Density-based Outlier Detection in Multi-dimensional Datasets

Xite Wang[1*], Zhixin Cao[1], Rongjuan Zhan[1], Mei Bai[1], Qian Ma[1] and Guanyu Li[1]
[1] Dalian Maritime University, Dalian, Liaoning, China
[e-mail: xite-skywalker@163.com, caozhixin@163.com]
[*]Corresponding author: Xite Wang

Received July 19, 2022; revised September 29, 2022; accepted November 20, 2022; published December 31, 2022

Abstract

Density-based outlier detection is one of the hot issues in data mining. A point is determined as outlier on basis of the density of points near them. The existing density-based detection algorithms have high time complexity, in order to reduce the time complexity, a new outlier detection algorithm DODMD (Density-based Outlier Detection in Multidimensional Datasets) is proposed. Firstly, on the basis of ZH-tree, the concept of micro-cluster is introduced. Each leaf node is regarded as a micro-cluster, and the micro-cluster is calculated to achieve the purpose of batch filtering. In order to obtain *n* sets of approximate outliers quickly, a greedy method is used to calculate the boundary of *LOF* and mark the minimum value as LOF_{min}. Secondly, the outliers can filtered out by LOF_{min}, the real outliers are calculated, and then the result set is updated to make the boundary closer. Finally, the accuracy and efficiency of DODMD algorithm are verified on real dataset and synthetic dataset respectively.

Keywords: outlier, multi-dimensional data, density-based, z-order curve, micro-cluster

1. Introduction

Outlier detection is one of the important research contents in the field of data mining [1]. Its goal is to obtain a small number of isolated and deviated information from other data objects in the huge amount of information. In early studies, many applications considered these data as noise and believed that these data should be eliminated to ensure the normal processing of other data [2]. Han proposed that "one person's noise may be another's signal" [3]. So the main goal of outlier detection is to mine valuable information from massive and complex data, and to deeply understand abnormal pattern [4]. Through the continuous research of scholars in the later period, a variety of outlier definition methods have been produced. The model-based outlier definition [5] and [6] is proposed, this definition first predicts a distribution model or probability model. If the data in the dataset does not meet the prediction model, it will be judged as outlier. However, many datasets have the characteristics of high dimension and complex type, so it is difficult to predict the accurate and efficient distribution model with this definition standard, and it will become more difficult for multi-dimensional data. The outlier definition of distance-based is proposed by Gustavo [7], it needs to select two parameters *k* and *r* in advance. But it is difficult to detect local outliers due to the limitation of the selection of two parameters. In order to overcome this limitation, the definition of density-based outlier has been widely used in the field.

1.1 Density-based outliers

In reality, the data is often extremely complex, and the outlier detection method based on distance sometimes has limitations. Based on distance, the whole dataset is usually considered without considering the concept of local. Therefore, this paper adopts the concept of density-based proposed by [8], and considers the density relationship between data points. Firstly, the local density is calculated, and then the data is scored by the local outlier factor to evaluate the anomaly degree of the data. The higher the value of local outlier, the smaller the density of the object near the object. The object is easier to determine as outlier than other objects. Different from model-based outlier definition [5] and [6], this method does not directly determine whether the object is abnormal, but uses local outlier factor to rank the data to determine the possibility of outlier. The density-based outlier considers the density relationship between objects, which makes the result more accurate.

1.2 Contributions

Some scholars have proposed many outlier detection methods for multidimensional data. This paper proposes a new density-based outlier detection algorithm DODMD in multidimensional data. Our contributions as follows.

(1) On the basis of ZH-tree, the concept of micro-cluster is introduced. Each leaf node is regarded as a micro-cluster, and the micro-cluster is calculated to achieve the purpose of batch filtering. A greedy method is used to retain the data objects with large outliers, and the values of the first n approximate result sets are calculated, and the minimum outlier LOF_{min} is marked.

(2) A new density-based outlier detection algorithm DODMD is proposed. The upper limit of outliers of each micro-cluster is calculated, and the LOF_{min} is used for batch filtering. At the same time, the $LOF_{min}$is dynamically updated to make the result more compact, reduce unnecessary calculation of data, and make the algorithm more efficient.

(3) The accuracy and efficiency of DODMD algorithm are verified on real dataset and synthetic dataset respectively.

The remaining specific content of this paper is as follows. Section 2 briefly introduces the related work of outlier detection. In Section 3, we discuss the problem of outlier detect of density-based in multidimensional data. Section 4 describe the algorithms used in this paper. Section 5 analyzes the experimental results. Finally, in section 6 the work done in this paper is summarized.

2. Related Work

Firstly, we briefly introduce the existing outlier definitions and traditional outlier detection methods. Secondly, the outlier detection methods of density-based in multidimensional data are summarized.

2.1 Tradition density-based outlier detection

Bay et al. [9] proposed a method called ORCA, which preprocesses the dataset before outlier detection, and uses random method to exclude normal data objects in the dataset. In addition, some scholars have devoted themselves to the research of spatial index structure to improve the search efficiency, such as R-tree [10], M-tree [11], etc.

Many scholars have proposed density-based outlier detection method. In reference [12], a IGBP algorithm is proposed. It uses the density of the relative object to indicate that the degree of the object is an outlier compared with its neighbors. In a distributed environment, greedy algorithm is used to detect density-based outliers in parallel. The LOCI method [13] is also a density-based method, which counts the number of neighbor objects of each data object within a distance range, and compares it with the average number of neighbor objects of neighbors. The outliers with large deviation are considered as outliers. In reference [14], the attributes of an object are divided into two attributes. They are outlier values and neighborhood of the object are calculated respectively with the two attributes, which can effectively improve the detection efficiency. In reference [15], different weights are added to different attributes when calculating distance. Higher weights are assigned to some outlier attributes than other normal attributes, which can improve the accuracy of the algorithm.

2.2 Density-based outlier detect in multidimensional data

At present, some scholars have proposed many outlier detection methods for multidimensional data. He et al. [16] proposed to divide the data set into many large and small clusters. If some objects deviate from these clusters, they are considered as outliers. Different detection methods have been developed for different application backgrounds, such as outlier detection method for stream data [17], and outlier detection method for uncertain data [18] and [19]. Aggarwal et al. [20] proposed a detection method to detect outliers by observing the low-dimensional projection of the search space. When a point is located in a low-dimensional subspace, it is regarded as an outlier. In order to find the abnormally low-density low-dimensional projection, the author uses a genetic algorithm to find the dimension combination of sparse data. The search subspace is composed of various dimensions, and the time complexity is always exponential. Angiulli et al. [21] used the spatial filling curve to reduce the multidimensional data to the low dimension for operation. Through this dimension reduction technology, the spatial proximity between the original data remains unchanged. This method creates many unnecessary nodes, resulting in insufficient space. Kriegel H P [22] proposed a novel solution to the problem of sparse

multidimensional data space, using the angle between points to detect outliers.

In this paper, the density-based definition of outliers is selected, and Angiulli's [21] method is used to reduce the dimensionality of multidimensional data. However, the shortcomings of Fabrizio's method are also avoided. Outliers are filtered by reasonable methods to improve the computational efficiency.

3. Problem Definition

Table 1 The mathematical notions used in this paper are summarized.

Table 1. Summary of notions

Notion	Definition
D	Dataset with |D| points in d space
d	Data dimension
MC	Micro-cluster
k_{max}	Maximum value of k
$N_k(p)$	k neighborhood of point p
$d(o,p)$	Distance between o and p
n	Number of outlier
$k-distance(p)$	The k-th distance from p
$k_{max}-distance(MC)$	The maximum of k-th distance of the points in micro-cluster MC
$k_{min}-distance(MC)$	The minimum of k-th distance of the points in micro-cluster MC
$Dist_{max}(p,MC)$	The maximum distance from p to micro-cluster MC
$Dist_{min}(p,MC)$	The minimum distance from p to micro-cluster MC
$Dist_{max}(MC_i,MC_j)$	The maximum distance between MC_i and MC_j
$Dist_{min}(MC_i,MC_j)$	The minimum distance from MC_i to MC_j

Given a d-dimensional dataset D, each point p in D is expressed as $p=(p[1],p[2],\dots,p[d])$, and the distance between p_1 and p_2 is expressed as

$$d(p_1,p_2)=\sqrt{\sum_{i=1}^{d}(p_1[i]-p_2[i])^2} \tag{1}$$

Definition 1(the k-distance of p) the k-distance of object p is expressed as $k-distance(p)$, which is the distance between p and another point $o\in D$, $d(o,p)$, such that

(1)There are at least k points $o'\in D-\{p\}$,make $d(p,o')\le d(p,o)$

(2)There are at most k-1 points $o''\in D-\{p\}$,make $d(p,o'')<d(p,o)$

Definition 2(k-distance neighborhood of p) the k-th distance neighborhood of p contains all points whose distance does not exceed its k-distance expressed as

$$N_k(p)=\{q\in D|\{p\}|d(p,q)\le k-distance(p)\} \tag{2}$$

Definition 3(reachable distance of p with respect to o) the reachable distance of p with respect to object o is expressed as

$$reach-dis_k(p,o)=\max\{k-distance(o),d(p,o)\} \tag{3}$$

The reachable distance between *o* and *p* is the maximum of the *k*-th distance of *o* and the distance between *o* and *p*. In **Fig. 1**, when *k*=5, the reachable distance between p and o_1 is $reach - dis_k(p, o_1) = d(p, o_1)$, and the reachable distance between p and o_2 is $reach - dis_k(p, o_2) = d(p, o_2)$.

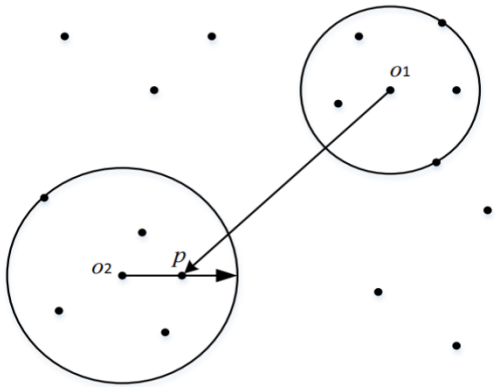


Fig. 1. The reachable distance between *p* and o_1 and the reachable distance between *p* and o_2

Definition 4(local reachable density of *p*) the local reachable density of *p* is expressed as the reciprocal of the average reachable distance between point in the *k*-th neighborhood of *p* and *p*, expressed as

$$lrd_k(p) = \frac{1}{\left[\frac{\sum_{o \in N_k(p)} reach\text{-}dist_k(p,o)}{|N_{k(p)}(p)|}\right]} \tag{4}$$

Definition 5(local outlier factor of *p*) local outlier factor of *p* is defined, the average value of the ratio of local reachable density of *p* to that of *k* nearest neighbor of *p*, expressed as

$$LOF_k(p) = \frac{\sum_{o \in N_k(p)} \frac{lrd_k(o)}{lrd_k(p)}}{|N_k(p)|} \tag{5}$$

Intuitively, if the local reachable density of *p* is much lower than that of its neighbors, the local outlier factor of *p* is very high. In general, the lrd_k of an object is similar to that of its neighbors, indicating that the object is in a cluster, and its *LOF* value is less than or equal to 1, and the object is a normal point. If the lrd_k of an object is larger than the lrd_k of its neighbors, it means that the object is located at the cluster boundary. If the *LOF* of the object is greater than 1, the object is a local outlier.

The calculation of *LOF* needs index structure to search the nearest neighbor of the object quickly. In this paper, the index structure ZH-tree based on space filling curve is adopted, and the concept of micro-cluster is introduced to achieve the purpose of filtering. The goal of this paper is to further improve the performance of the algorithm based on *LOF*, that is, to find the top-*n* outliers with the largest *LOF* values. Finally, the DODMD algorithm is proposed.

4. DODMD Description

In this chapter, we propose a density-based outlier detection algorithm DODMD. First, based on the process and characteristics of the *LOF* algorithm in the previous section, the ZH-tree index structure [23] based on a spatial filling curve is adopted and improved appropriately. Then, based on the *LOF* analysis of how to further improve the performance of the algorithm, the concept of micro-cluster is introduced, and the related definitions, theorems, and corollaries are introduced. Finally, the DODMD algorithm is proposed.

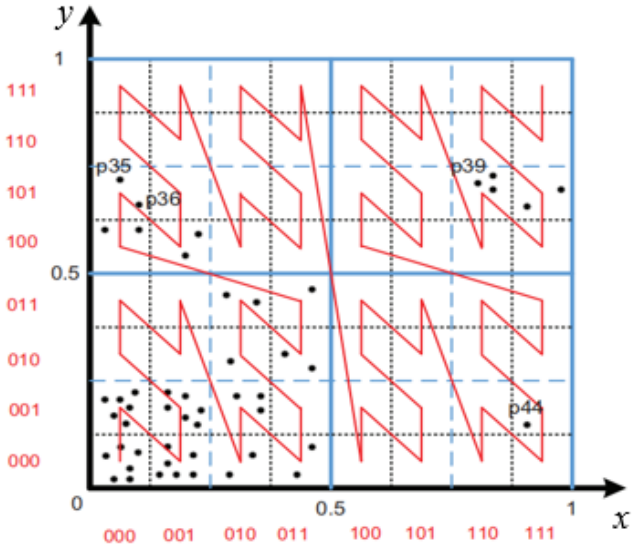


Fig. 2. Z-order curve

ZH-tree [23] is an index structure based on space filling curve. We first use an example to briefly introduce how ZH-tree is constructed. The height of ZH-tree is *h*, the root node 0 represents the whole data space, and the non-leaf node at level $i([1,H])$ represents the subspace generated in the *i*-th iteration. The child node of each non-leaf node *e* is the space generated by the *i*+1 iteration of the corresponding space of the node. When k_{max}=3 and *h*=3, **Fig. 4** shows the ZH-tree constructed by the points in **Fig. 2**. As shown in **Fig. 2**, there is a two-dimensional space. Each dimension is divided into 8 isometrical intervals. The corresponding coordinates of intervals are represented in binary, from 000 to 111. Therefore, the two-dimensional space is divided into 64 equal-sized cells. Each cell is assigned a Z-address by cross connecting the first and second two-dimensional coordinates. For example, in **Fig. 2**, the Z-address of the cell where point p_{35} locates is 010001 (cross connecting 000 and 101, shown in **Fig. 3**). Then, in **Fig. 4**, we sort cells according to their Z-address in ascending order to form the leaf layer of ZH-tree. We continually extract common Z-address prefixes of cells to form the nodes in upper layers. At last, the ZH-tree index is constructed. The detailed method of ZH-tree can be found in [23].

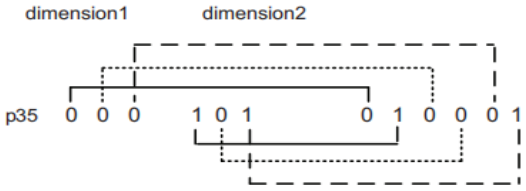


Fig. 3. Z-order curve

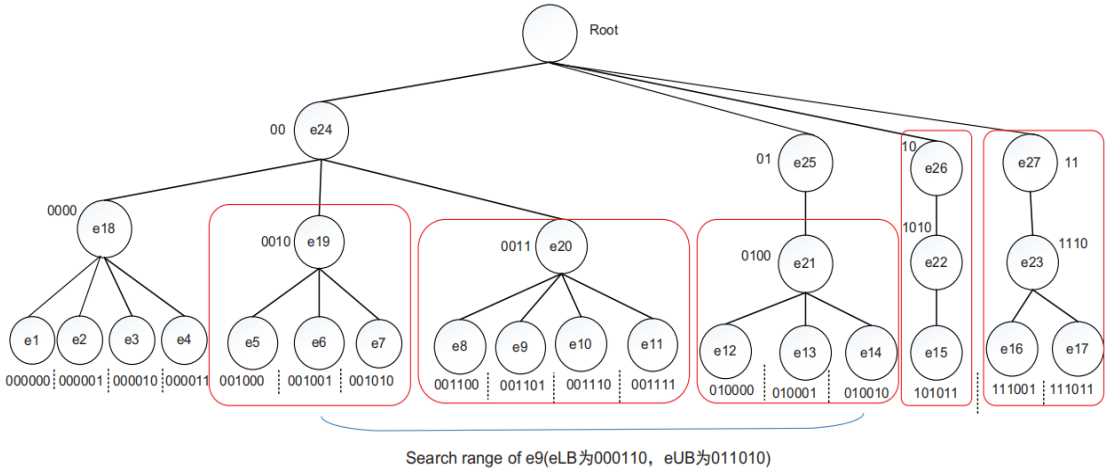


Fig. 4. ZH-tree of the points of **Fig. 2**

4.1 Approach of obtain ZH-tree nearly outliers

In order to find the final result, we need to calculate the local outlier factor LOF_{min}. According to the analysis, the maximum LOF_{min} of the micro-cluster can be calculated first. When the maximum LOF is still less than LOF_{min}, the points in the micro-cluster are safe points, and it is not necessary to calculate the real LOF. Therefore, finding LOF_{min} quickly is the key problem. In this section, we first introduce the related concepts, deduce how to find the boundary of LOF. Then propose a greedy method to select the outlier points to calculate the LOF_{min}, and finally determine the final result set by filtering refinement method.

4.1.1 Obtaining bounder of *LOF*

According to the density-based definition, if the upper and lower bounds of locally accessible density of p exist in $N_k(p)$, then the LOF upper and lower bounds of p can be easily obtained. Therefore, the following theorem is proposed.

Theorem 1 let $lrd_k(o).upper$ and $lrd_k(o).lower$ denote the upper and lower bounds of $lrd_k(o)$ of point p, and $o \in N_k(p)$, then

$$\frac{Min\{lrd_k(o).lower\}}{lrd_k(p).upper} < LOF_k(p) < \frac{Max\{lrd_k(o).upper\}}{lrd_k(p).lower} \tag{6}$$

Proof Since LOF is the ratio of the mean value of the k-th neighborhood object to p, the average must be greater than $Min\{lrd_k(o) \in N_k(p)\}/lrd_k(p)$. By obtaining the upper bound of $lrd(p)$ and the lower bound of $lrd_k(o)$, the estimated value of the lower bound of $LOF_k(p)$ can be further reduced. In the same way, the upper bound value of $LOF_k(p)$ can be obtained.

According to theorem 1, the following inference can be made to find the upper and lower bounds of the local reachable density of each point.

Corollary 1 Given a point p, o belongs to $N_k(p)$, the upper and lower bounds of its local reachable density are

$$\frac{1}{Max\{reach-dist_k(p,o)\}} \le lrd_k(p) \le \frac{1}{Min(reach-dist_k(p,o)\}} \quad (7)$$

Proof Because the local accessible density of p is the reciprocal of the average of the reachable distances of p, easy to get that the average value is smaller than the maximum value of $reach-dis_k(p,o)$ and larger than the minimum value of $reach-dis_k(p,o)$. So we can get the above formula.

According to the above inference, the maximum and minimum values of $reach-dis_k(p,o)$ can be restricted by the following theorem.

Theorem 2 Let *k-distance(o).upper* is the upper bound of the *k*-distance of *o*, *k-distance(o).lower* is the lower bound of the *k*-distance of *o*, and let the $d(o,p).upper$ and $d(o,p).lower$ are the upper and lower bounds of the distance between *o* and *p*, then

$$Max\{d(o,p).lower, k-distance(o).lower\} \le reach-dist(o,p) \quad (8)$$

$$reach-dist(o,p) \le Max\{d(o,p).upper, k-distance(o).upper\} \quad (9)$$

Proof According to definition 3, the reachable distance from point *o* to point *p* is the maximum of the *k*-th distance of *o* and the distance between *o* and *p*. Then the reachable distance from point *o* to point *p* must be greater than or equal to the maximum value of the lower bound of the *k*-th distance of *o* and the lower bound of the distance between *o* and *p*, and less than or equal to the maximum value of the upper bound of the *k*-th distance of *o* and the upper bound of the distance between *o* and *p*, then the theorem is proved.

According to the above theorem, as long as the distance between points and the upper and lower bounds of *k* distance of each point are calculated, the upper and lower bounds of *LOF* can be easily obtained.

Definition 6(micro-cluster *MC*, Micro-Clusters) Given a set of data $p_1, p_2, \ldots, p_n$, these data in a subspace is *MC*(*n*,*up*,*low*,*c*), where *n* is the number of points, *up* and *low* are the lower left corner and the upper right corner which can contain all data points. up is composed of the maximum value of each dimension of all points in the micro-cluster, *low* is the minimum value of each dimension of all points in the micro cluster, *c* is up and low the center of the two-point connection.

Theorem 3 *MC*(*n*,*up*,*low*,*c*) is a micro-cluster, *p* is a point outside the micro-cluster, then the minimum and maximum distance between point *p* and *MC* is

$$Dist_{Min}(p,MC) = d(p,c) - dp(up,low)/2 \quad (10)$$

$$Dist_{Max}(p,MC) = d(p,c) + dp(up,low)/2 \quad (11)$$

Proof According to definition 8, *up* is composed of the maximum value of each dimension of all points in the micro-cluster, and *low* is composed of the minimum value of each dimension of all points in the micro-cluster. *c* is the line connecting the two centers of *up* and *low*. Taking *c* as the center and the distance between the two points of *up* and *low* as the diameter, we can get that the boundary of the micro-cluster is a circle, and the minimum distance from a point outside the circle to the circle is the distance from the point to the center of the circle minus the radius of the circle. The maximum distance from a point outside the circle to the circle is the distance from the point to the center of the circle plus the radius of the circle. In the same way, the minimum distance from a point outside the micro-cluster to the micro-cluster is the distance from the point to the center of the micro-cluster

minus the radius of the micro-cluster, and the maximum distance is the distance from the point to the center of the micro-cluster adds the radius of the micro-cluster.

As shown in **Fig. 5**, the maximum and minimum distances from *p* to *MC* are shown when *p* is outside the micro-cluster *MC*. It is not expected that the minimum distance will be less than 0 if the circle of *p* is overlapped with that of another one. At this time, the true distance between all points of another micro-cluster and point *p* is calculated, and the minimum distance is the minimum distance from *p* to the other micro-cluster.

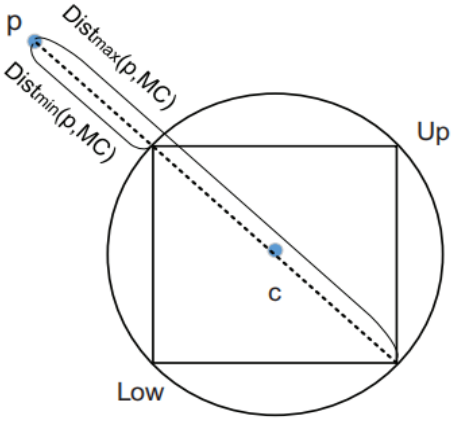


Fig. 5. Maximum and minimum distances between point *p* and *MC*

Definition 7(the maximum and minimum distance between MC_i and MC_j) let $MC_i(n_i, up_i, low_i, c_i)$ and $MC_j(n_j, up_j, low_j, c_j)$ is two micro-clusters, then the minimum and maximum distance between MC_i and MC_j is

$$Dist_{Min}(MC_i, MC_j) = d(c_i, c_j) - dp(up_i, low_i) - d(up_j, low_j)/2 \quad (12)$$

$$Dist_{Max}(MC_i, MC_j) = d(c_i, c_j) + dp(up_i, low_i) + d(up_j, low_j)/2 \quad (13)$$

In **Fig. 6**, the maximum and minimum distances between two clusters are shown. According to the above definition, we can find the upper and lower bounds of the *k*-th distance of a data point with several clusters nearby.

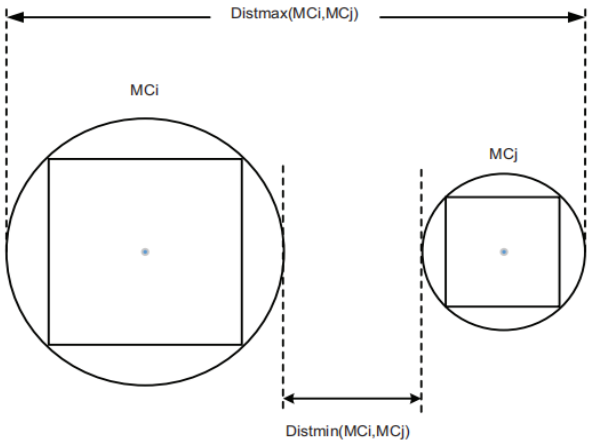


Fig. 6. Maximum and minimum distance between MC_i and MC_j

Corollary 2 let *p* be a data point and *MC*(*n*,*up*,*low*,*c*) be the micro-cluster of point *p*. Let

$MC_1(n_1, up_1, low_1, c_1)$,..., $MC_l(n_l, up_l, low_l, c_l)$ which may contain k nearest neighbors of point p. For the convenience of discussion, the other (n-1) points in are regarded as micro-clusters, that is, point o_i is a micro-cluster, so there are l+n-1 clusters.

1) Let $\{Dist_{min}(p, MC_1), \dots, Dist_{min}(p, MC_{l+n-1})\}$ is increasing order. When $n_1 + \cdots + n_i \geq k$, and $n_1 + \cdots + n_{i-1} < k$, the minimum k-th distance of p is expressed as $Dist_{min}(p, MC_i)$.

2)Let $\{Dist_{max}(p, MC_1), \dots, Dist_{max}(p, MC_{l+n-1})\}$ is increasing order. When $n_1 + \cdots + n_i \geq k$, and $n_1 + \cdots + n_{i-1l} < k$, the maximum k-th distance of p is expressed as $Dist_{max}(p, MC_i)$.

Given a micro-cluster MC, $k_{max} - distance(p)$ is used to represent $Max\{k_{max} - distance(p_1), \dots, k_{max} - distance(p_n)\}$ and $k_{min} - distance(p)$ to represent $Max\{k_{min} - distance(p_1), \dots, k_{min} - distance(p_n)\}$.

Proof In order to find the maximum and minimum of the k-th distance of point p, considering that the nearest neighbor of the k-th distance of p may be included in its own micro-cluster or other nearest neighbor's micro-cluster, then the k-th distance of p is transformed into the distance from point p to the micro-cluster. According to theorem 3, all possible maximum and minimum distances from point p to micro-cluster can be obtained is the maximum value of the k-th distance of p. Similarly, taking the minimum value of the minimum value of all distances from point p to the micro-cluster is the minimum value of the k-th distance of p, then the corollary is proved.

Definition 7(inner reachable distance boundary) the inner reachable distance boundary of a micro-cluster is defined as follows

$$r_{max}(MC) = Max\{d(up, low), k_{max} - distance(MC)\} \quad (14)$$

$$r_{min}(MC) = k_{max} - distance(MC) \quad (15)$$

Intuitively, given any two points p and o in MC, $r_{max}(MC)$ represents the maximum value of reach-distance(p,o) in MC, and $r_{min}(MC)$ represents the minimum value of reach-distance(p,o) within MC.

Definition 9(the outer reachable distance boundary of two micro-clusters) one MC_i with respect to the other MC_j outer reachable distance boundary is defined as follows

$$r_{max}(MC_i, MC_j) = Max\{Dist_{max}(MC_i, MC_j), k_{max} - distance(MC_j)\} \quad (16)$$

$$r_{min}(MC_i, MC_j) = Min\{Dist_{min}(MC_i, MC_j), k_{max} - distance(MC_j)\} \quad (17)$$

Intuitively, given any two points p and o in MC_i and MC_j , $r_{max}(MC_i, MC_j)$ and $r_{min}(MC_i, MC_j)$ represent the maximum and minimum values.

4.1.2 Obtaining minimum value of *LOF*

In order to find the LOF_{min} quickly, we need to find n points with larger LOF, and use a greedy method to quickly obtain these n points, and then calculate the real LOF of these n points to mark LOF_{min}.

Definition 10(node density) d-dimensional space, for the i-th leaf node e in ZH-tree, the ratio of the number of point in e($e.num$) to the volume of e is defined as node density $e.den$.

$$e.den = \frac{e.num}{2^{-id}} \quad (18)$$

According to the definition, the lower the node density, the more likely there are outliers. As shown in **Fig. 2**, the node density of p_{39} subspace is 768, which is larger than node density of p_{44} subspace is 256, so p_{44} is more likely to be an outlier than p_{39}.

Definition 10(point density) In *d*-dimensional space, for a point *p* of leaf node *e*, the number of *k*-nearest neighbors *p.num* in nodes of *p* and the volume ratio *v* of hypercube composed of *p* and *k*-neighbors in nodes are defined as point density *p.den*.

$$p.den = \frac{p.num}{v} \tag{19}$$

In order to obtain *n* sets of approximate outliers quickly, each leaf node of ZH-tree is scanned, and a heap is created to save the leaves with lower density of the first *n* nodes. After obtaining *n* nodes with the highest density, if each point in the node calculates the *LOF*, it may lead to insufficient memory space. Further, the *n* nodes in the heap are refined one by one, and the point density of each point is calculated. Then it is put back into the heap *nn* to update the heap, and *n* sets of approximate outlier points *nn* are obtained. Finally, the real *LOF* of each point is calculated, and the smallest *LOF* is selected and marked as LOF_{min}.

Based on the above definition, the algorithm ZHNO (ZH-tree Nearly Outlier) is shown in Algorithm 1.

Create a heap of size *n* to store *n* sets of approximate outliers named *nn* (step1). The node density of all leaf nodes *e* in ZH-tree is calculated, and the first *n* minimum values are updated and put into heap *nn* (step2-11). The point density of each point in all nodes *e* in the heap is calculated, and the minimum point density of the first *n* heap *nn* is updated (step12-17). The *LOF* of each point in *nn* is calculated and sorted, the smallest *LOF* marked it as LOF_{min} (step18).

Algorithm 1 ZHNO algorithm

Input: ZH-tree
Output: n sets of approximate outliers *nn*,LOF_{min}
1: Create a heap nn of size n;
2: **for** each e in ZH-tree **do**
3: Calculate *e.den*;
4: **if** *nn.size<n then*
5: *nn*.update(*e*);
6: *else*
7: **if** *e.den<nn.last.den* **then**
8: *nn*.update(*e*);
9: **end if**
10: **end if**
11: **end for**
12: **for** each e in *nn* **do**
13: the *p.den* of each data point in *e* is caculated;
14: **if** *p.den* <*nn*.last.den **then**
15: *nn*.update(*p*);
16: **end if**
17: **end for**
18: The *LOF* of each point in *nn* is caculated and sorted, the smallset *LOF* is marked as LOF_{min}

4.2 Filtering approach

To get the upper limit of *LOF* for filtering, the leaf nodes need to be scanned twice. In the first scan $k_{max} - distance(MC)$ and $k_{min} - distance(MC)$ of all clusters are calculated. In the second scan, the upper bound of *LOF* is calculated and filtered and refined. Algorithm 2 describes the DODMD algorithm.

Algorithm 2 DODDMD algorithm

Input: ZH-tree, approximate outliers *nn*,LOF_{min}
Output: n sets of outliers

for each MC_i of e in ZH-tree **do**
  According to inference 2, a set of possible neighbor nodes of MC_i is found;
  $k_{max} - distance(MC_i)$=0;
  $k_{max} - distance(MC_i) = \infty$;
  for each p in MC_i **do**
    Caculate $k_{max} - distance(p)$, $k_{min} - distance(p)$;
    if $k_{max} - distance(p) > k_{max} - distance(MC_i)$ **then**
      $k_{max} - distance(MC_i)$=$k_{max} - distance(p)$;
    end if
    if $k_{min} - distance(p) < k_{max} - distance(MC_i)$ **then**
      $k_{min} - distance(MC_i)$=$k_{min} - distance(p)$;
    end if
  end for
end for
for each MC_i of e in ZH-tree **do**
  Using definition 10, caculate inner $r_{max}(MC_i)$, $r_{min}(MC_i)$;
  $LOF(MC_i).upper = r_{max}(MC_i)/\ r_{min}(MC_i)$;
  Using definition 11, caculate outer $r_{max}(MC_i)$, $r_{min}(MC_i)$;
  $LOF(MC_i).upper = r_{max}(MC_i)/\ r_{min}(MC_i)$;
  if $LOF(MC_i).upper < r_{max}(MC_i)/\ r_{min}(MC_i)$ **then**
    $LOF(MC_i).upper = r_{max}(MC_i)/\ r_{min}(MC_i)$;
  end if
  if $LOF(MC_i).upper > LOF_{min}$ **then**
    The *LOF* of each point in LOF_{min} is caculated and *nn* is updated;
    $LOF_{min} = nn.last().LOF$;
  end if
end for

Next, the time complexity of DODMD algorithm is analyzed. Given *d*-dimensional dataset *D*, the data is divided into *l* clusters. DODMD algorithm is divided into two steps. In the first step, *n* data points are selected and real *LOF* is calculated, and threshold LOF_{min} is marked with time complexity $O(n \times d \times |D|)$. After obtaining LOF_{min}, the second step of filtering and thinning requires scanning the micro-clusters twice. The time complexity of the first scan is $O(l \times d \times |D|)$. In the second scan LOF_{min} should be used to filter. Assuming that there are *R* points that cannot be filtered, the real *LOF* of these *R* points should be calculated, and the time complexity is y $O(R \times d \times |D|)$. Therefore, the time complexity of DODMD is y $O((l + n + R) \times d \times |D|)$.

5. Experimental Results and Analysis

In order to better verify the performance of the proposed algorithm, artificial datasets and real datasets are used to verify. The experimental environment is Intel Core i7 975h 2.6GHz, 8GB memory, 500GB hard disk and windows 10 operating system. All algorithms are implemented in Java language. The algorithms in real datasets comparison are LDOF [24] and SimplifiedLOF [25], distance-based algorithms KNNDD [26] and KNNSOS [27], and the comparison algorithms of synthetic datasets are RandomLOF [28] and SimplifiedLOF. The following will introduce the source of real and synthetic datasets and analysis of their experimental results in detail.

5.1 Analysis of experimental results on real datasets

This section will verify the DODMD algorithm on the real dataset, using java language to write several existing density-based and distance-based outlier detection algorithms for comparison, and compare the accuracy of density-based method and distance-based method. Density-based algorithms for comparison include LDOF and SimplifiedLOF , distance-based algorithms KNNDD and KNNSOS , and the experimental parameter k is 10.

5.1.1 Dataset description

The real datasets used in this section are all from UCI machine learning database [29]. These data sets have been used in previous outlier detection algorithms. Some datasets contain non numeric attributes, which need to be deleted before normalization.

The datasets used in this section are shown in **Table 2**, including the number of data, the number of attributes and the proportion of exception objects in each dataset. The data in these datasets contain multiple class tags, and the data are classified according to these class tags. According to the definition of outlier, we regard the class with more instances as normal class and the class with fewer instances as exception class. If a small number of classes already exist in the dataset, these very few classes are considered outliers and the rest are normal points. The following is a brief introduction to these data sets, the breast cancer diagnosis data set (Wdbc) is from the Wisconsin Hospital, which is calculated from the digital image of the fine needle extraction of the breast mass, including the composition of the nucleus existing in the fine needle extraction. There are two types of data in the dataset, one is malignant, the other is benign. We define malignant as abnormal and benign as normal. The glass dataset (Glass) is from the United States forensic service. The classification of glass types is motivated by criminal search. In a crime scene, if the remaining glass can be identified, it can be considered as evidence. There are six types of glass in this dataset. The sixth type of tableware is considered to be abnormal and the remaining five are regarded as normal. Ecoli is the E.coli dataset, in which there are seven classes. The class "omL", class "imL" and class "imS" are defined as abnormal classes, and the rest are regarded as normal classes. Abalone is abalone data set, which is obtained by measuring and predicting the age of abalone by physical means. It contains 29 classes. We define class "1", class "5", class "14" and class "29" as abnormal classes, and the remaining 25 classes form a normal class.

Table 2. Data set

Dataset	Number of instances	Number of attributes	Ratio (%)
Wdbc	569	30	37.3
Glass	214	9	4.2
Ecoli	336	7	2.7
Abalone	4177	8	5.8

5.1.2 AUC performance analysis

This section uses AUC as the evaluation index to analyze the performance of DODMD, LDOF, SimplifiedLOF, KNNDD and KNNSOS. The higher the AUC value, the better the algorithm performance. These algorithms are run in Wdbs, Glass, Ecoli and Abalone. The results of the experiments are shown in **Table 3**.

Table 3. AUC performance table

Dataset	DODMD	LDOF	SimplifieLOF	KNNDD	KNNSOS
Wdbc	0.6101	0.5487	0.6199	0.5749	0.5238
Glass	0.7827	0.7170	0.8135	0.8021	0.5585
Ecoli	0.8603	0.8124	0.8559	0.8235	0.5416
Abalone	0.5620	0.5797	0.5558	0.5575	0.5462

According to **Table 3**, it can be seen that the AUC performance of DODMD algorithm and SimplifiedLOF algorithm is similar, and the performance is relatively stable on most data sets, while the performance of LDOF algorithm is only higher on Abalone data sets, which shows that LDOF algorithm only performs well on individual data sets, but not very stable on most data sets. This is because the performance of LDOF depends on data sets. It can't be as applicable as DODMD algorithm and SimplifiedLOF algorithm. Compared with KNNDD and KNNSOS, the AUC performance of the two distance-based algorithms is much lower than that of the density-based algorithm, and they are unstable. They only perform better on individual data sets. Therefore, it shows that the density-based method is better than the distance-based method.

5.1.3 ROC curve analysis

In **Fig. 7** (a), (b), (c) and (d) shows the ROC curves of DODMD, LDOF, KNNDD and KNNSOS on the datasets of Wdbc, Glass, Ecoli and Abalone. The closer the ROC curve (0,1) is, the better the performance of the algorithm is. The curves of DODMD algorithm and SimplifiedLOF algorithm are very close, which shows that the performance of the two algorithms is similar. In each data set, the ROC curves of LDOF algorithm are mostly located under the DODMD algorithm and SimplifiedLOF algorithm, which shows that the performance of LDOF algorithm is inferior to the two algorithms. Generally speaking, the performance of DODMD algorithm is better than that of LDOF algorithm, and is not weaker than SimplifiedLOF. At the same time, several density-based algorithms are almost all above the distance-based algorithm, which shows that the density-based method is better than the distance-based method.

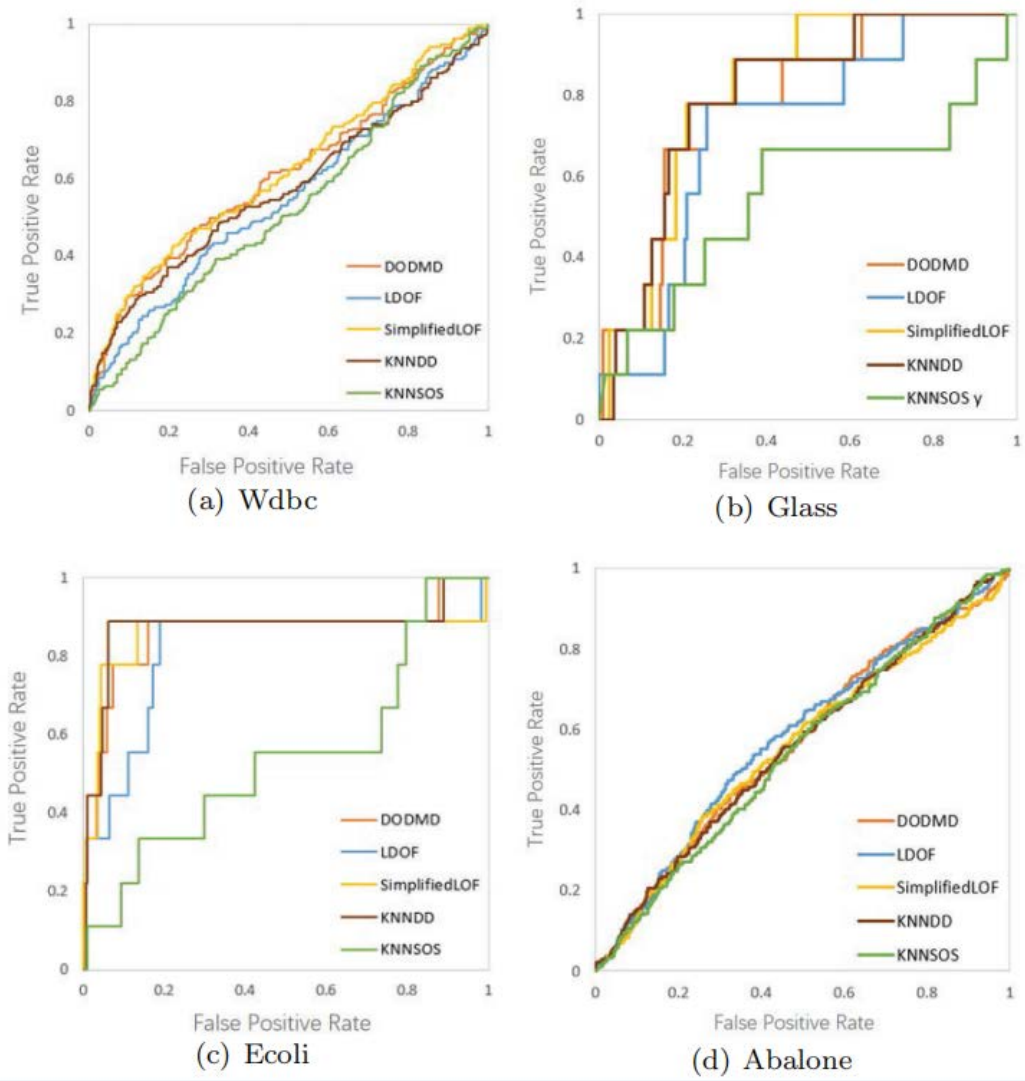


Fig. 7. ROC curves on datasets Wdbc,Glass,Ecoli and Abalone

5.2 Analysis of experimental results on synthetic datasets

In the field of machine learning and data mining, real data sets cannot meet the needs of experiments. Therefore, many researches are devoted to using synthetic datasets to verify the time performance of the algorithm. In this section, RandomLOF and SimplifiedLOF are used for comparative experiments.

5.2.1 Dataset description

In this experiment, the mechanism of abnormal objects is different from that of normal objects. Therefore, two clusters with (2,2) and (-2, -2) centers are generated in the data space with Gaussian distribution. These data are considered as normal objects. And 100 data points are generated in the way of uniform distribution in the whole data space. If the density of some data objects is significantly different from that of the inner objects in the neighborhood, they are considered as outliers. The overall distribution of data is shown in **Fig. 8**. In order to

make the experiment more accurate and credible, the data of dimension (10,12,14,16,18,20) and data scale (5 × 103,10 × 103,15 × 103,20 × 103) were generated respectively.
The default values and related variation ranges of experimental related variables are shown in **Table 4**.

Table 4. Parameters table

Parameter	Default value	Variation range
Numbers($\times 10^3$)	10	5-12
Data dimension	10	10-20
Parameter *k*	10	5-25

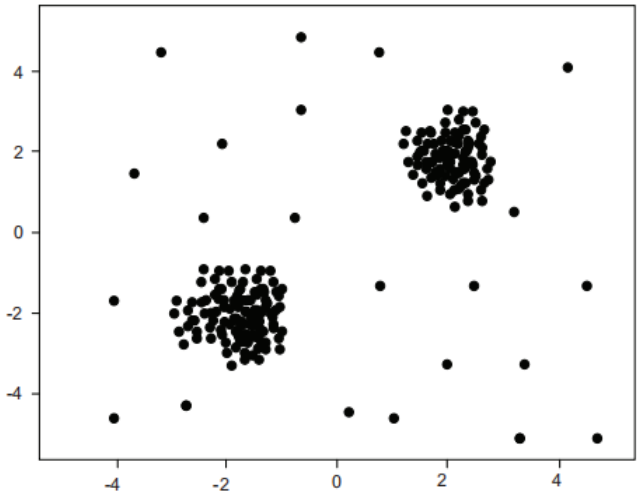


Fig. 8. Distribution of artificially synthesized data

5.2.2 Time efficiency analysis

The time efficiency of the algorithm is also an important standard to evaluate the advantages and disadvantages of the algorithm. This section compares the time efficiency of DODMD, RandomLOF and SimplifiedLOF, and compares different dimensions and parameters.

In **Fig. 9** we test the influence of data set size on the three algorithms using 5 × 103,10 × 103,15 × 103,20 × 103 data respectively. With the increase of the size of the data set, the time of the two algorithms becomes longer, but the overall performance of DODMD is better than RandomLOF and SimplifiedLOF. Although more data points need to be calculated when calculating *k*-nearest neighbor, DODMD has more efficiency than RandomLOF and SimplifiedLOF because of its index structure. In addition, DODMD can effectively filter the data objects in the point cluster in the filtering step, thus reducing a large number of query operations.

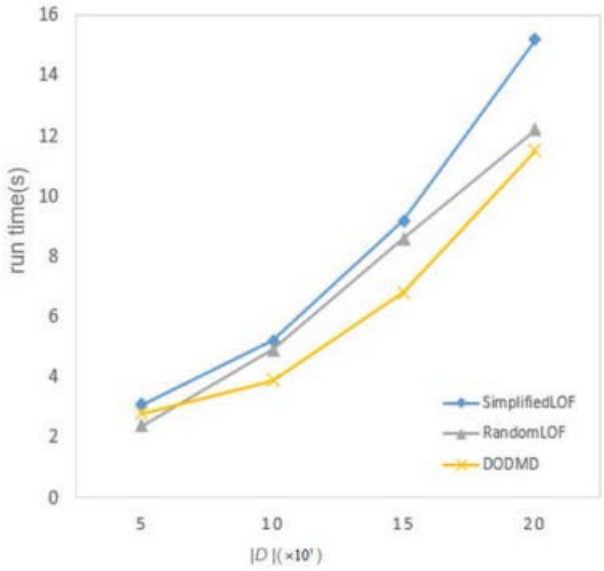


Fig. 9. Influence of data size

In **Fig. 10**, three algorithms are tested. With the influence of dimension on algorithm performance, the running time of three algorithms becomes longer with the increase of *d*. This is because with the growth of *d*, it takes longer to search for neighbors. The running time of MBDOB and RandomLOF is similar, and both of them have shorter running time than SimplifiedLOF. DODMD algorithm has less running time, mainly by virtue of ZH-tree index structure to effectively reduce the dimension characteristics, so that the algorithm performance is higher.

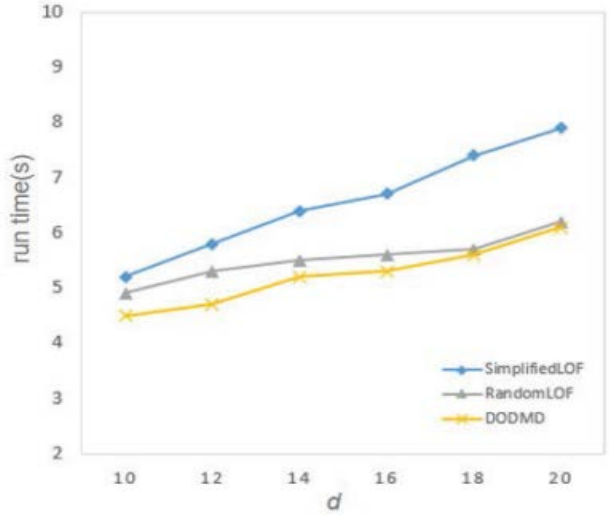


Fig. 10. Influence of dimensions

Fig. 11 describes the influence of parameter *k* on the three algorithms. The increasing of parameter *k* leads to the increase of running time of the three algorithms. It can be seen from the curve that the DODMD processing time is better than the other two algorithms, because with the constant increase of parameter *k*, the number of searches needed to query neighbors

increases. However, DODMD algorithm can effectively prune, so the processing time is less. However, if the value of parameter *k* is too large, it will affect the data partition of ZH-tree index, resulting in excessive data of nodes. It will increase the amount of calculation between data objects in the calculation node, resulting in low efficiency of the algorithm.

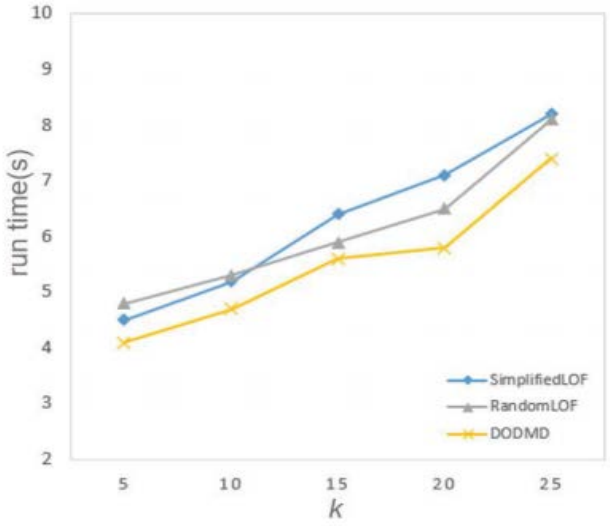


Fig. 11. Influence of parameter *k*

6. Conclusion

Outlier detection method should start from the definition of outlier, use efficient method to detect outlier, and finally verify the algorithm through experiments. The main work of this paper includes the following aspects.

(1) The index structure of ZH-tree is adopted and improved, the concept of micro-cluster is introduced into the index to achieve the purpose of filtering in the process of outlier detection. The clustering attribute of ZH-tree index can effectively help search the neighbors of objects, and its hierarchical structure can effectively prune the space and reduce unnecessary calculation.

(2) A new outlier detection method, DODMD algorithm, is proposed. The idea of DODMD algorithm is to use the upper limit of outlier value of each micro-cluster is calculated, and the threshold is used to filter, which reduces unnecessary calculation of data and makes the algorithm more efficient.

(3) Finally, Experiments on different data sets verify the performance of DODMD algorithm.

Acknowledgement

This work is supported by the National Natural Science Foundation of China (61602076, 61702072, 62002039, 61976032), the China Postdoctoral Science Foundation funded projects (2017M611211, 2017M621122, 2019M661077), the Natural Science Foundation of Liaoning Province (20180540003), CERNET Innovation Project (NGII20190902), fundamental research funds for Dalian Maritime University (3132022634).

References

[1] Xue A, Yao L, Ju S, et al, "Survey of outlier mining methods," *Computer science*, vol. 35, no. 11, pp.13-18, Oct. 2008. Article(CrossRefLink)

[2] Zhao X, Cui W, Wu Y, et al, "Outlier Interpretation on Multi-dimensional Data via Visual Analytic," in *Proc. of Computer Graphics Forum*, vol. 38, no.3, pp.213-214, Oct. 2019. Article(CrossRefLink)

[3] Han J, Micheline K, "Data mining: concepts and techniques," *data mining concepts models methods & algorithms second edition*, vol. 5, no. 4, pp.1-18, Jan. 2006.

[4] Jin L, Chen J, Zhang X, "An Outlier Fuzzy Detection Method Using Fuzzy Set Theory," *IEEE Access*, vol. 357, no. 99, pp.59321-59322, 2019. Article(CrossRefLink)

[5] Mengliang Shao, Deyu Qi, Huili Xue, "Big data outlier detection model based on improved density peak algorithm," *J. Intell. Fuzzy Syst.*, vol. 40, pp.6185-6194, May. 2021. Article(CrossRefLink)

[6] Yan Gao, "Deep Model-Based Semi-Supervised Learning Way for Outlier Detection in Wireless Capsule Endoscopy Images," *IEEE Access*, vol. 8, pp.81621-81632, Oct. 2020. Article(CrossRefLink)

[7] Gustavo Henrique Orair, Carlos H. C. Teixeira, "Distance-Based Outlier Detection: Consolidation and Renewed Bearing," *Proceedings of the VLDB Endowment*, vol. 3, no.2, pp.1469-1480, Oct. 2010. Article(CrossRefLink)

[8] Bo Tang, Haibo He, "A Local Density-Based Approach for Local Outlier Detection," *CoRR*, Mar. 2016. Article(CrossRefLink)

[9] Bay S D, Schwabacher M, "Mining distance-based outliers in near linear time with randomization and a simple pruning rule," in *Proc. of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Washington, USA, pp.29-38, Aug. 2003. Article(CrossRefLink)

[10] Tian Xia, "Improving the R*-tree with outlier handling techniques," in *Proc. of International Workshop on Geographic Information Systems*, Bremen, Germany, pp.125-134, Nov. 2005.Article(CrossRefLink)

[11] Xiangmin Zhou, Guoren Wang, Jeffrey Xu Yu, "M+-tree : A New Dynamical Multidimensional Index for Metric Spaces," in *Proc. of Fourteenth Australasian Database Conference (ADC2003)*, vol. 17, pp.161-168, Feb. 2003 Article(CrossRefLink)

[12] Lin Mei, Fengli Zhang, et al, "A Distributed Density-based Outlier Detection Algorithm on Big Data," *Int. J. Netw. Secur.*, vol.22, no.5, pp. 775-781, Jan. 2020. Article(CrossRefLink)

[13] Papadimitriou S, Kitagawa H, Gibbons P B, et al, "LOCI: Fast Outlier Detection Using the Local Correlation Integral," in *Proc. of the 19th International Conference on Data Engineering*, March, Bangalore, India, pp.315-326, Mar. 2003. Article(CrossRefLink)

[14] Chongsheng Zhang, Zhongbo Wu, Bo Qu, Hong Chen, "Mining Top-n Local Outliers in Constrained Spatial Networks," in *Proc. of the 4th international conference on Advanced Data Mining and Applications*, Springer, vol. 5139, no. 008, pp.725-732, Oct. 2008. Article(CrossRefLink)

[15] Hu C, Qin X, "A Density-Based Local Outlier Detecting Algorithm," *Computer research and development*, vol.47, no.12, pp. 2110-2116, Dec. 2010. Article(CrossRefLink)

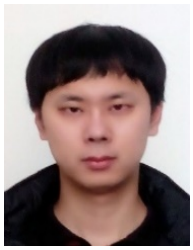
[16] He Z, Xu X, Deng S, "Discovering Cluster Based Local Outliers," *Pattern Recognition Letters*, vol.24, no. 9, pp. 1641-1650, 2003. Article(CrossRefLink)

[17] Huawen Liu. Xuelong, Jiuyong Li, "Efficient Outlier Detection for High-Dimensional Data," *IEEE Trans. Syst. Man Cybern. Syst.*, vol.48, no.12, pp. 2451-2461, 2018. Article(CrossRefLink)

[18] Matsumoto T, Hung E, Edward, Yiu L, et al, "Parallel outlier detection on uncertain data for GPUs," *Distributed and Parallel Databases*, vol.33, no. 3, pp.417-447, 2015. Article(CrossRefLink)

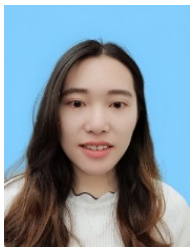
[19] Zhong Y, Wang x, Bai M, et al, "FODU: fast outlier detection method in uncertain data sets," *Computer engineering and applications*, vol.55, no. 19, pp.105-114, 2019. Article(CrossRefLink)

[20] Mostafa Rahmani, George K. Atia, "Randomized Robust Subspace Recovery and Outlier Detection for High Dimensional Data Matrices," *IEEE Trans. Signal Process.*, vol. 65, no. 6, pp.1580-1594, May. 2017. Article(CrossRefLink)
[21] Angiuulli F, Pizzuti C, "Outlier mining in large high-dimensional data sets," *IEEE Transactions on Knowledge & Data Engineering*, vol. 17, no. 2, pp.203-215, Feb. 2005. Article(CrossRefLink)
[22] Kriegel H P, Schubert M, Zimek A, "Angle-based outlier detection in high-dimensional data," in *Proc. of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Las Vegas, Nevada, USA, pp. 444-452, Aug. 2008. Article(CrossRefLink)
[23] Wang X, Shen D, Bai M, et al, "An Efficient Algorithm for Distributed Outlier Detection in Large Multi-Dimensional Dataset," *Journal of Computer Science and Technology*, vol. 30, no. 6, pp.1233-1248, Oct. 2015. Article(CrossRefLink)
[24] Zhang K, Hutter M, Jin H, "A New Local Distance-Based Outlier Detection Approach for Scattered Real-World Data," in *Proc. of Pacific- Conference on Knowledge Discovery & Data Mining*, Bangkok, Thailand, pp.813-822, Apr. 2009. Article(CrossRefLink)
[25] Schubert E, Zimek A, Kriegel H P, "Local outlier detection reconsidered: a generalized view on locality with applications to spatial, video, and network outlier detection," *Data Mining and Knowledge Discovery*, vol. 28, no. 1, pp.190-237, 2014. Article(CrossRefLink)
[26] De Ridder D, "An Experimental Comparison of One-class Classification Methods," in *Proc. of the 4th Annual Conference of the Advanced School for Computing and Imaging*, Delft, NL, pp. 121-130, 1998. Article(CrossRefLink)
[27] Schubert E, Gertz M, "Intrinsic t-Stochastic Neighbor Embedding for Visualization and Outlier Detection," in *Proc. of International Conference on Similarity Search & Applications*, Munich, Germany, pp.188-203, Oct. 2017. Article(CrossRefLink)
[28] Nguyen M Q, Omiecinski E, Mark L, et al, "A Fast Randomized Method for Local Density-Based Outlier Detection in High Dimensional Data," *Data Warehousing and Knowledge Discovery*, Bilbao, Spain, pp.215-226, Aug. 2010. Article(CrossRefLink)
[29] Asuncion A, Newman D, "UCI machine learning repository," 2015. Article(CrossRefLink)



XITE WANG received the Ph.D. degree in computer science and technology from Northeastern University, China, in 2015. He is currently an Associate Professor with Dalian Maritime University. His research interests include big-data management and parallel data processing.

ZHIXIN CAO received the BS degree in Dalian Maritime University, China, in 2019. His research interests include big-data management and parallel data processing.



RONGJUAN ZHAN received the BE degree in measurement and control technology from Liaoning University of Petrochemical Technology, China, in 2020. She is currently working toward the BS degree at Dalian Maritime University and her research interests include big-data management.

MEI BAI received the Ph.D. degree in computer science and technology from Northeastern University, China, in 2016. She is currently an Associate Professor with Dalian Maritime University. Her research interests include data management, cloud computing, and query processing.

QIAN MA received her Ph.D. degree in computer software and theory from Northeastern Univer- sity, China, in 2019. Currently, she is a post doctor as a staff at Dalian Maritime University, China. Her research interest is data management and data cleaning.

Guanyu Li received the B.S. and M.S. degree from Dalian Maritime University, Dalian, China, in 1985, 1993. He is currently a Professor and also a Ph.D. Supervisor with the Faculty of Information Science and Technology, Dalian Maritime University. His major research interests include machine learning, intelligent information, etc.