Computers in Biology and Medicine 150 (2022) 106184

Contents lists available at ScienceDirect

Computers in Biology and Medicine

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

A mutual neighbor-based clustering method and its medical applications

Jun Chen [a], Xinzhong Zhu [b], Huawen Liu [c,*]

[a] Zhejiang Industry Polytechnic College, Shaoxing 312000, PR China
[b] Zhejiang Normal University, Jinhua 321000, PR China
[c] Shaoxing University, Shaoxing 312000, PR China

ARTICLE INFO

Keywords:
Clustering analysis
Mutual neighbor
Manifold data
K-means
Machine learning
Medicine disease

ABSTRACT

Clustering analysis has been widely used in various real-world applications. Due to the simplicity of K-means, it has become the most popular clustering analysis technique in reality. Unfortunately, the performance of K-means heavily relies on initial centers, which should be specified in prior. Besides, it cannot effectively identify manifold clusters. In this paper, we propose a novel clustering algorithm based on representative data objects derived from mutual neighbors to identify different shaped clusters. Specifically, it first obtains mutual neighbors to estimate the density for each data object, and then identifies representative objects with high densities to represent the whole data. Moreover, a concept of path distance, deriving from a minimum spanning tree, is introduced to measure the similarities of representative objects for manifold structures. Finally, an improved K-means with initial centers and path-based distances is proposed to group the representative objects into clusters. For non-representative objects, their cluster labels are determined by neighborhood information. To verify the effectiveness of the proposed method, we conducted comparison experiments on synthetic data and further applied it to medical scenarios. The results show that our clustering method can effectively identify arbitrary-shaped clusters and disease types in comparing to the state-of-the-art clustering ones.

1. Introduction

With the great advancement of information technology, data collected from real-world applications is getting larger and larger, and the distributions of data are becoming more and more complicated [1]. This poses great challenges to conventional data mining and analysis algorithms, which often assume the scale of data is not so massive [2]. Thus, it is necessary to amend the conventional mining algorithms to acclimate such big data, or develop new mining algorithms accordingly . Note that it is nontrivial to the former. Thus, most endeavors have been attempted to the latter case.

As a typical data analysis technique, clustering has been widely and successfully used in various scenarios, range from biological data analysis, medical diagnosis, text categorization, data summarizing, social network analysis, to image and video processing [3,4]. Until now, a great number of clustering methods, including distance-based, density-based, subspace-based, model-based and hierarchy clustering, have been developed [5,6]. Among them, K-means is undoubtedly the most popular one, due to the fact that it is conceptually simple and versatile [7].

Generally, the classic K-means algorithm [8] works in an intuitive manner. It first randomly chooses k data objects as initial centers of clusters, and then arranges other data objects to one of the clustering centers in the light of nearest distances. Afterwards, the clustering centers are updated according to the mean of their own members. This clustering process is iterated, until the centers are not changed. It is noticeable that nonetheless the popularity of K-means, its final performance heavily relies on the selection of initial centers. Moreover, it cannot effectively identify non-spherical clusters, because of the global property of Euclidean distance [9]. To remedy these problems, several variants, such as K-medians [10], K-means++ [11] and K*-means [12], have been proposed during the past decades. The K-medoids clustering algorithm [13] represents cluster centers by actual data objects within clusters. Recently, Huang et al. [14] adopted deep learning techniques to extract hidden representations of data objects and hierarchically employed K-means with deep structures. However, the time costs are expensive and most importantly, the interpretation of deep learning ones is poor, making them unsuitable for medical applications especially.

The intrinsic properties of data objects, e.g., distributions, densities and structural information, have also been exploited to excavate clusters with complex structures in literature. Representative examples include DPC (Density peak clustering) [15] and densityCut [16]. Both of them assume that cluster centers often have higher densities than their neighbors and far from each other. APC (Affinity propagation

* Corresponding author.
E-mail addresses: chenjun20080057@zjipc.edu.cn (J. Chen), zxz@zjnu.edu.cn (X. Zhu), liu@usx.edu.cn (H. Liu).

https://doi.org/10.1016/j.compbiomed.2022.106184
Received 28 August 2022; Received in revised form 23 September 2022; Accepted 8 October 2022
Available online 12 October 2022

clustering) [17] probes structural information of data points by a message-passing mechanism, where the similarities of points are iteratively propagated until cluster centers gradually emerge. However, they only take global densities or structural information into consideration, without involving local information [18].

In this paper, we propose a novel clustering algorithm coupling K-means with mutual neighbors, dubbed MN-Kmeans. Specifically, it exploits mutual neighbors to derive representative data objects. Based on the representative objects, a minimum spanning tree is generated to estimate the similarities of objects thorough a notion of path-based distance. This kind of path-based distance is quite suitable for evaluating dissimilarities between data objects on different shaped clusters, including both spherical and manifold clusters, so that the proposed method can efficiently identify arbitrary-shaped clusters. Finally, an improved K-means with initial centers and path-based distances is exploited to group the representative objects into clusters. To demonstrate the effectiveness of MN-Kmeans, we extensively conducted experiments on benchmark datasets with arbitrary-shaped clusters by comparing MN-Kmeans to the state-of-the-art clustering algorithms. The experimental results show that MN-Kmeans has outstanding performance in identifying arbitrary-shaped clusters.

In essence, the main contributions of this paper are briefly summarized as follows:

- We exploit representative objects, derived from mutual neighbors, to precisely represent data distributions and identify cluster centers.
- A notion of path-based distance in a minimum spanning tree is adopted to estimate the similarities of data objects. This distance has local property and is suitable to manifold data.
- An improved K-means algorithm is proposed based on representative objects and path-based distances. The advantage is that it can identify arbitrary-shaped clusters.

The rest of this paper is organized as follows. Section 2 briefly reviews the related work about K-means clustering analysis. In Section 3, we present the framework of the proposed clustering method. The experimental results on synthetic data are reported in Section 4, and the applications to medical data are discussed in Section 5, followed by the conclusion of the paper in Section 6.

2. Related work

As mentioned above, clustering techniques can be broadly grouped into hierarchy-based, partition-based, density-based, graph-based and subspace-based ones [3,5], where the partition-based clustering seems to be more popular, because of its versatility. It initially designates several clusters and then iteratively reallocates data objects to these clusters. The partition-based clustering methods can be further classified into K-means, K-medoids and probabilistic ones [19]. Here we concentrate on K-means and its extensions.

K-means [8] is by far the most popular clustering algorithm and widely used in real-world applications nowadays. It first randomly picks k data objects as initial centers, and then allocates the rest objects to one of these centers according to appropriate distance measurements. Subsequently, the cluster centers are updated as the mean (or weighted average) of their affiliated members. This process iterates until the centers have not been changed. Since the performance of K-means heavily reckons on the initial centers, K-mean++ [11] improves the quality of initial centers by taking the objects far from them as new cluster centers.

Another concern for K-means, which has also been received considerable attraction, is its computational efficiency. For example, multi-stage K-means (MKM) [20] filters data objects via a coarse-to-fine search strategy and speeds up the allocation step of K-means by a hashing technique. Compressed K-means (CKM) [21] encodes high-dimensional data into short binary codes to reduce expensive computation and memory costs. A-means [22] is based on the criterion that a data object may shift between its two closest centers at different iterations. Thus, it allows some data objects to be clustered in an early stage and excluded in subsequent iterations. Newling and Fleuret [23] developed a general improvement of K-means through estimating distance bounds, so as to reduce the number of distance calculations. Recently, Peng et al. [24] adopted a localization strategy to allocate data objects and a neighbor update strategy to quickly search neighbors for each cluster.

Heuristic strategies and data structures, e.g., KD-tree and dual-tree, have also been adopted to further improve the effectiveness and efficiency of K-means in literature. As a representative, Curtin [25] developed a dual-tree clustering algorithm which can yield the exact same results as the classical K-means algorithm. Especially, the single-iteration runtime is linear if cover trees are considered. Ball K-means [26] reduces the point-centroid distance computation by representing each cluster as a ball. Within these balls, data objects are tagged as stable or active ones, where the former is not changed while the latter are adjusted within a few neighbor clusters. As a result, only the distances between an object and its neighbor center, instead of all centers, are estimated. K*-means [12] first establishes a hierarchical structure for K-means by initializing $k*$ seeds of centers to reduce the risk of random selection, and then uses clusters merging and pruning strategies to improve the efficiency of K-means.

As we know, the classical K-means algorithm cannot effectively identify non-spherical clusters. To end this, various sophisticated techniques have been adopted. GK-means [27] is a typical example of such kind. It applies an approximate k-nearest neighbors graph to allocate data objects to clusters that their neighbors reside. Similarly, GKM (Graph-based K-means) [28] takes use of a graph technique to expose nonlinear manifold structures of data. Based on the structures, global information of data geometric distribution is available, and clusters can also be obtained accordingly. Besides, data density has also been exploited to detect non-spherical clusters. For instance, DBSCAN (Density-Based Spatial Clustering of Applications with Noise) [29] exploits data densities, as well as density-connected or reachable property, to identify clusters and noise, where are determined by a minimum number of neighbors in a given radius. Contrastively, DPC [15] takes those objects with larger densities and distances as cluster centers, and then tags the rest objects to the nearest centers with higher densities. Although it is very versatile, setting the cutoff threshold for DPC is a challenging issue. Thus, many variants, such as FastD-Peak [30] and SNN-DPC [31], attempt to address this problem. Fast LDP-MST [32] identifies arbitrary-shaped clusters by density peaks, coupled with minimum spanning tree for the sake of efficiency.

3. Materials and methods

In this section, we introduce an improved K-means clustering algorithm based on mutual neighbor, named MN-Kmeans. Fig. 1 gives a toy example of MN-Kmeans on a synthetic data. As illustrated in Fig. 1, MN-Kmeans mainly consists of three stages, where the first stage is to derive representative objects with high densities, estimated by mutual neighbors, from data. Afterwards, cluster centers are identified according to a notion of path-based distance, calculated on a minimum spanning tree. Finally, data objects are grouped into clusters by virtue of K-means, along with neighborhood information.

3.1. Data density

Data density is often used to formally represent distribution information of data objects, i.e., how dense or sparse the data objects locate in. Since the concept of density is vivid and intuitive to convey neighborhood information, it is also used to measure the structural property, e.g., manifold network or graph, of objects in literature [33]. Here we also exploit this concept to represent the data objects. Before

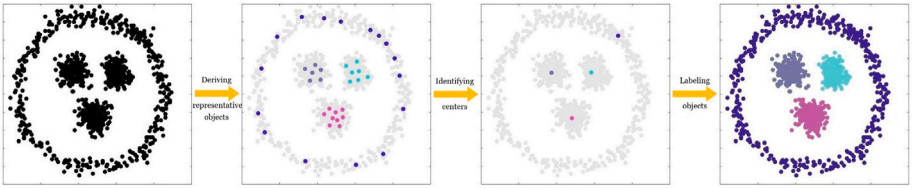

Fig. 1. A toy example of MN-Kmeans on a synthetic data, where MN-Kmeans consists of three stages: deriving representative objects, identifying cluster centers and grouping objects into clusters.

we delve into the density, let us retrospect to neighborhood relation of objects first.

Here and later, bold-faced uppercase and lowercase letters, e.g., $\mathbf{X}$ and $\mathbf{x}$, denote data matrices and (column) vectors respectively. x_{ij} and x_i indicate the (i,j)-th element of $\mathbf{X}$ and the ith element of $\mathbf{x}$ respectively. A notation $\|\cdot\|_{p,q}$ denotes the $\ell_{p,q}$ norm of matrices or vectors. For clarity, the letter $\mathbf{x}$ also represents a random variable (feature vector). Let $\mathbf{X} \in R^{n\times p}$ be a data collection consisting of n objects, i.e., $\mathbf{X}=\{\mathbf{x}_i\}_{i=1}^{n}$, where $\mathbf{x}_i \in \mathcal{R}^p$ is represented by a p-dimensional feature vector.

Given a data object $\mathbf{x} \in \mathbf{X}$, its nearest neighbors refer to the subset of objects that is defined as follows.

$$N(\mathbf{x}) = \{\mathbf{x}_i | d(\mathbf{x},\mathbf{x}_i) \le d(\mathbf{x},\mathbf{x}_j), \forall i,j = 1..n\}, \tag{1}$$

where $d(\mathbf{x},\mathbf{x}_i)$ is the Euclidean distance between $\mathbf{x}$ and $\mathbf{x}_i$. Based on this definition, the k nearest neighbors (kNN), $N_k(\mathbf{x})$, of $\mathbf{x}$ is that $N(\mathbf{x})$ contains k nearest neighbors, i.e., $|N_k(\mathbf{x})| = k$. Since the concept of kNN is intuitive, it has been widely used and many kNN algorithms have been developed. However, kNN is sensitive to the parameter k [34]. To address this concern, several variants, such as reverse nearest neighbors, shell neighbors, shared neighbors, natural neighbors and mutual neighbors, have been introduced [35].

One of the variants of kNN is mutual neighbor (MN). The central idea of mutual neighbor bridges nearest neighbors and reverse neighbors together to represent neighborhood information of data. Given a data object $\mathbf{x} \in \mathbf{X}$, its mutual neighbors are formally defined as

$$MN(\mathbf{x}) = \{\mathbf{x}_i | \mathbf{x}_i \in N_k(\mathbf{x}) \wedge \mathbf{x} \in N_k(\mathbf{x}_j)\}, \tag{2}$$

where $N_k(\mathbf{x})$ is the k nearest neighbors of $\mathbf{x}$. From the definition, we know that the notation of mutual neighbor is insensitive to the parameter. Since the mutual neighbor can exactly capture the interconnectivity of adjacent relations between data objects, it is often used to represent the connectivity properties of neighborhood graphs.

We exploit the notion of mutual neighbor to represent the density or distribution of data. As we know, a data object has many neighbors and many data objects also take it as neighbors simultaneously, if it locates at dense regions. On the contrary, the object has less neighbors and few of them take it as neighbor at the same time if it is in sparse regions. This property is inherently and naturally consisting to the notion of mutual neighbor. That is to say, the objects locating within dense regions have more mutual neighbors. With this assumption, we take the number of mutual neighbors for a data object as its density. Formally, the density of $\mathbf{x} \in \mathbf{X}$ is denoted as

$$DS(\mathbf{x}) = \frac{|MN(\mathbf{x})|}{k}, \tag{3}$$

where k is the number of neighbors or the maximum number of mutual neighbors.

Generally, a cluster center locates at the central position, and has highest density than others. Inspiring by this, we also choice those data objects with high densities to stand for other objects within the same cluster. Let $\mathbf{x} \in \mathbf{X}$ be a data object, it is a representative object if its density is larger than its mutual neighbors, that is,

$$RC(\mathbf{x}) = \{\mathbf{x}_i \in MN(\mathbf{x}) | DS(\mathbf{x}) \ge DS(\mathbf{x}_i)\}. \tag{4}$$

Given the data collection $\mathbf{X}$, there are many representative objects, each standing for its mutual neighbors. It seems to be reasonable to take them as potential cluster centers. In fact, these representative objects are local ones. This implies that not all representative objects are cluster centers, but the cluster centers are definitely representative ones.

We can further refine the cluster centers from the representative objects. Note that the representative objects are accessible to each other and connected thorough data objects. Thus, there is some paths, i.e., connection relations, between two adjacent representative objects. With the connection relations, the representative objects can be further combined together to form cluster centers. For instance, let y and z be representative objects of x and y, respectively, that is, $RC(x) = y$ and $RC(y) = z$. Then the representative object of x can be represented as z, because the density of z is larger than that of y, and z is more likely becoming a cluster center than y. This updating strategy is called chain rule, which can help us to identify cluster centers from a larger region. In the subsequent processing of our method, we only need to cluster these representative objects, which greatly reduces the scale of data. The clustering result on representative objects can be easily expanded to the entire data according to the connection relations.

3.2. Path-based distance

Given a data collection $\mathbf{X}$, there are many representative objects. It is observed that only several of them may become cluster centers while most of them are unimportant, if they are clustered. Generally, the representative ones with larger densities have higher probabilities to become cluster centers. For the sake of efficiency, the representative objects with lower densities should be excluded before the clustering stage. Specifically, we introduce a threshold θ to filter those representative objects with lower density. For each object $\mathbf{x}$, if its density is larger than the threshold, i.e., $DS(\mathbf{x}) \ge \theta$, it will be considered during the clustering stage. Otherwise, it will be considered as noisy one and filtered straightforwardly. Empirically, the value of θ can be determined by various strategies. As an example, it can be assigned to the $p * n$-th minimum density, where p is a proportion of data densities, sorted in ascending order.

As discussed above, we now only need to cluster the representative objects with high densities. Unfortunately, the Euclidean distance fails to measure the dissimilarities between objects within manifold structures, because of its global property [36]. To end this, local strategies, e.g., neighborhood relations, are often taken into consideration for manifold data. Here, we exploit the neighborhood relation, coupling with the Euclidean distance, to redefine the distance between the representative objects.

Assume that $\mathbf{x}$ and $\mathbf{y}$ are two representative objects, i.e., $DS(\mathbf{x}) \ge \theta$ and $DS(\mathbf{y}) \ge \theta$. The common neighbors of $\mathbf{x}$ to $\mathbf{y}$ are the intersection of their mutual neighbors, that is,

$$CN(\mathbf{x},\mathbf{y}) = \{\mathbf{x}_i | \mathbf{x}_i \in MN(\mathbf{x}) \cap MN(\mathbf{y})\}. \tag{5}$$

Based on the common neighbors, the distance between the representative objects $\mathbf{x}$ and $\mathbf{y}$ is formally represented as follows,

$$ND(\mathbf{x},\mathbf{y}) = \begin{cases} \frac{d(\mathbf{x},\mathbf{y})}{|CN(\mathbf{x},\mathbf{y})| \times \sum_{\mathbf{z}\in CN(\mathbf{x},\mathbf{y})} DS(\mathbf{z})} & , \quad |CN(\mathbf{x},\mathbf{y})| > 0, \\ \delta \times (1 + d(\mathbf{x},\mathbf{y})) & , \quad |CN(\mathbf{x},\mathbf{y})| = 0. \end{cases} \tag{6}$$

where $d(\mathbf{x},\mathbf{y})$ is the Euclidean distances between $\mathbf{x}$ and $\mathbf{y}$, and δ is the maximum value of $d(\mathbf{x},\mathbf{y})$.

According to Eq. (6), one may observe that the distance or dissimilarity between two representative objects will be compressed, if they have common neighbors. Otherwise, their distance is amplified. Besides, the more the common neighbors, the closer the distance. This is inherently consistent with the common sense that two representative objects locating a dense area are more similar than those separated by sparse areas, because they are connected with more neighbors.

To further measure the local property of manifold data, we exploit a concept of path-based distance on a minimum spanning tree to represent the distance of representative objects. Specifically, we first construct a minimum spanning tree from the representative objects and their distances, representing vertexes and edges of the spanning tree respectively. With the minimum spanning tree, the path-based distance of two vertexes refers to the path from a vertex to another one. That is to say, let $\mathbf{x}_1$ and $\mathbf{x}_m$ be two representative objects, their path-based distance $PD(\mathbf{x}_1, \mathbf{x}_m)$ is the weighted path from $\mathbf{x}_1$ to $\mathbf{x}_m$ on the spanning tree. Formally,

$$PD(\mathbf{x}_1, \mathbf{x}_m) = \sum_{i=1}^{m-1} ND(\mathbf{x}_i, \mathbf{x}_{i+1}), \tag{7}$$

where $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m$ is the path from $\mathbf{x}_1$ to $\mathbf{x}_m$ on the spanning tree. Since the path-based distance takes neighborhood information into consideration, it has the local property for manifold data and can effectively measure similarity between the representative objects, making the objects within the same cluster is more similar than that in different clusters.

3.3. Data clustering

With the help of the representative objects and the path-based distance, here we introduce an improved K-means clustering algorithm. As mentioned above, the performance of K-means heavily relies on the quality of initial cluster centers. To choice high quality initial centers, we first take the representative object with the maximum density as the first cluster center. Subsequently, the representative object with the largest path-based distance from the chosen centers is considered as the following center. This selection stage proceeds until the number of chosen centers reaches to the given one. The chosen initial centers have higher quality and can speed up the convergence of clustering procedure.

Following the routine of K-means, we iteratively update the centers and cluster the rest data objects in the next step, after the initial cluster centers available. To capture the intrinsic geometric structures of data, the new centers should be obtained from the manifold. Meanwhile, the similarities between the new centers are estimated on their path-based distances.

Assume the data collection $\mathbf{X}$ consists of m clusters $C_i (i = 1, 2, \ldots, m)$. For the ith cluster center $\mathbf{c}_i$ of $\mathbf{X}$, it can be determined as follows,

$$\mathbf{c}_i = \arg\min_{\mathbf{x} \in C_i} \sum_{\mathbf{z} \in C_i} PD(\mathbf{x}, \mathbf{z}). \tag{8}$$

From the equation above, the new center of the ith cluster C_i refers to the representative object that has the minimum sum of path-based distances to others in the same cluster.

Once the initial cluster centers are available, the next stage of clustering is to determine which cluster the data objects belonging to. Similar to the routine strategy of the classical K-means algorithm, we assign each object to the closest centers with the minimum path distance. Let $\mathbf{c}_i (i = 1..m)$ be m cluster centers. The cluster label of the representative object $\mathbf{x}$ is also determined by the path distance to the cluster centers, that is,

$$\mathbf{c}_k = \arg\min_{\mathbf{c}_i} PD(\mathbf{x}, \mathbf{c}_i). \tag{9}$$

In a nutshell, Algorithm 1 summarizes the implementation details of the improved K-means clustering method with mutual neighbors, dubbed as MN-Kmeans. The clustering process of MN-Kmeans mainly consists of four stages, where the first stage is to obtain the mutual neighbors (step 3) and estimate the density (step 4) for each data object. Based on the densities of objects, representative objects are picked out while noisy ones are filtered. Within this stage (from step 6 to step 9), the chain rule is applied to further refine representative objects. Meanwhile, the efficiency of clustering can be improved along with less representative objects.

The third stage (from step 10 to step 12) plays a core role in MN-Kmeans. It aims at constructing the minimum spanning tree from the representative objects obtained by the previous stage. Before constructing the tree, the distances based on common neighbors between the representative objects should be estimated at first. Once the tree available, we can calculate the path-based distances between any pair of vertexes in the tree. The reason of taking the path-based distance is to exactly represent the local structural property of manifold data.

Revamping the classical K-means algorithm to orient manifold data is the last stage of our method (from step 13 to step 18). Rather than randomly picking cluster centers in the classical one, here we deliberately choose m initial cluster centers with high densities and large distances. With the high-quality centers, we iteratively tag the representative objects to the cluster labels which correspond to the centers closest to the objects. Afterwards, the cluster centers will also be updated accordingly. This iteration is continue, until the centers have not been changed. Finally, all objects in the data collection are tagged to their nearest centers.

Algorithm 1. MN-Kmeans: Mutual neighbors-based K-means

Input: The data collection X, the cluster number m and the proportion of data p;
Output: The cluster labels of data objects in X;
(1) $n = |X|$;
(2) For each object $\mathbf{x}_i \in X (i = 1..n)$ **do**
(3) Get the mutual neighbors of $\mathbf{x}_i$ via Eq. (2);
(4) Estimate the density of $\mathbf{x}_i$ via Eq. (3);
(5) **end**
(6) Sort the densities $DS(\mathbf{x})$ in a descending order;
(7) Obtain top p-percent objects whose densities are larger than the threshold;
(8) Derive representative objects $RC \subseteq X$ from the top p objects via Eq. (4);
(9) For each object $\mathbf{x} \in RC$, update $RC(\mathbf{x})$ by virtue of the chain rule;
(10) For each pair of representative objects $\mathbf{x}$ and $\mathbf{y}$, calculate $ND(\mathbf{x}, \mathbf{y})$ by Eq. (6);
(11) Construct a minimum spanning tree with $ND(\mathbf{x}, \mathbf{y})$;
(12) For each pair of vertexes $\mathbf{x}$ and $\mathbf{y}$, estimate their path-based distance by Eq. (7);
(13) Initialize m cluster centers $\mathbf{c}_i (i = 1..m)$;
(14) **Repeat**
(15) Assign each representative object to its nearest center;
(16) Update m centers $\mathbf{c}_i$ by Eq. (9);
(17) **Until** the clustering process converges;
(18) Tag each object to the closest cluster label;

Given the data collection $\mathbf{X}$ consisting of n data objects. The time of the first stage, i.e., obtaining mutual neighbors and estimating the densities, of MN-Kmeans mainly lies in searching mutual neighbors. In a general case, its time complexity is $O(n^2)$. If heuristic strategies or data structures, e.g., KD-tree, were adopted, the time cost can be reduced to $O(n \log n)$. For the stage of identifying representative objects, most of time are spent on sorting the densities of objects (step 6), whose time cost is $O(n \log n)$. Indeed, both deriving the initial representatives

Table 1
The brief properties of synthetic data.

Data sets	*#Objects*	*#Clusters*
D6	1400	4
E6	8537	7
CTH	1156	4
Square	1741	6

and updating them with the chain rule are $O(n)$. Suppose p is the quantity of representative objects. In the third stage, it took $O(n+p^2)$ to get common neighbors, $O(p^2)$ to construct the minimum spanning tree and $O(p^2)$ to estimate the path distances for the representative objects. Since the last stage of MN-Kmeans works like the classical K-means algorithm, its time complexity is $O(p^2)$. As mentioned above, the quantity of representative objects is far less than that of the original objects. Therefore the overall time complexity of MN-Kmeans is $O(n^2)$.

4. Experiments on synthetic data

To demonstrate the effectiveness of the proposed method, we made a comparison of MN-Kmeans to K-means [8], DPC [15], DPC-KNN [37] and DBSCAN [29] on synthetic data sets. Among the baselines, K-means, DBSCAN and DPC are three classical and popular clustering algorithms, whereas DPC-KNN is also an improved DPC clustering algorithm for manifold data. It first performs PCA to reduce dimension of data and then constructs a kNN graph on all objects to derive neighborhood information of the objects. The experiments were conducted on a PC with i7 1.8 GHz CPU and 8 GB RAM.

In this section, we mainly discuss the experimental results of MN-Kmeans to the baselines on four synthetic data sets. These synthetic data sets were frequently used to verify the performance of clustering algorithms. They contain many noisy objects and consist of complex cluster structures. Table 1 presents brief properties of them.

For the baseline clustering algorithms, i.e., DPC, DPC-KNN and DBSCAN, they have different parameters which should be set accordingly. In our experiments, we followed the routine and assigned the parameters for the clustering algorithms to default or recommended values as suggested by authors in literature. For example, the *MinPts* and r of DBSCAN were assigned as 6 and 0.7, respectively. For DPC and DPC-KNN, five percents of objects were taken as candidate cluster centers. Since it is not reasonable to quantitatively evaluate the clustering performance on the synthetic data, we only presented the visual results.

Fig. 2 presents the performance comparison of the clustering algorithms on the synthetic data sets, where each row corresponds to one clustering algorithm and each column stands for one data set. From Fig. 2, we can learn that K-means failed to identify non-spherical clusters from manifold data. Similarly, DPC and DPC-KNN are also not suitable for manifold structural data. Even so, the DPC clustering algorithm accurately got the number of clusters, but it divided each cluster in manifold data into different clusters and took different clusters as one cluster. DBSCAN could identify correctly the clusters in *D6*, *CTH* and *Square*, but it considered many normal data objects in *E6* as noises. Besides, facing with well-distributed data objects, DBSCAN may fail to discover clusters from them. It can be observed that the proposed method, MN-Kmeans, exactly identified those spherical and manifold clusters from the synthetic data sets, even in the case of noises.

5. Applications to medical data

To further validate the effectiveness of MN-Kmeans, in this section we also applied it to the scenarios of medical diagnosis to cluster disease types. We downloaded seven public medical data sets from UCI Machine Learning Repository. They are *Cryotherapy*, *Dermatology*, *Haberman*, *HCV-Egy*, *Lymphography*, *Thyroid* and *Yeast*. The brief description information, including the numbers of data objects, features and clusters, of medical data sets are summarized in Table 2. Following gives brief information about these medical data.

Table 2
The brief properties of experimental data sets.

Data sets	*#Features*	*#Objects*	*#Clusters*
Cryotherapy	6	90	2
Dermatology	34	366	6
Haberman	3	306	2
HCV-Egy	28	1385	4
Lymphography	18	148	4
Thyroid	5	215	3
Yeast	22	187	2

- *Cryotherapy* consists of 90 patient information about wart treatment using cryotherapy.
- *Dermatology* is about six types of Eryhemato-Squamous disease with 366 patients.
- *Haberman* describes the survival of breast cancer patients after surgery.
- *HCV-Egy* contains blood donors and Hepatitis C patients, which can be clustered into four groups.
- *Lymphography* includes 148 patients with four different sub-types of lymph.
- *Thyroid* records patient information about thyroid disease.
- *Yeast* concerns about cellular localization sites of proteins.

To quantify the effectiveness of MN-Kmeans, we adopted AC (Accuracy) and NMI (normalized mutual information) [38] to compare the clustering results from each algorithm to the ground truth. They are frequently used to measure clustering performance in literature. Let p_i and q_i be the ground truth and derived cluster label of the ith object $\mathbf{x}_i \in \mathbf{X}$. AC is formally defined as

$$AC(\mathbf{X}) = \frac{1}{n}\sum_{i=1}^{n} I(p_i = q_i), \tag{10}$$

where $I(\cdot)$ is an indication function, and $I(p_i = q_i)$ equals to 1 if the clustering label qi of $\mathbf{x}_i$ is the same to its ground-truth p_i. Otherwise, $I(p_i = q_i)$ is zero.

NMI is capable of describing the structural quality of clusters and the degree of conformity with the actual cluster structure. It is also used to measure the similarity between the ground truth and the clusters derived by a clustering algorithm. For two different data clusters C_i and C_j, their NMI is shown as follows:

$$NMI(C_i, C_j) = \frac{2 \times I(C_i, C_j)}{H(C_i) + H(C_j)}, \tag{11}$$

where $I(C_i, C_j) = H(C_i) + H(C_j) - H(C_i, C_j)$ is the mutual information between C_i and C_j. $H(C_i)$ is the information entropy associated with the cluster C_i.

For these two metrics above, their value is in the range of $[0, 1]$. What is more, the larger their value is, the more similar the two clusters are. or the better the clustering is. When the NMI value is 1, the two clusters are exactly the same.

Table 3 provides the performance comparison of AC achieved by the clustering algorithms on the experimental data sets, where the bold value indicates that it is the best one among the clustering algorithms on the corresponding data set. According to the experimental results, one can conclude that the proposed method significantly outperforms the comparison baselines, because the accuracy achieved by MN-Kmeans is the best one over all data sets. Among the baselines, K-means and DBSCAN had also comparable performance, but DPC-KNN achieved relatively poor performance.

Analogous cases can be observed from the performance comparison of NMI, whose values are offered in Table 4. From the table, we can also

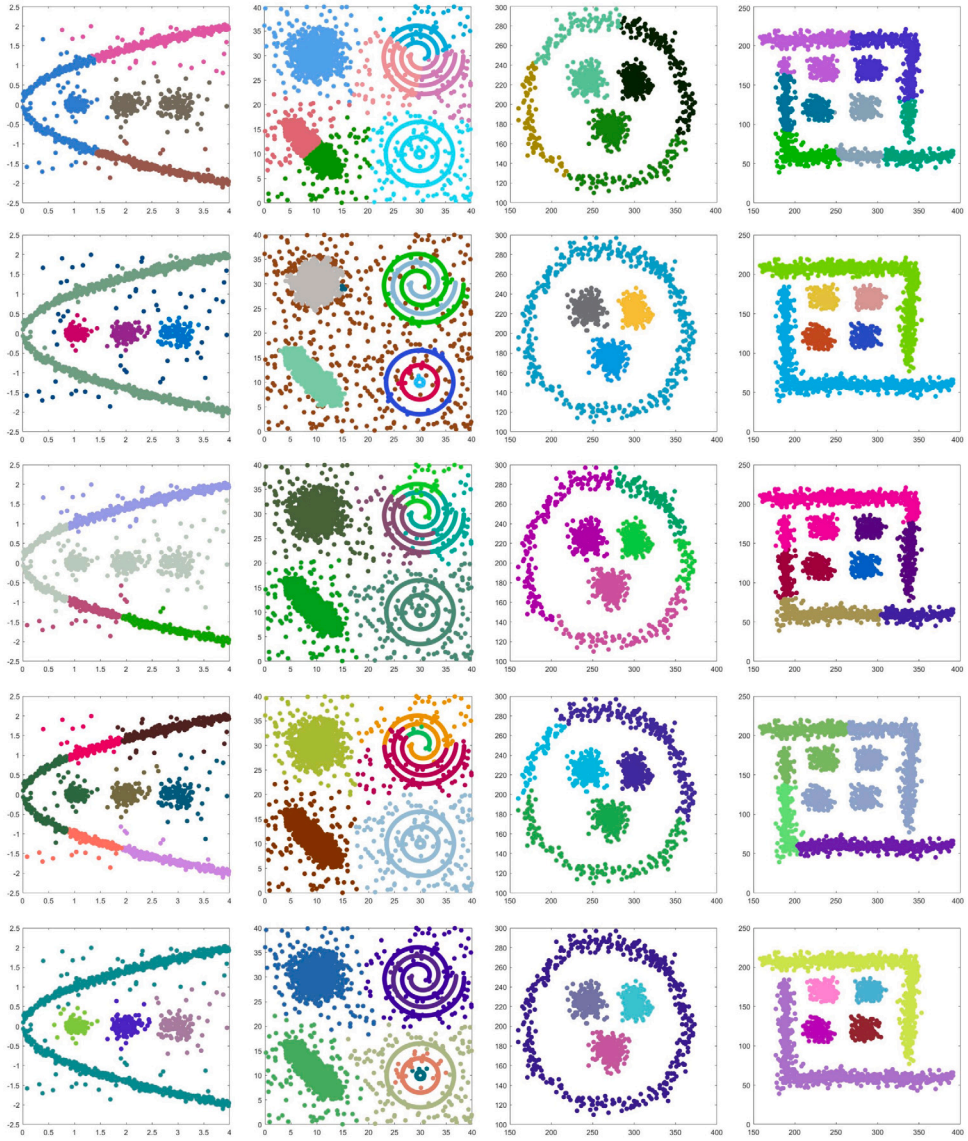

Fig. 2. The performance comparison of clustering algorithms on the synthetic data sets with noisy data. Each row corresponds to a clustering method (From top to bottom, they are K-means, DBSCAN, DPC, DPC-KNN and MN-Kmeans, respectively), and each column represents a data set (From left to right, they are *D6*, *E6*, *CTH* and *Square*, respectively).

Table 3
The AC comparison of clustering methods on the data sets.

	K-means	DBSCAN	DPC	DPC-KNN	MN-Kmeans
Cryotherapy	0.567	0.567	0.567	0.601	**0.611**
Dermatology	0.858	0.306	0.653	0.626	**0.934**
Haberman	0.501	0.735	0.569	0.536	**0.755**
Lymphography	0.439	0.547	0.568	0.501	**0.581**
HCV-Egy	0.267	0.261	0.269	0.201	**0.281**
Thyroid	0.781	0.837	0.758	0.712	**0.861**
Yeast	0.411	0.319	0.317	0.375	**0.456**

observe that MN-Kmeans is still superior to the baselines, because it had seven highest NMI values. Although the NMI value of MN-Kmeans on *Soybean* is smaller than that of DPC-KNN, it is still larger than others. Another interesting fact is that DBSCAN had not achieved good performance on the real data as it did on the synthetic data. Perhaps the underlying reason is that it involves several parameters, which are hard to set without prior knowledge and may bring great impacts to its performance.

Computational efficiency is another aspect which should be considered in real-world applications. We also recorded the running time of clustering algorithms in our experiments and showed in Fig. 3. As illustrated in Fig. 3, MN-Kmeans had comparable efficiency to the baselines. Although it took relatively more time than K-means and DBSCAN on *Yeast* and *Dermatology*, it was still better than DPC and DPC-KNN. It is reasonable that obtaining mutual neighbors usually requires more time than that of k nearest neighbors. Meanwhile, we adopted traditional techniques, rather than KD-tree, to derive mutual neighbors in the implement of MN-Kmeans. It is noticeable that DPC-KNN had poor efficiency, because it performed both PCA and kNN before the clustering stage of DPC.

Table 4
The NMI comparison of clustering methods on the data sets.

	K-means	DBSCAN	DPC	DPC-KNN	MN-Kmeans
Cryotherapy	0.377	0.381	0.377	0.331	**0.422**
Dermatology	0.848	0.461	0.493	0.512	**0.891**
Haberman	0.110	0.493	0.106	0.211	**0.502**
Lymphography	0.408	0.311	0.377	0.306	**0.433**
HCV-Egy	0.211	0.201	**0.331**	0.201	**0.331**
Thyroid	0.387	0.403	0.286	0.262	**0.429**
Yeast	0.253	0.229	0.236	0.266	**0.281**

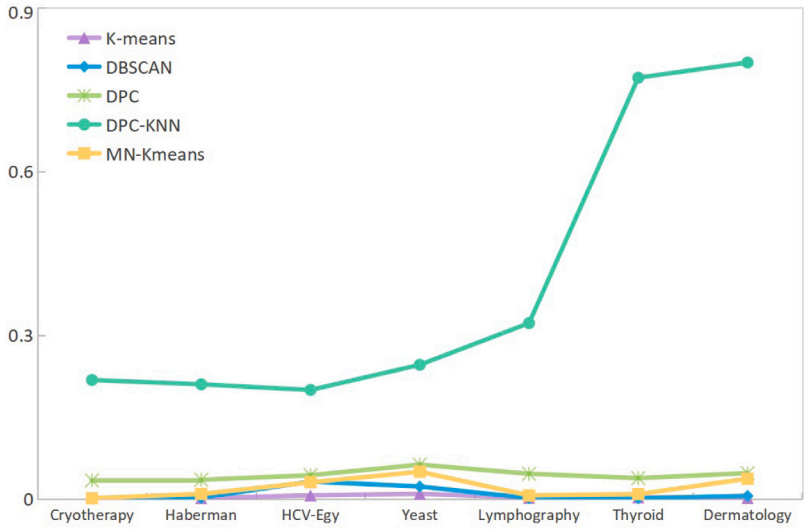

Fig. 3. The running time (second) of clustering algorithms on the experimental data sets.

6. Conclusions

Since the Euclidean distance is a global one, it cannot exactly represent neighborhood information, making the classical K-means algorithm fail to identify non-spherical clusters. To end this, in this paper a novel clustering algorithm based on representative objects derived from mutual neighbors is proposed. It first exploits mutual neighbors to estimate data densities, which can exactly represent the distributions of data objects. Later it picks representative objects out and calculates their path-based distances by a minimum spanning tree. The path-based distances have local properties and are capable of measuring similarities of objects in manifold structures. Finally, an improved K-means with the path-based distances is developed to achieve the clustering purpose, after deliberately choosing initial centers. The experimental results on both synthetic and medical data sets show that the proposed clustering algorithm is more superior to the popular clustering algorithms in identifying arbitrary-shaped clusters.

Since MN-Kmeans needs to derive representative neighbors from mutual neighbors and construct the minimum spanning tree, its time complexity is relative higher than the classical K-means algorithm. Our future work will focuses on this issue by virtue of sophisticated techniques.

Declaration of competing interest

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

Acknowledgments

The authors would like to thank the anonymous referees and EIC for their valuable comments and suggestions, which have improved the paper vastly. This work was partially supported by the national NSF of China (NSFC) (61976195, 61976196), Outstanding Talents of “Ten Thousand Talents Plan” in Zhejiang Province, China (2018R51001), and Zhejiang Provincial Natural Science Foundation of China (LZ22F030003).

References

[1] M. Wang, W. Fu, X. He, S. Hao, X. Wu, A survey on large-scale machine learning, IEEE Trans. Knowl. Data Eng. 34 (6) (2022) 2574–2594.
[2] H. Liu, X. Li, S. Zhang, Q. Tian, Adaptive hashing with sparse matrix factorization, IEEE Trans. Neural Netw. Learn. Syst. 31 (10) (2020) 4318–4329.
[3] P. Giordani, M.B. Ferraro, F. Martella, An Introduction to Clustering with R, Springer, Singapore, 2020.
[4] R. Xu, D. Wunsch, Survey of clustering algorithms, IEEE Trans. Neural Netw. 16 (3) (2005) 645–678.
[5] A. Saxena, M. Prasad, A. Gupta, N. Bharill, O.P. Patel, A. Tiwari, M.J. Er, W. Ding, C.-T. Lin, A review of clustering techniques and developments, Neurocomputing 267 (2017) 664–681.
[6] R. Yu, Y. Tian, J. Gao, Z. Liu, X. Wei, H. Jiang, Y. Huang, X. Li, Feature discretization-based deep clustering for thyroid ultrasound image feature extraction, Comput. Biol. Med. 146 (2022) 105600.
[7] Y. Hozumi, R. Wang, C. Yin, G. Wei, UMAP-assisted K-means clustering of large-scale SARS-CoV-2 mutation datasets, Comput. Biol. Med. 131 (2021) 104264.
[8] J. Macqueen, Some methods for classification and analysis of multivariate observations, in: Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability, 1967, pp. 281–297.
[9] B. Mullick, R. Magar, A. Jhunjhunwala, A.B. Farimani, Understanding mutation hotspots for the SARS-CoV-2 spike protein using Shannon entropy and K-means clustering, Comput. Biol. Med. 138 (2021) 104915.
[10] X. Wu, F. Shi, Y. Guo, Z. Zhang, J. Huang, J. Wang, An approximation algorithm for lower-bounded K-median with constant factor, Sci. China Inf. Sci. 65 (2022) 140601.
[11] D. Arthur, S. Vassilvitskii, K-Means++: The advantages of careful seeding, in: Proceedings of the 8th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA07, 2007, pp. 1027–1035.
[12] J. Qi, Y. Yu, L. Wang, J. Liu, Y. Wang, An effective and efficient hierarchical K-means clustering algorithm, Int. J. Distrib. Sens. Netw. 13 (8) (2017) 1550147717728627.

[13] E. Schubert, P.J. Rousseeuw, Faster K-medoids clustering: Improving the PAM, CLARA, and CLARANS algorithms, in: G. Amato, C. Gennaro, V. Oria, M. Radovanović (Eds.), Similarity Search and Applications, Springer International Publishing, Cham, 2019, pp. 171–187.
[14] S. Huang, Z. Kang, Z. Xu, Q. Liu, Robust deep K-means: An effective and simple method for data clustering, Pattern Recognit. 117 (2021) 107996.
[15] A. Rodriguez, A. Laio, Clustering by fast search and find of density peaks, Science 344 (2014) 1492–1496.
[16] J. Ding, S. Shah, A. Condon, Densitycut: An efficient and versatile topological approach for automatic clustering of biological data, Bioinformatics 32 (17) (2016) 2567–2576.
[17] B.J. Frey, D. Dueck, Clustering by passing messages between data points, Science 315 (5814) (2007) 972–976.
[18] X. Xie, H. Liu, S. Zeng, L. Lin, W. Li, A novel progressively undersampling method based on the density peaks sequence for imbalanced data, Knowl.-Based Syst. 213 (2021) 106689.
[19] B. Roy, T. Stepisnik, C. Vens, S. Dzeroski, Survival analysis with semi-supervised predictive clustering trees, Comput. Biol. Med. 141 (2022) 105001.
[20] Q. Hu, J. Wu, L. Bai, Y. Zhang, J. Cheng, Fast K-means for large scale clustering, in: Proceedings of the 2017 ACM Conference on Information and Knowledge Management, CIKM17, ACM, 2017, pp. 2099–2102.
[21] X. Shen, W. Liu, I. Tsang, F. Shen, Q.-S. Sun, Compressed K-means for large-scale clustering, in: Proceedings of the 31st AAAI Conference on Artificial Intelligence, AAAI17, AAAI Press, 2017, pp. 2527–2533.
[22] J. Ortega, N. Ortega, J. Ruiz-Vanoye, R. R., S. Saenz-Sanchez, J. Rodriguez-Lelis, A. Martinez-Rebollar, A-means: Improving the cluster assignment phase of K-means for big data, Int. J. Combinatorial Optim. Probl. Inform. 9 (2) (2018) 3–10.
[23] J. Newling, F. Fleuret, Fast K-means with accurate bounds, in: Proceedings of the 33rd International Conference on Machine Learning, ICML16, JMLR.org, 2016, pp. 936–944.
[24] D. Peng, Z. Chen, J. Fu, S. Xia, Q. Wen, Fast K-means clustering based on the neighbor information, in: Proceedings of the 2021 International Symposium on Electrical, Electronics and Information Engineering, ACM, 2021, pp. 551–555.
[25] R.R. Curtin, A dual-tree algorithm for fast K-means clustering with large k, in: Proceedings of the 2017 SIAM International Conference on Data Mining, SDM, 2017, pp. 300–308.
[26] S. Xia, D. Peng, D. Meng, C. Zhang, G. Wang, E. Giem, W. Wei, Z. Chen, Ball k-means: Fast adaptive clustering with no bounds, IEEE Trans. Pattern Anal. Mach. Intell. 44 (1) (2022) 87–99.
[27] C.-H. Deng, W.-L. Zhao, Fast K-means based on k-NN graph, in: Proceedings of the IEEE 34th International Conference on Data Engineering, ICDE18, 2018, pp. 1220–1223.
[28] E. Tu, L. Cao, J. Yang, N. Kasabov, A novel graph-based K-means for nonlinear manifold clustering and representative selection, Neurocomputing 143 (2014) 109–122.
[29] A. Bryant, K. Cios, RNN-DBSCAN: A density-based clustering algorithm using reverse nearest neighbor density estimates, IEEE Trans. Knowl. Data Eng. 30 (6) (2018) 1109–1121.
[30] Y. Chen, X. Hu, W. Fan, L. Shen, Z. Zhang, X. Liu, J. Du, H. Li, Y. Chen, H. Li, Fast density peak clustering for large scale data based on kNN, Knowl.-Based Syst. 187 (2020) 104824.
[31] R. Liu, H. Wang, X. Yu, Shared-nearest-neighbor-based clustering by fast search and find of density peaks, Inform. Sci. 450 (2018) 200–226.
[32] T. Qiu, Y. Li, Fast LDP-MST: An efficient density-peak-based clustering method for large-size datasets, IEEE Trans. Knowl. Data Eng. (2022) 1, http://dx.doi.org/10.1109/TKDE.2022.3150403.
[33] H. Liu, X. Li, J. Li, S. Zhang, Efficient outlier detection for high-dimensional data, IEEE Trans. Syst. Man Cybern.: Syst. 48 (12) (2018) 2451–2461.
[34] S. Faisal, G. Tutz, Imputation methods for high-dimensional mixed-type datasets by nearest neighbors, Comput. Biol. Med. 135 (2021) 104577.
[35] H. Liu, X. Xu, E. Li, S. Zhang, X. Li, Anomaly detection with representative neighbors, IEEE Trans. Neural Netw. Learn. Syst. PP (2023) 1–11, http://dx.doi.org/10.1109/TNNLS.2021.3109898.
[36] H. Liu, E. Li, X. Liu, K. Su, S. Zhang, Anomaly detection with kernel preserving embedding, ACM Trans. Knowl. Discov. Data 15 (5) (2021) 91:1–91:18.
[37] 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.
[38] Q.-Z. Dai, Z.-Y. Xiong, J. Xie, X.-X. Wang, Y.-F. Zhang, J.-X. Shang, A novel clustering algorithm based on the natural reverse nearest neighbor structure, Inf. Syst. 84 (2019) 1–16.