the intercluster distance based on the linked and extended representatives. Then, we propose the NTHC algorithm, which includes outlier detection and removal, preliminary clustering and subcluster merging. Finally, we present the analysis of the time complexities of the NTHC algorithm.

3.1 Definitions

The number of *k*RNNs for each point x_i in the data set X can be represented by Eq. (6).

Definition 1 (the number of *k*-reverse nearest neighbors). For any point x_i in data set X, the set of *k*-reverse nearest neighbors of point x_i is $kRNNs(x_i)$; the number of *k*RNNs of point x_i can be expressed as

$$r(x_i) = |kRNNs(x_i)| \tag{6}$$

Then, we construct the nearest neighbor graph $G_{1\text{-}NN}$ as follows:

Definition 2 (1-Nearest Neighbor Graph). Let $G_{1\text{-}NN}$ be a 1-nearest neighbor graph in which x_i and x_j are connected by a directional edge, if x_j is the first nearest neighbor $NN_1(x_i)$ of x_i.

There exist two cases with respect to the connection relation on $G_{1\text{-}NN}$:

(1) If the data points x_i and x_j are the first nearest neighbors to each other, the edge connecting x_i and x_j is bidirectional. Here, this structure, composed of vertex x_i, x_j, and the corresponding bidirectional edge e_{ij}, is considered as stable. Thus, vertices x_i and x_j can be grouped into the same cluster.

Figure 2 illustrates an example of $G_{1\text{-}NN}$, where the points that are the first nearest neighbors to each other can be merged into one cluster (marked with red dashed circle).

(1) If x_j is the first nearest neighbor of x_i, and x_i is not the first nearest neighbor of x_j, the edge (x_i, x_j) is a one-way edge. This one-way relation cannot guarantee the stability of structure. As shown in Fig. 2, m is the first nearest neighbor of n, but n is not the first nearest neighbor of m. In this case, we can further judge with the shared nearest neighbors whether this one-way relation is stable or not.

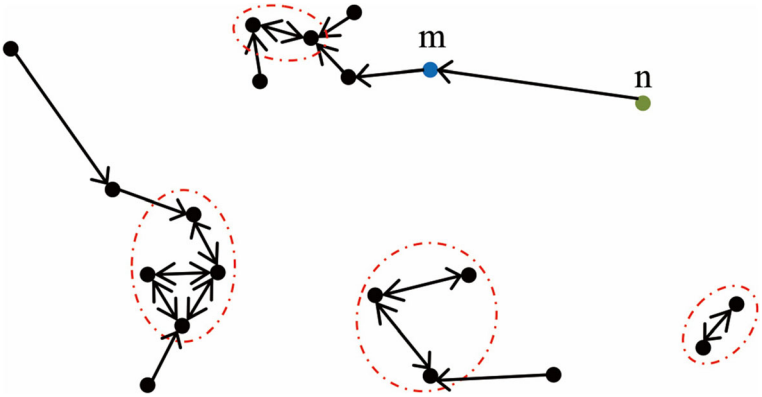


Fig. 2 An example of the nearest neighbor graph. Each point represents a data point. The one-way edge (m, n) represents that m is the first nearest neighbor of n

Based on the above discussion, we introduce the definition of the stability of the data point pair:

Definition 3 (Stability of the data point pair) The stability of the data point pair $s(x_i, x_j)$ is defined as follows:

$$s(x_i, x_j) = \begin{cases} 1, if\ (NN_1(x_i) = x_j \wedge NN_1(x_j) = x_i) \vee (size(kNN(x_i) \cap kNN(x_j)) \geq th_s \cdot k) \\ 0, else \end{cases} \quad (7)$$

where k is the value of k in k nearest neighbors, $NN_1(x_i)$ and $NN_1(x_j)$ denote the first nearest neighbor of x_i and x_j, respectively, and $kNN(x_i)$ and $kNN(x_j)$ denote the k nearest neighbors of x_i and x_j, respectively.

Let C_i and C_j denote two clusters, respectively, and we define the linked representatives of C_i and C_j as follows:

Definition 4 (The set of linked representatives of C_i relative to C_j) The set of linked representatives of C_i relative to C_j can be defined as follows:

$$R_{i|j} = \{x_i | x_i \in C_i, \exists x_j \in C_j, x_i \in kNN(x_j) \wedge x_j \in kNN(x_i)\} \quad (8)$$

According to Definition 4, we can select the data point pairs satisfying k shared nearest neighbors in C_i and C_j as the linked representatives of those two clusters, respectively. As an example, in Fig. 3, the sets of data points marked with green circles and purple circles represent $R_{i|j}$ and $R_{j|i}$, respectively. Since there are few point pairs satisfying k shared nearest neighbors, we further define the extended representatives to ensure the number of representatives without losing accuracy.

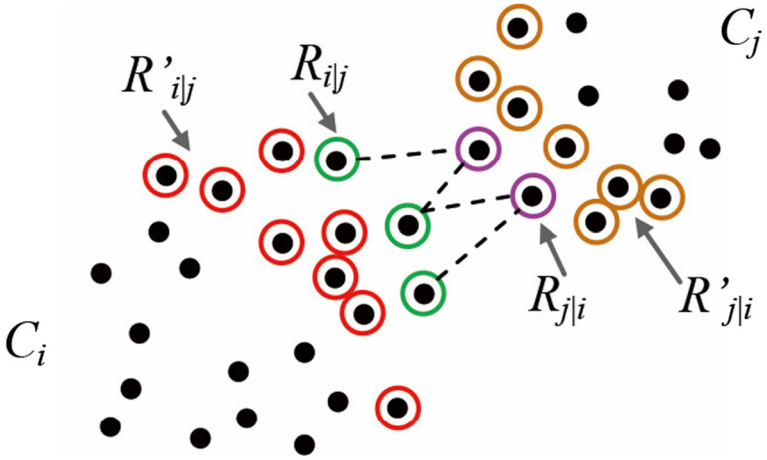


Fig. 3 An example of the set of representatives. The green circles and purple circles represent the linked representative points of C_i and C_j, respectively, and the red circles and orange circles represent the extended representative points of C_i and C_j, respectively

Definition 5 (The set of extended representatives) Denote $R'_{i|j}$ be the set of extended representatives of C_i relative to $R_{i|j}$, which can be defined as follows:

$$R'_{i|j} = \left\{ x_i | x_i \in C_i \backslash R_{i|j}, \exists x_j \in R_{i|j}, x_i \in kNN\left(x_j\right) \right\} \tag{9}$$

Definition 5 is less restrictive than Definition 4, that is, it only requires the nearest neighbor relation in Definition 5. As shown in Fig. 3, the sets of data points marked with red circles and orange circles represent $R'_{i|j}$ and $R'_{j|i}$, respectively.

Next, we define the intercluster distance based on representatives.

Definition 6 (Intercluster distance) The intercluster distance of C_i and C_j is defined as follows:

$$d\left(C_i, C_j\right) = \frac{\sum_{x_i \in R_i} \sum_{x_j \in R_j} D\left(x_i, x_j\right)}{|R_i| \times |R_j|} \tag{10}$$

where $R_i = R_{i|j} \cup R'_{i|j}$, $R_j = R_{j|i} \cup R'_{j|i}$, and $D(x_i, x_j)$ is the Euclidean distance between x_i and x_j.

3.2 Processes

The overall process is divided into three steps: outlier detection and removal, preliminary clustering and subcluster merging. The detailed algorithm process is shown below.

Algorithm 1 NTHC.

Input: Data set X, the number of clusters K.
Output: K clusters.

1: Count the number of kRNNs of data point x_i, denoted as $r(x_i)$.
2: Eliminate the outliers with lower value of $r(x_i)$.
3: Construct a directed graph from the data points using the 1-nearest neighbor.
4: Calculate the stability $s(x_i, x_j)$ of all data point pairs (x_i, x_j).
5: Merge all data point pairs with stable structure, and form a set of subclusters $\{C_1, C_2, \ldots, C_m\}$.
6: Calculate the distance $d(C_i, C_j)$ between every pair of subclusters according to Eqs. (8)-(10).
7: Merge the subcluster pair with the minimum distance d until obtaining K clusters.
8: Merge the outliers to their nearest clusters.

Lines 1 ~ 2 of the NTHC algorithm refer to the step of outlier detection and removal. All data points are sorted in ascending order by $r(x_i)$. The first $N \times th_r$ data points are regarded as outliers and eliminated from the data set X. Here, threshold th_r has a range of [0,1). When th_r is equal to 0, it is equivalent to not removing any data points. And when th_r is close to 1, it is equivalent to considering all data points as outliers. The goal of eliminating outliers is to ensure the accuracy of subsequent clustering. However, the cluster shape will be changed if too many points are taken as outliers. In this paper we set $th_r = 0.2$, which will be discussed in detail in Section 4.5.

Lines 3 ~ 5 of the NTHC algorithm refer to the step of preliminary clustering. According to Eq. (7), the edge (x_i, x_j) is considered as stable structure if $s(x_i, x_j)$ is equal to 1. For all the edges in $G_{1\text{-}NN}$, we retain the edges with stable structure and remove the remaining edges. Finally, the data points connected by edges are merged into the set of subclusters.

We count the number of shared neighbors between two points x_i and x_j according to Eq. (7). If the number of shared neighbors is greater than $th_s \cdot k$, this one-way structure is considered as stable and the points x_i and x_j can be merged into the same cluster. Here, k represents the value of k in k nearest neighbors, and th_s is a threshold in the range (0, 1). In this paper we set a constant value of th_s to avoid the parameter adjustment. The principles of setting th_s are as follows. (1) The threshold th_s should not be too small. If the value of th_s is too small, more data point pairs will be regarded as stable pairs and further merged into the same cluster. And a wrong merge will have a certain impact on the subsequent clustering merging. (2) The threshold th_s can be set as a larger value. If the value of th_s is too large, some data point pairs, which should be merged into the same cluster, will be misjudged as unstable pairs and grouped into different clusters. However, this kind of error can be fixed to a certain extent in the subsequent subcluster merging.

Lines 6 ~ 8 of the NTHC algorithm refer to the step of subcluster merging. We merge successively the cluster pairs with the minimum value of intercluster distance according to Definition 6 until the final clustering result is obtained.

For subcluster merging, it is a crucial problem that how to define the similarity between two subclusters. If there is a problem with an intercluster similarity measure, incorrect merge will affect the final clustering result. The traditional intercluster similarity measures include single linkage, complete linkage and average linkage. Both single linkage and complete linkage select a data point to represent the subcluster, which makes the result sensitive to the representatives. And for average linkage, all points in the cluster participate in the calculation of distance, so the result is sensitive to the shape and density of cluster.

In cciMST (a clustering algorithm based on minimum spanning tree and cluster centers) algorithm [56], the data points at the intersection of clusters are taken as the representatives and the average distance of all the representative pairs is regarded as the intercluster distance. This method has some effect, but its efficiency is lower because it needs to calculate and sort the distance of all data point pairs. Inspired by this method, this paper takes the intercluster nearest neighbors as the representatives and calculates the representative pairwise distance to measure the intercluster similarity according to Eq. (10). The intercluster nearest neighbors are mostly located at the intersection of clusters. In addition, the k-nearest neighbors of all the data points are calculated in the initialization stage. Therefore, we can avoid the sorting operation when calculating the intercluster distance, which can further improve the algorithm efficiency.

3.3 Complexity analysis

The detailed analysis of Algorithm 1 is as follows:

Lines 1 ~ 2: The distance calculation of all data point pairs results in complexity of $O(N^2)$. The complexity of computing the nearest neighbor list can be optimized to $O(N\log N)$ using the algorithms including R* tree [6], K-D tree [12] and ball tree [22], etc. The proposed algorithm is designed upon the neighbor relations involving k nearest neighbors, reverse nearest neighbors, 1-nearest neighbor, and shared nearest neighbors. After constructing the k-nearest neighbor list, other neighbor relations can be obtained from it. The number of k reverse nearest neighbors $r(x_i)$ can be obtained from the k-nearest neighbor list, which requires $O(N)$. Sorting the data points according to the value of $r(x_i)$ needs $O(N\log N)$.

Lines 3 ~ 5: In k-nearest neighbor list, the neighbors of the data point are sorted according to the distances from the data point to its neighbors. Therefore, the information of the first nearest neighbors can be obtained from the k-nearest neighbor list. Next, each edge in $G_{1\text{-}NN}$ is judged whether it is stable according to Eq. (7). Since the bidirectional edge is a stable structure, we need only to count the number of shared nearest neighbors of two endpoints of the one-way edge. Therefore, the time complexity of preliminary clustering is about $O(N)$.

Lines 6 ~ 8: Suppose that the number of clusters to be merged is M. In addition, the representatives of cluster pairs can be obtained from k-nearest neighbor list. Next, we calculate the intercluster distance according to Eq. (10), where the distance between the data point pairs is already calculated in the initialization stage. Since the size of M and the number of representatives are indeterminate values, the approximate time complexity of subcluster merging is $O(N^2)$.

Thus, the overall time complexity of the proposed algorithm is $O(N^2)$.

4 Experiment

4.1 Experiment procedure

In this section, we evaluate the proposed algorithm on five synthetic data sets DS1-DS5 [11, 52, 62, 63] and eight real data sets, including Glass, Zoo, Ecoli, Breast-Cancer, Segment, SCADI, SECOM, and CNAE9. The eight real data sets are taken from the UCI data sets [5], where SCADI, SECOM and CNAE9 are high dimensional data sets. The descriptions of these data sets are shown in Table 2 and Fig. 4. And we employ the Friedman test [19] to assess the statistically significant differences among the sixteen clustering algorithms. Moreover, we further discuss the value of three parameters involved in the proposed algorithm, including k, th_r, and th_s. We test the parameters on the last six data sets in Table 2, including Soybean, Wine, WheatSeeds, DrivFace, Pendigits and SpamBase.

NTHC is compared to the following fifteen clustering algorithms:

1. k-means [34]
2. single linkage (SL) [50]
3. TAHC [61]
4. NC-link [38]
5. KnA [8]
6. GDD [33]
7. DPC [52]
8. spectral clustering (SPC) [11]
9. cciMST [45]
10. Chameleon [39]
11. FINCH [55]
12. SNN [28]

Table 2 Description of the five synthetic data sets and the fourteen real data sets

Data sets	Data Size (N)	Dimensionality (d)	Number of clusters (K)
DS1	512	2	4
DS2	299	2	3
DS3	1000	2	4
DS4	644	2	4
DS5	2000	2	5
Glass	214	9	6
Zoo	101	16	7
Ecoli	336	7	8
Breast-Cancer	683	10	2
Segment	2310	19	7
SCADI	70	205	7
SECOM	1567	590	2
CNAE9	1080	856	9
Soybean	47	35	4
Wine	178	13	3
WheatSeeds	210	7	3
DrivFace	606	6400	3
Pendigits	3498	16	10
SpamBase	4601	57	2

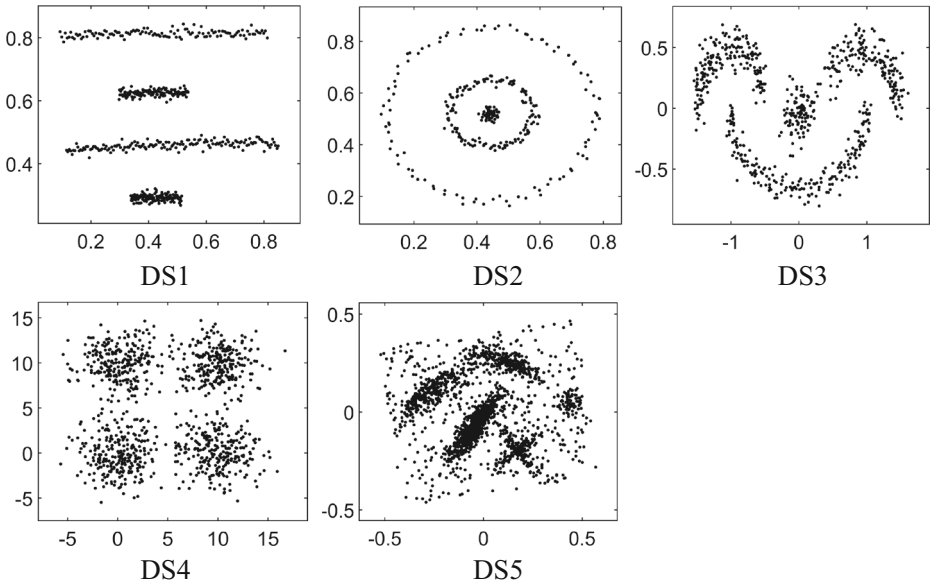


Fig. 4 Five synthetic data sets

13. RNN-DBSCAN (RNN-DB) [9]
14. RSC [57]
15. CHKNN [51]

Among the above compared methods, k-means is one of the partitional clustering algorithms. SL, TAHC, NC-LINK and KnA are hierarchical clustering algorithms. Both GDD and DPC are density-based clustering algorithms. SPC, cciMST and chameleon are graph-based clustering algorithms. The remaining five algorithms including FINCH, SNN, RNN-DB, RSC and CHKNN are neighbor-based algorithms. FINCH and CHKNN are k-nearest-neighbor based clustering algorithms. SNN and RNN-DB are variants of DBSCAN based on respectively shared nearest neighbors and reverse nearest neighbors. RSC is a reciprocal-nearest-neighbors based clustering algorithm. The source code of the proposed algorithm is available online.[1]

For k-means and SPC, we take the best clustering results out of 10 trial runs performed for each data set. As for KnA, the parameter K in its partitional clustering process is set as $0.1\,N$ (N represents the total number of data points). For DPC, the cutoff distance d_c is 2%. The parameters of chameleon are set as follows: $k = 10$ (k is the number of nearest neighbors), minSize = 2.5% and alpha = 2.0. For SNN, the parameters are set as follows: $k = 50$, $Eps = 20$ and $MinPts = 34$. And for RSC, the parameter α is 1.5. As for the three parameters P_1, P_2 and P_3 in CHKNN, we choose the optimal values according to the suggestion of the reference [51].

4.2 Clustering results on synthetic data sets

The data set DS1 is composed of four parallel clusters with different densities. DS2 contains one Gaussian distributed cluster and two round clusters. Both DS1 and DS2 are well separated.

[1] https://github.com/yian158/clustering-analysis.

DS3 is composed of one spherical cluster and three half-moon clusters. DS4 contains four spherical clusters. DS5 is composed of one none-sphere-shaped cluster and four spherical clusters. The clustering results for the five synthetic data sets (DS1-DS5) are shown in Figs. 5, 6, 7, 8, 9.

As a partitional clustering algorithm, k-means can identify the four spherical clusters properly in DS4, but it fails to provide the desired clustering results for the other four data sets containing nonspherical clusters. Some hierarchical clustering algorithms, like SL and KnA, perform well on DS1 and DS2. However, the hierarchical clustering algorithms are sensitive to noise and outliers. Therefore, the hierarchical clustering algorithms (including SL, TAHC and KnA) tend to produce unsatisfactory results when partitioning the data set with noise. For the two density-based clustering algorithms, the performance of GDD is superior to that of DPC. As DPC is sensitive to the cutoff distance, this algorithm may need to be run multiple times. Among the graph-based clustering algorithms, the clustering performance of

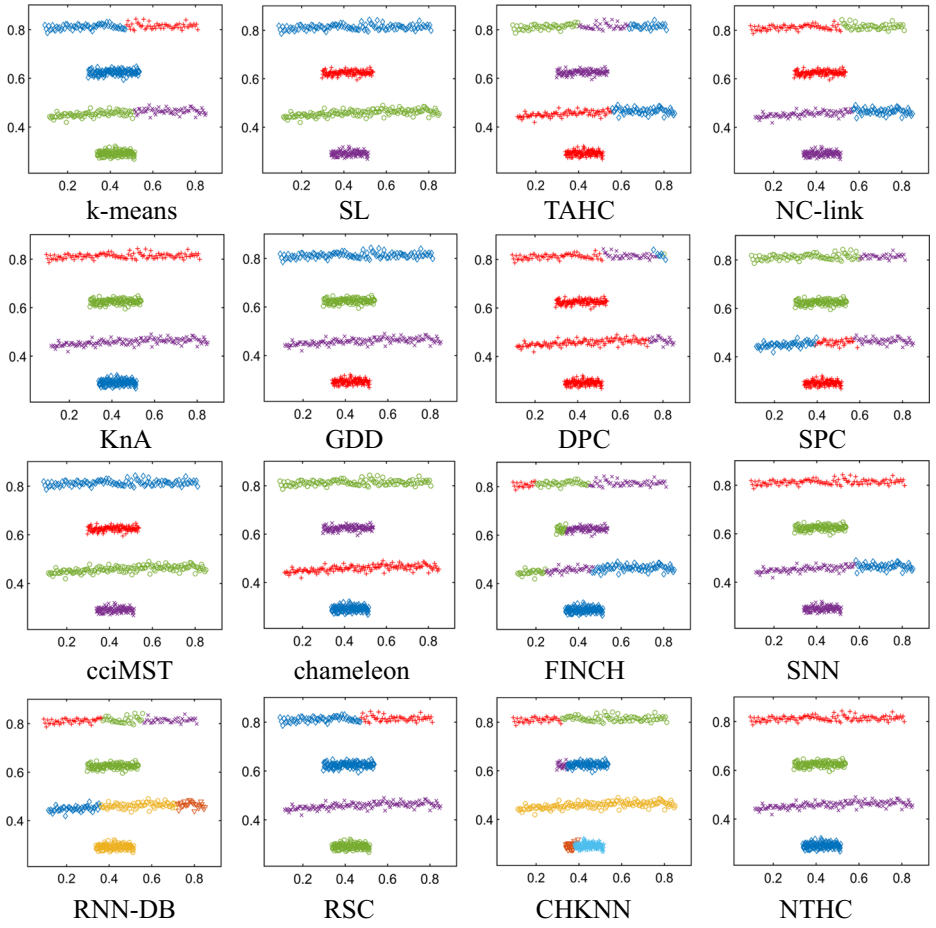


Fig. 5 Clustering result on DS1

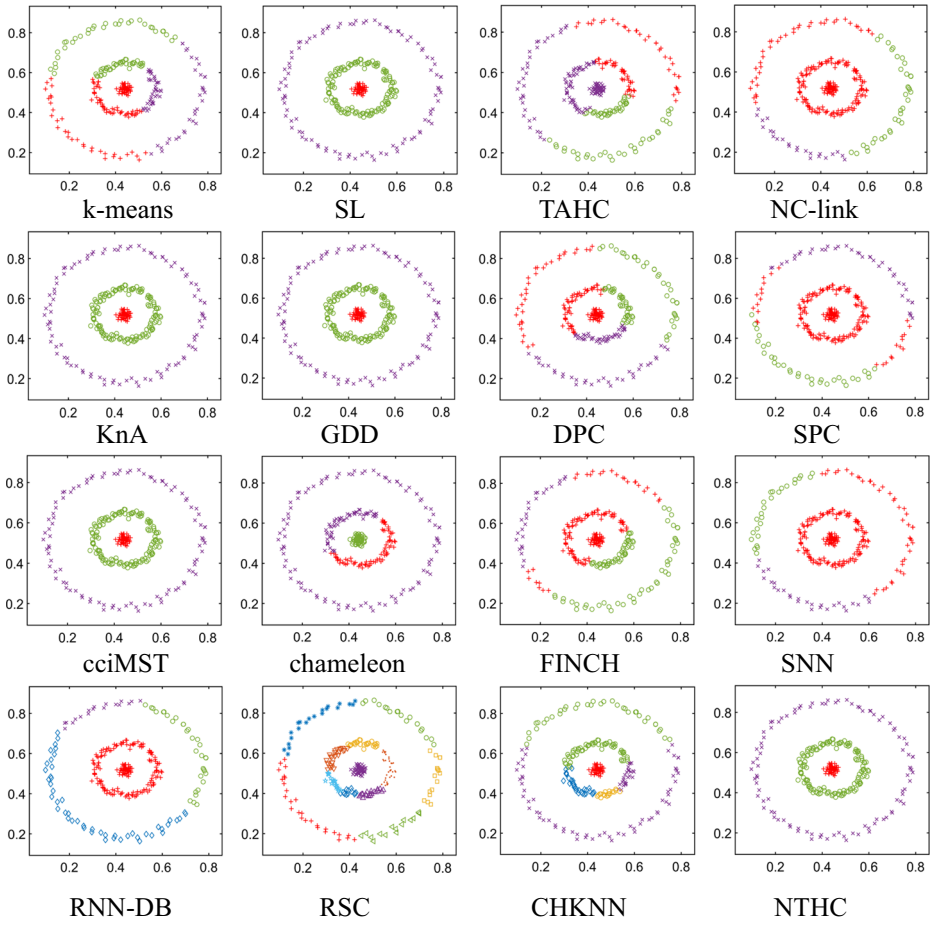


Fig. 6 Clustering result on DS2

cciMST is better than that of SPC and chameleon. However, the time complexity of cciMST is much greater than $O(N^2)$.

Among the six neighbor-based clustering algorithms (FINCH, SNN, RNN-DB, RSC, CHKNN and NTHC), both FINCH and RNN-DB failed to detect the expected clusters for all the data sets. FINCH performed the merging process according to the 1-nearest neighbor graph, which may lead to the wrong results. RNN-DB defined the density of data point based on its reverse nearest neighbor. And incorrect density calculation can affect the results. For the four neighbor-based clustering algorithms (SNN, RNN-DB, RSC, and CHKNN), they do not require the number of clusters in advance, thus the number of clusters obtained from these algorithms are likely not exactly equal to the actual cluster number. SNN requires three input parameters that may influence the results. SNN can find the proper clusters for DS4, but it failed to find the proper clusters on the remaining four data sets. RSC introduced the reciprocal nearest neighbors to merge data points and can automatically determine the number of clusters. But for some data sets such as DS2 and DS5, it cannot determine the desired number of

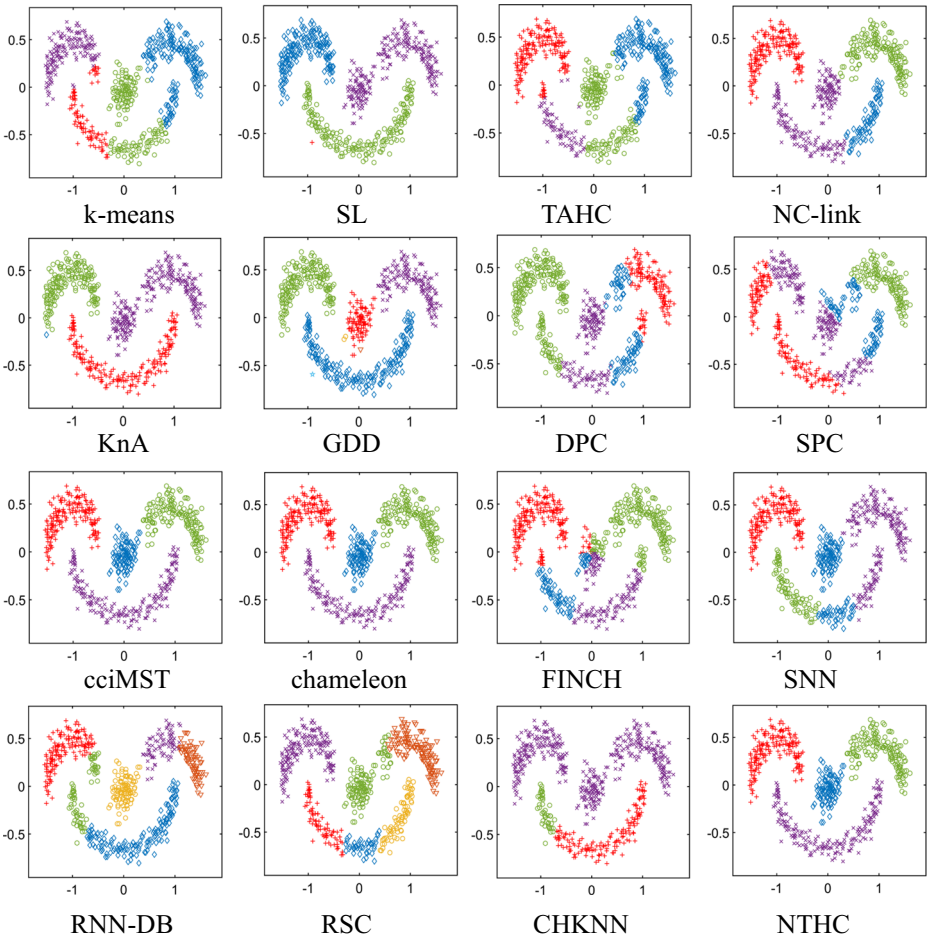


Fig. 7 Clustering result on DS3

clusters and performs relatively poor. CHKNN cannot provide improper clustering results for the five data sets. CHKNN needs three parameters to build a hybrid K-nearest-neighbor graph which will guide the process of identifying the isolated cluster. Hence incorrect parameters may affect the clustering results. NTHC can find the correct clusters for all the five data sets.

4.3 Clustering results on real data sets

To evaluate the goodness of clustering results, we exploit four external clustering validation indices: Accuracy (AC), Precision (PR), Recall (RE) and F1-measure (F1) [18]. Tables 3-10 show the cluster validity indices results for eight real data sets. For GDD, SNN, RNN-DB, RSC and CHKNN, the number of clusters obtained from them is likely not exactly equal to the actual cluster number, therefore, some indices cannot be computed. In this case, we mark with "- "in Tables 3, 4, 5, 6, 7, 8, 9, 10.

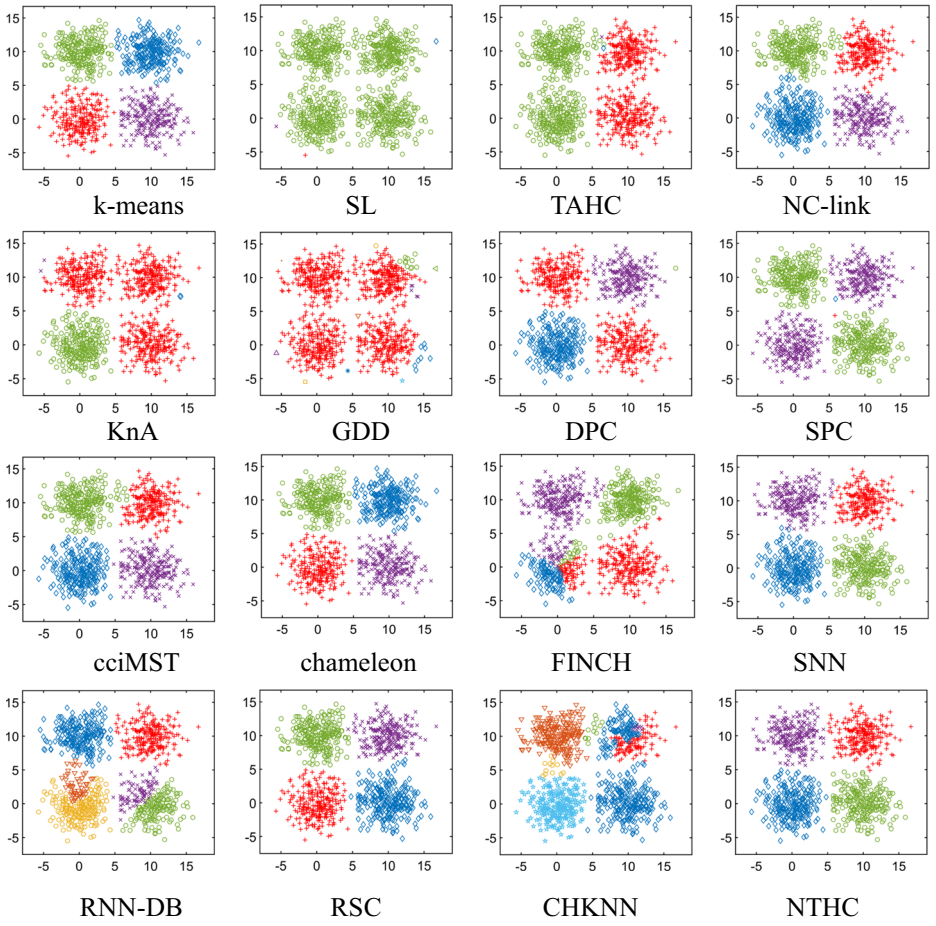


Fig. 8 Clustering result on DS4

For the Glass data set, the performance of NTHC is the best. The clustering performances of NTHC on Zoo and Breast-Cancer data sets are superior to others. For the Ecoli data set, the AC, RE and F1 values of NTHC are higher than other algorithms. For the Segment data set, the F1 value of NTHC is higher than other algorithms. And for three high dimensional data sets (SCADI, SECOM and CNAE9), the performance of NTHC is also superior to other algorithms.

4.4 Friedman test on experimental results

In this section, we test the statistically significant differences among the sixteen clustering algorithms using the Friedman test. Given a clustering algorithms and b experiments (here we set $a = 16$, $b = 13$), the null hypothesis (H_0) of the Friedman test is that all the a algorithms are equivalent. The a algorithms are ranked according to their F1 values in each experiment. Then the Friedman statistic is calculated as follows:

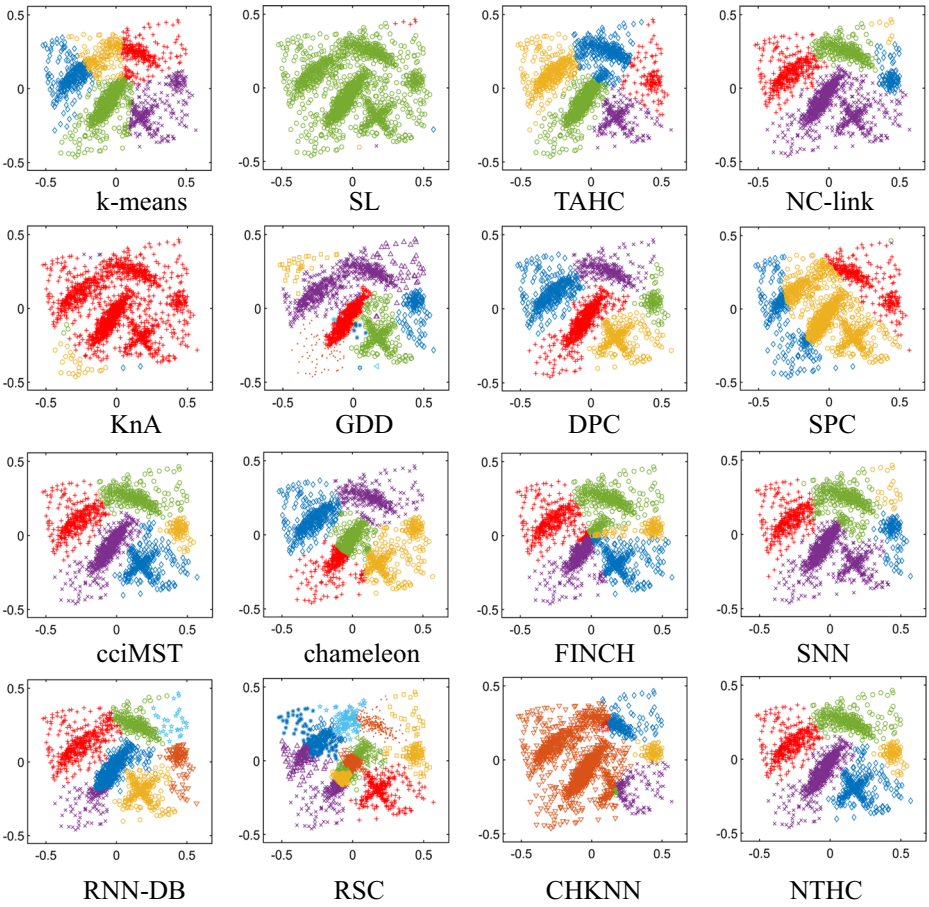


Fig. 9 Clustering result on DS5

$$x_F^2 = \frac{12b}{a(a+1)} \sum_{i=1}^{a} \left(r_i - \frac{a+1}{2} \right)^2 = 34.2437 \tag{11}$$

where r_i is the average rank value of the i-th algorithm. The average rank values of the sixteen clustering algorithms are shown in Table 11.

Next, the statistic F_F is given as:

$$F_F = \frac{(b-1)\chi_F^2}{b(a-1)-\chi_F^2} = 2.5562 \tag{12}$$

The statistic F_F follows an F-distribution with (a-1) and (a-1)(b-1) degrees of freedom. The critical value of F(15,180) is 1.722 when α is equal to 0.05. Therefore, we reject the null hypothesis H_0 and conclude that there are some significant differences among the sixteen algorithms.

Table 3 The performance comparison of sixteen clustering algorithms on Glass

Methods	AC	PR	RE	F1
k-means	0.4219	0.409	0.3959	0.4009
SL	0.3691	**0.5601**	0.2143	0.31
TAHC	0.3879	0.3	0.1929	0.2348
NC-link	0.4813	0.5081	0.3527	0.4164
KnA	0.4626	0.5532	0.3297	0.4132
GDD	0.3505	0.075	–	–
DPC	0.3859	0.3406	0.253	0.2852
SPC	0.4074	0.3578	0.2633	0.294
cciMST	0.3738	0.3788	0.4235	0.3999
chameleon	0.3692	0.3942	0.2451	0.3022
FINCH	0.4626	0.4013	0.3401	0.3682
SNN	0.4346	0.3644	0.327	0.3447
RNN-DB	0.3692	–	0.1832	–
RSC	0.2804	0.1998	–	–
CHKNN	0.4486	–	0.3961	–
NTHC	**0.5**	0.5028	**0.4767**	**0.4894**

Then, we further perform the Nemenyi test to determine which of the algorithms are statistically different. According to the Nemenyi test, the performance of two algorithms can be regarded as significantly different when their corresponding average rank values differ by at least the critical difference *CD*:

$$CD = q_\alpha \sqrt{\frac{a(a+1)}{6b}} = 6.4052 \tag{13}$$

where q_α is the critical value. For $\alpha = 0.05$ and $a = 16$, $q_\alpha = 3.426$.

Figure 10 shows the results of the above tests. The vertical axis denotes the sixteen algorithms and the horizontal axis denotes the average rank values of the algorithms. Each algorithm corresponds to a horizontal line, where the line length represents the critical difference *CD*, and the black dot on the line corresponds to the average rank value of each

Table 4 The performance comparison of sixteen clustering algorithms on Zoo

Methods	AC	PR	RE	F1
k-means	0.7168	0.582	0.5984	0.5887
SL	0.8713	0.7219	0.7286	0.7252
TAHC	0.6238	0.4641	0.311	0.3724
NC-link	0.7822	0.5814	0.5683	0.5748
KnA	0.8218	0.5726	0.6969	0.6286
GDD	0.2673	0.1228	–	–
DPC	0.3802	0.3725	0.2904	0.321
SPC	0.8416	0.6598	0.718	0.6866
cciMST	0.8218	0.5875	0.6397	0.6125
chameleon	0.5941	0.5404	0.4594	0.4966
FINCH	0.8614	0.74	0.7934	0.7658
SNN	0.7327	–	0.4286	–
RNN-DB	0.7822	–	0.5432	–
RSC	0.7624	0.6236	–	–
CHKNN	0.6040	–	0.2857	–
NTHC	**0.9505**	**0.9187**	**0.9**	**0.9093**

Table 5 The performance comparison of sixteen clustering algorithms on Ecoli

Methods	AC	PR	RE	F1
k-means	0.5402	0.5203	0.5087	0.5142
SL	0.4494	**0.5968**	0.2989	0.3984
TAHC	0.5952	0.1501	0.2258	0.1803
NC-link	0.6280	0.4512	0.4188	0.4344
KnA	0.6696	0.4423	0.5187	0.4775
GDD	0.4405	0.4944	–	–
DPC	0.3583	0.171	0.1466	0.1556
SPC	0.4875	0.5569	0.338	0.4197
cciMST	0.5119	0.4664	0.4389	0.4522
chameleon	0.478	0.4388	0.3397	0.3829
FINCH	0.6607	0.5044	0.5178	0.511
SNN	0.756	0.5673	0.5153	0.5401
RNN-DB	0.628	–	0.2354	–
RSC	0.3720	0.2084	–	–
CHKNN	0.7262	–	0.3255	–
NTHC	**0.8006**	0.5687	**0.5251**	**0.546**

algorithm. As can be seen from Fig.10, NTHC has the best average ranking, and the performance of NTHC is clearly superior to that of TAHC, DPC and SPC.

4.5 Parameter setting

In this section, we will discuss the values of three parameters involved in NTHC, including k, th_r, and th_s. Figure 11 shows the values of F1 on six data sets with the variation of different parameters. When varying one parameter value, the other two parameters are fixed. The fixed values of k, th_r, and th_s are 10, 0.2 and 0.65, respectively.

The parameter k is the number of nearest neighbors. Figure 11(a) shows the values of F1 on six data sets with different k values, in which the k value ranges from 5 to 70. We can see from Fig. 11(a) that, for Wine, WheatSeeds, DrivFace, and SpamBase, the value of F1 varies

Table 6 The performance comparison of sixteen clustering algorithms on Breast-Cancer

Methods	AC	PR	RE	F1
k-means	0.9612	0.961	0.9533	0.9572
SL	0.6515	0.8255	0.502	0.6244
TAHC	0.9356	0.9279	0.9311	0.9295
NC-link	0.9458	0.9424	0.938	0.9402
KnA	0.6515	0.8255	0.5021	0.6244
GDD	0.347	0.0054	–	–
DPC	0.6357	0.6336	0.4942	0.5413
SPC	0.6515	0.8255	0.502	0.6244
cciMST	0.6032	0.5415	0.5354	0.5385
chameleon	0.675	0.8333	0.5356	0.6521
FINCH	0.9078	0.8935	0.9155	0.9044
SNN	0.9546	0.9582	0.9419	0.95
RNN-DB	0.5637	0.2057	–	–
RSC	0.7116	0.6232	–	–
CHKNN	0.5051	0.1475	–	–
NTHC	**0.9722**	**0.9637**	**0.9776**	**0.9706**

Table 7 The performance comparison of sixteen clustering algorithms on Segment

Methods	AC	PR	RE	F1
k-means	0.5787	0.5958	0.5787	0.5864
SL	0.1459	0.449	0.1459	0.2202
TAHC	0.1433	0.1633	0.1433	0.1527
NC-link	0.4346	**0.7431**	0.4346	0.5485
KnA	0.4320	0.6002	0.432	0.5024
GDD	0.3758	0.0166	–	–
DPC	0.188	0.5168	0.188	0.267
SPC	0.1458	0.1634	0.1458	0.1541
cciMST	0.6359	0.6343	**0.6359**	0.6351
chameleon	0.3773	0.5671	0.3377	0.4233
FINCH	0.5623	0.5959	0.5623	0.5787
SNN	0.5948	0.6317	0.5948	0.6127
RNN-DB	**0.6584**	0.5754	–	–
RSC	0.5048	0.3806	–	–
CHKNN	0.1567	–	0.1567	–
NTHC	0.581	0.7121	0.581	**0.6399**

smoothly from 15 to 70. For Pendigits, the value of F1 fluctuates slightly within the range of 15–70. Note that because Soybean contains only 47 data points, the algorithm terminates when k increases to about 41. In addition, the value of F1 drops considerably from about 28 due to the small amount of data points. And it can be seen from Fig. 11(a) that, for all the six data sets, the value of F1 achieves the maximum around 10. Thus, the parameter k is suggested to be set to 10.

The parameter th_r is a proportion parameter to detect the outliers. We can see from Fig. 11(b) that the value of F1 fluctuates relatively greatly with the change of th_r. The clustering performance is sensitive to the parameter th_r. Our algorithm can achieve good performance on the six data sets when th_r is set to 0.2. Hence, the th_r value is set to 0.2 in this paper.

Table 8 The performance comparison of sixteen clustering algorithms on SCADI

Methods	AC	PR	RE	F1
k-means	0.63	0.4863	0.514	0.4986
SL	0.657	0.4397	0.4083	0.4234
TAHC	0.5857	0.2498	0.28	0.264
NC-link	0.7143	0.557	0.6179	0.5859
KnA	0.5571	0.4886	0.4912	0.4899
GDD	0.1	0.1	–	–
DPC	0.34	0.2663	0.2223	0.2372
SPC	0.657	0.4599	0.448	0.4538
cciMST	0.5714	0.4652	0.4408	0.4527
chameleon	0.5714	0.4851	0.2463	0.3267
FINCH	0.6857	0.5238	0.4926	0.5077
SNN	0.5714	–	0.2808	–
RNN-DB	0.6286	–	0.2808	–
RSC	0.4429	0.2865	–	–
CHKNN	0.6286	–	0.2808	–
NTHC	**0.8**	**0.5891**	**0.686**	**0.6339**

Table 9 The performance comparison of sixteen clustering algorithms on SECOM

Methods	AC	PR	RE	F1
k-means	0.6227	0.5229	**0.5541**	0.5372
SL	**0.933**	0.4668	0.4997	0.4827
TAHC	**0.933**	0.4668	0.4997	0.4827
NC-link	**0.933**	0.4668	0.4997	0.4827
KnA	0.9324	0.5921	0.5038	0.5444
GDD	0.0013	0.0013	–	–
DPC	0.926	0.5932	0.5227	0.5557
SPC	**0.933**	0.4668	0.4997	0.4827
cciMST	0.9311	0.4667	0.4986,	0.4822
chameleon	0.9196	0.5739	0.5282	0.5501
FINCH	0.9221	0.5778	0.5251	0.5502
SNN	**0.933**	0.4668	0.4997	0.4827
RNN-DB	0.9221	0.5778	0.5251	0.5502
RSC	0.6439	0.5040	0.5145	0.5092
CHKNN	0.9221	0.5778	0.5251	0.5502
NTHC	**0.933**	**0.6337**	0.5041	**0.5615**

The parameter th_s is a threshold to measure the similarity of two data points. As shown in Fig. 11(c), the performance of NTHC on six data sets tends to be relatively insensitive to the value of the parameter th_s. According to Fig. 11(c), our algorithm achieves good clustering results on the six data sets when th_s is set to 0.65. Therefore, the th_s value is suggested to be set to 0.65 in this paper.

5 Conclusion

In this paper, a novel neighborhood-based hierarchical clustering algorithm NTHC, is presented. It utilizes the reverse nearest neighbor to detect and remove the outliers in the data set. Guided by the 1-nearest neighbor graph, the data points with stable connection are merged. We

Table 10 The performance comparison of sixteen clustering algorithms on CNAE9

Methods	AC	PR	RE	F1
k-means	0.354	0.5312	0.354	0.4235
SL	0.1167	0.6791	0.1167	0.1991
TAHC	0.113	0.2347	0.113	0.1525
NC-link	0.1157	0.568	0.1157	0.1923
KnA	0.1176	0.6791	0.1176	0.2005
GDD	0.0194	0.0086	–	–
DPC	0.125	0.446	0.125	0.1964
SPC	0.1194	0.5029	0.1194	0.1929
cciMST	0.1157	0.4939	0.1157	0.1875
chameleon	0.3039	0.4948	0.3009	0.3742
FINCH	0.3259	**0.6931**	0.3259	0.4434
SNN	0.275	0.5683	0.275	0.3706
RNN-DB	0.1111	–	0.1111	–
RSC	0.4052	0.3481	–	–
CHKNN	0.3093	–	0.3093	–
NTHC	**0.5287**	0.6205	**0.5287**	**0.571**

Table 11 The average rank values of the sixteen clustering algorithms

Method	Average rank value
k-means	7.615385
SL	10.5
TAHC	11.42308
NC-link	9.076923
KnA	9.269231
GDD	8.166667
DPC	11.69231
SPC	10.92308
cciMST	7.807692
chameleon	8.384615
FINCH	8.461538
SNN	7.818182
RNN-DB	6.2495
RSC	8
CHKNN	6.3881
NTHC	4.193223

propose the intercluster distance measure based on the linked representatives and extended representatives. The main innovations of this paper include the organic combination of various kinds of neighbor relations. And we propose the improved hierarchical clustering algorithm based on this idea.

We evaluate our algorithm on five synthetic data sets and eight real data sets. The performance of our algorithm is compared with fifteen clustering algorithms. NTHC can discover the correct clusters for all the five synthetic data sets. And the performances of NTHC on most data sets are superior to the compared methods. Furthermore, we test the statistically significant differences among the sixteen clustering algorithms using the Friedman test. The average rank value of NTHC is 4.19. And NTHC has the best average ranking among all sixteen clustering algorithms.

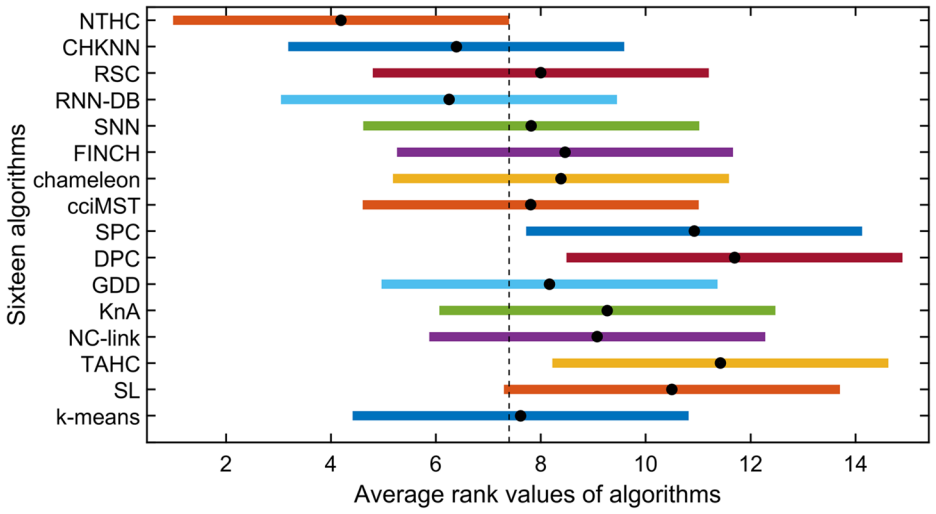


Fig. 10 Graphical presentation of results

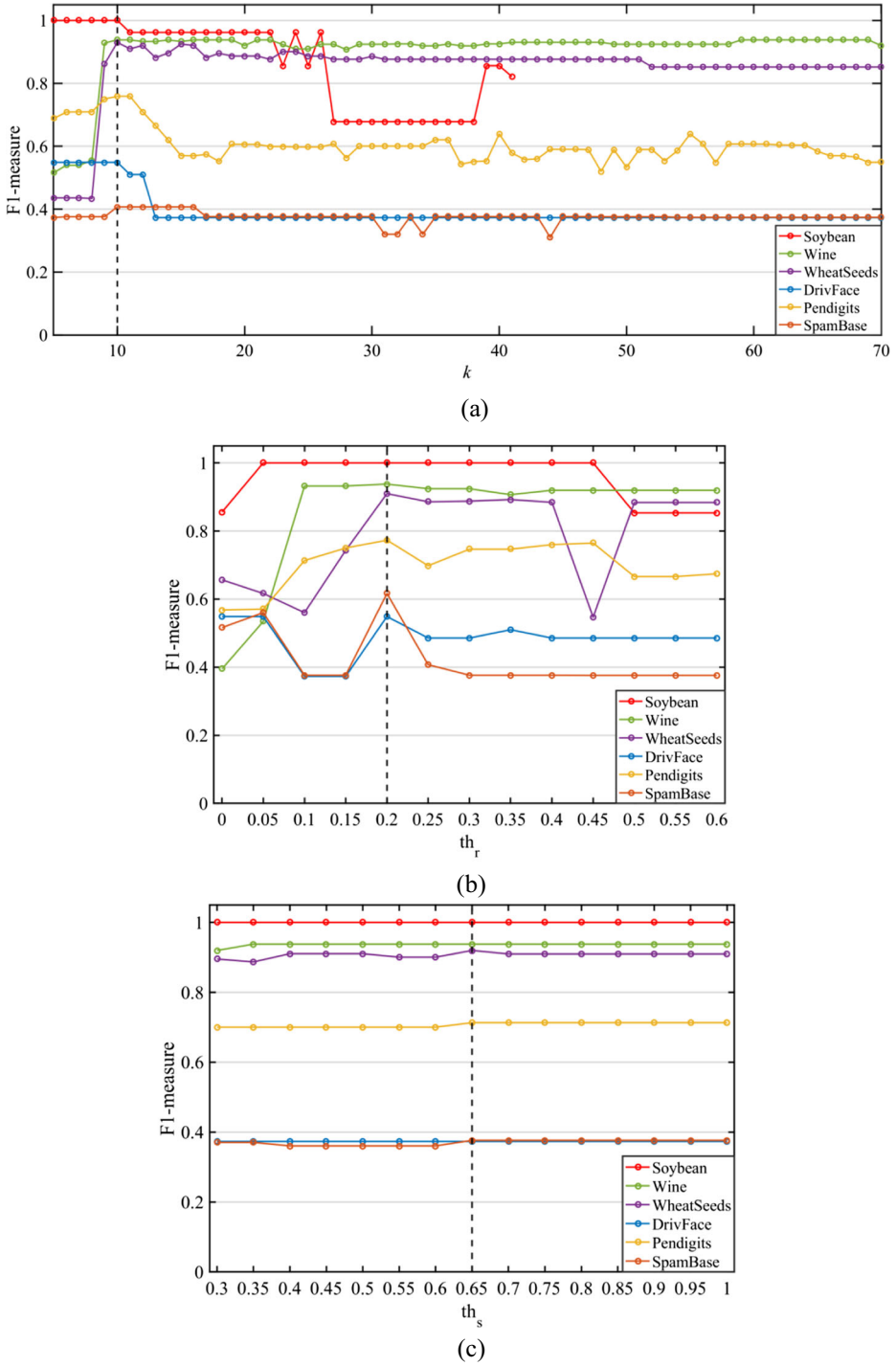


Fig. 11 The effect of the values of three parameters (including k, th_r, and th_s) on clustering results. (a) k vs F1, (b) th_r vs F1, (c) th_s vs F1

However, our algorithm requires the number of clusters in advance. In the future work, we plan to further optimize our algorithm to automatically determine the number of clusters. In addition, we intend to apply the neighbor relation to other different types of clustering algorithms.

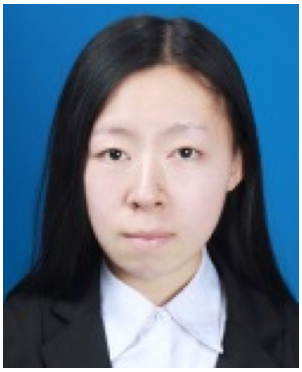
Acknowledgments This work is partially supported by the National Natural Science Foundation of China (Grant no. 61373004, 61501297).

References

1. Abbas M, El-Zoghabi A, Shoukry A (2021) DenMune: density peak based clustering using mutual nearest neighbors. Pattern Recogn 109:107589
2. Ali A, Zhu Y, Chen Q, Yu J, Cai H (2019) Leveraging spatio-temporal patterns for predicting citywide traffic crowd flows using deep hybrid neural networks. In Proceedings of the 2019 IEEE 25th International Conference on Parallel and Distributed Systems, pp. 125–132.
3. Ali A, Zhu Y, Zakarya M (2021) A data aggregation based approach to exploit dynamic spatio-temporal correlations for citywide crowd flows prediction in fog computing. Multimed Tools Appl 2:1–33
4. Ankerst M, Breunig MM, Kriegel H-P, Sander J (1999) OPTICS: ordering points to identify the clustering structure. ACM SIGMOD Rec 28(2):49–60
5. Asuncion A, Newman D (2007) UCI machine learning repository
6. Beckmann N, Kriegel H-P, Schneider R, Seeger B (1990) The R*-tree: an efficient and robust access method for points and rectangles, in Proceedings of the 1990 ACM SIGMOD international conference on Management of data, 322–331
7. Blömer J, Lammersen C, Schmidt M, Sohler C (2016) Theoretical analysis of the k-means algorithm–a survey. Algorithm Eng 9220:81–116
8. Bouguettaya A, Yu Q, Liu X, Zhou X, Song A (2015) Efficient agglomerative hierarchical clustering. Expert Syst Appl 42(5):2785–2797
9. Bryant A, Cios K (2018) RNN-DBSCAN: a density-based clustering algorithm using reverse nearest neighbor density estimates. IEEE Trans Knowl Data Eng 30(6):1109–1121
10. Cassisi C, Ferro A, Giugno R, Pigola G, Pulvirenti A (2013) Enhancing density-based clustering: parameter reduction and outlier detection. Inf Syst 38(3):317–330
11. Chang H, Yeung D-Y (2008) Robust path-based spectral clustering. Pattern Recogn 41(1):191–203
12. Chen Y, Zhou L, Tang Y, Singh JP, Bouguila N, Wang C, Wang H, Du J (2019) Fast neighbor search by using revised kd tree. Inf Sci 472:145–162
13. Chowdhary CL, Acharjya D (2016) A hybrid scheme for breast cancer detection using intuitionistic fuzzy rough set technique. Intl J Healthcare Inf Syst Inf (IJHISI) 11(2):38–61
14. Chowdhary CL, Acharjya D (2017) Clustering algorithm in possibilistic exponential fuzzy c-mean segmenting medical images. J Biomim, Biomater Biomed Eng 30:12–23
15. Chowdhary CL, Acharjya D (2017) Segmentation of mammograms using a novel intuitionistic possibilistic fuzzy c-mean clustering algorithm, Nature Inspired Computing, vol. 652, pp. 75–82: Springer
16. Chowdhary CL, Acharjya D (2020) Segmentation and feature extraction in medical imaging: a systematic review. Procedia Computer Science 167:26–36
17. Chowdhary CL, Sai GVK, Acharjya D (2016) Decrease in false assumption for detection using digital mammography, Computational Intelligence in Data Mining—Volume 2, pp. 325–333: Springer
18. Dahal S (2015) Effect of different distance measures in result of cluster analysis
19. Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7(1):1–30
20. Ding S, Xu X, Fan S, Xue Y (2018) Locally adaptive multiple kernel k-means algorithm based on shared nearest neighbors. Soft Comput 22(14):4573–4583
21. Ding S, Cong L, Hu Q, Jia H, Shi Z (2019) A multiway p-spectral clustering algorithm. Knowl-Based Syst 164:371–377
22. Dolatshah M, Hadian A, Minaei-Bidgoli B (2015) Ball*-tree: Efficient spatial indexing for constrained nearest-neighbor search in metric spaces, Computer Science, arXiv preprint arXiv:1511.00628
23. Dong S (2021) Multi class SVM algorithm with active learning for network traffic classification. Expert Syst Appl 176:114885

24. Dong S, Zhou D, Ding W, Gong J (2013) Flow cluster algorithm based on improved K-means method. IETE J Res 59(4):326–333
25. Dong S, Zhang X, Li Y (2018) Microblog sentiment analysis method based on spectral clustering. J Inf Process Syst 14(3):727–739
26. Dong S, Wang P, Abbas K (2021) A survey on deep learning and its applications. Comput Sci Rev 40: 100379
27. Du M, Ding S, Jia H (2016) Study on density peaks clustering based on k-nearest neighbors and principal component analysis. Knowl-Based Syst 99:135–145
28. Ertöz L, Steinbach M, Kumar V (2003) Finding clusters of different sizes, shapes, and densities in noisy, high dimensional data, in Proceedings of the 2003 SIAM international conference on data mining, 47–58
29. Ester M, Kriegel H-P, Sander J, Xu X (1996) A density-based algorithm for discovering clusters in large spatial databases with noise, AAAI press
30. Fan J-c, Jia P-l, Ge L (2019) M k-NN G-DPC: density peaks clustering based on improved mutual K-nearest-neighbor graph. Int J Mach Learn Cybern 11(6):1–17
31. Franti P, Virmajoki O, Hautamaki V (2006) Fast agglomerative clustering using a k-nearest neighbor graph. IEEE Trans Pattern Anal Mach Intell 28(11):1875–1881
32. Gowda KC, Krishna G (1978) Agglomerative clustering using the concept of mutual nearest neighbourhood. Pattern Recogn 10(2):105–112
33. Güngör E, Özmen A (2017) Distance and density based clustering algorithm using Gaussian kernel. Expert Syst Appl 69:10–20
34. Hartigan JA, Wong MA (1979) Algorithm AS 136: a k-means clustering algorithm. J R Stat Soc: Ser C: Appl Stat 28(1):100–108
35. İnkaya T (2015) A parameter-free similarity graph for spectral clustering. Expert Syst Appl 42(24):9489–9498
36. İnkaya T, Kayalıgil S, Özdemirel NE (2015) An adaptive neighbourhood construction algorithm based on density and connectivity. Pattern Recogn Lett 52:17–24
37. Jarvis RA, Patrick EA (1973) Clustering using a similarity measure based on shared near neighbors. IEEE Trans Comput 100(11):1025–1034
38. Jeon Y, Yoo J, Lee J, Yoon S (2017) Nc-link: a new linkage method for efficient hierarchical clustering of large-scale data. IEEE Access 5:5594–5608
39. Karypis G, Han E-H, Kumar V (1999) Chameleon: hierarchical clustering using dynamic modeling. Computer 32(8):68–75
40. Lai JZ, Huang T-J (2011) An agglomerative clustering algorithm using a dynamic k-nearest-neighbor list. Inf Sci 181(9):1722–1734
41. Li H, Liu X, Li T, Gan R (2020) A novel density-based clustering algorithm using nearest neighbor graph. Pattern Recogn 102:1–13
42. Li J, Huang G, Zhou Y (2020) A sentiment classification approach of sentences clustering in webcast barrages. J Inf Process Syst 16(3):718–732
43. Li X, Lv J, Yi Z (2018) Outlier detection using structural scores in a high-dimensional space. IEEE Trans Cyberne 50(5):2302–2310
44. Liu R, Wang H, Yu X (2018) Shared-nearest-neighbor-based clustering by fast search and find of density peaks. Inf Sci 450:200–226
45. Lv X, Ma Y, He X, Huang H, Yang J (2018) CciMST: a clustering algorithm based on minimum spanning tree and cluster centers. Math Probl Eng 2018:1–14
46. Lv Y, Ma T, Tang M, Cao J, Tian Y, Al-Dhelaan A, Al-Rodhaan M (2016) An efficient and scalable density-based clustering algorithm for datasets with complex structures. Neurocomputing 171:9–22
47. Lv Y, Liu M, Xiang Y (2020) Fast Searching Density Peak Clustering Algorithm Based on Shared Nearest Neighbor and Adaptive Clustering Center. *Symmetry* 12(12):2014
48. Ma Y, Lin H, Wang Y, Huang H, He X (2021) A multi-stage hierarchical clustering algorithm based on centroid of tree and cut edge constraint. Inf Sci 557:194–219
49. Maier M, Hein M, Von Luxburg U (2009) Optimal construction of k-nearest-neighbor graphs for identifying noisy clusters. Theor Comput Sci 410(19):1749–1764
50. Murtagh F, Contreras P (2017) Algorithms for hierarchical clustering: an overview, II, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, vol. 7, no. 6, pp. e1219
51. Qin Y, Yu ZL, Wang C-D, Gu Z, Li Y (2018) A novel clustering method based on hybrid K-nearest-neighbor graph. Pattern Recogn 74(1):1–14
52. Rodriguez A, Laio A (2014) Clustering by fast search and find of density peaks. Science 344(6191):1492–1496
53. Ros F, Guillaume S (2019) Munec: a mutual neighbor-based clustering algorithm. Inf Sci 486:148–170

54. Ros F, Guillaume S, El Hajji M, Riad R (2020) KdMutual: a novel clustering algorithm combining mutual neighboring and hierarchical approaches using a new selection criterion. Knowl-Based Syst 204:106220
55. Sarfraz S, Sharma V, Stiefelhagen R (2019) Efficient parameter-free clustering using first neighbor relations, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 8934–8943
56. Vadapalli S, Valluri SR, Karlapalem K (2006) A simple yet effective data clustering algorithm, in Sixth International Conference on Data Mining (ICDM'06), 1108–1112
57. Xie W-B, Lee Y-L, Wang C, Chen D-B, Zhou T (2020) Hierarchical clustering supported by reciprocal nearest neighbors. Inf Sci 527:279–292
58. Yang J, Ma Y, Zhang X, Li S, Zhang Y (2017) An initialization method based on hybrid distance for k-means algorithm. Neural Comput 29(11):3094–3117
59. Yaohui L, Zhengming M, Fang Y (2017) Adaptive density peak clustering based on K-nearest neighbors with aggregating strategy. Knowl-Based Syst 133:208–220
60. Ye H, Lv H, Sun Q (2016) An improved clustering algorithm based on density and shared nearest neighbor, in 2016 IEEE Information Technology, Networking, Electronic and Automation Control Conference, pp. 37–40
61. Yu M, Hillebrand A, Tewarie P, Meier J, van Dijk B, Van Mieghem P, Stam CJ (2015) Hierarchical clustering in minimum spanning trees. Chaos: Interdisc J Nonlinear ScI 25(2):023107
62. Zhong C, Miao D, Wang R (2010) A graph-theoretical clustering method based on two rounds of minimum spanning trees. Pattern Recogn 43(3):752–766
63. Zhong C, Miao D, Fränti P (2011) Minimum spanning tree based split-and-merge: a hierarchical clustering method. Inf Sci 181(16):3397–3410
64. Zhou Q (2018) Traffic flow data analysis and mining method based on clustering recognition algorithm. Adv Transport Stud 3:101–108



Yan Wang is a Master's student at College of Information, Mechanical and Electrical Engineering, Shanghai Normal University, China. Her research interest includes pattern recognition and image processing.

Yan Ma is currently a professor with the Department of Computer Science, Shanghai Normal University, China. She received the Ph.D. degree from Shanghai Jiangtong University. Her research interests include data mining, image segmentation, and action recognition.

Hui Huang is currently an associate professor with the Department of Computer Science, Shanghai Normal University, China. She received the Ph.D. degree from University of Rennes I, Rennes, France. Her research interests include image processing, machine learning and pattern recognition.