Received October 30, 2018, accepted November 28, 2018, date of publication December 11, 2018, date of current version January 4, 2019.

Digital Object Identifier 10.1109/ACCESS.2018.2886197

An Efficient Density-Based Local Outlier Detection Approach for Scattered Data

SHUBIN SU, LIMIN XIAO, (Member, IEEE), LI RUAN, FEI GU, SHUPAN LI, ZHAOKAI WANG, AND RONGBIN XU

[1]State Key Laboratory of Software Development Environment, Beihang University, Beijing 100191, China
[2]School of Computer Science and Engineering, Beihang University, Beijing 100191, China

Corresponding author: Limin Xiao (xiaolm@buaa.edu.cn)

This work was supported in part by the National Key R&D Program of China under Grant 2017YFB1010000, in part by the National Natural Science Foundation of China under Grant 61772053, in part by the State Key Laboratory of Software Development Environment under Grant SKLSDE-2018ZX-10, and in part by the Science Challenge Project under Grant TZ2016002.

ABSTRACT After the local outlier factor was first proposed, there is a large family of local outlier detection approaches derived from it. Since the existing approaches only focus on the extent of overall separation between an object and its neighbors, and ignore the degree of dispersion between them, the precision of these approaches will be affected by various degrees in the scattered datasets. In addition, the outlier data occupy a relatively small amount in the dataset, but the existing approaches need to perform local outlier factor calculation on all data during the outlier detection, which greatly reduces the efficiency of the algorithms. In this paper, we redefine a local outlier factor called local deviation coefficient (LDC) by taking full advantage of the distribution of the object and its neighbors. And then, we propose a safe non-outlier objects elimination approach named as rough clustering based on multi-level queries (RCMLQ) to preprocess the datasets to eliminate the non-outlier objects to the utmost. Finally, an efficient local outlier detection approach named as efficient density-based local outlier detection for scattered data (E2DLOS) is proposed based on the LDC and RCMLQ. The RCMLQ greatly reduces the amount of data that needs to be quantified for local outlier factor and the LDC is more sensitive to the degree of anomaly of the scattered datasets, and so the E2DLOS improves the existing local outlier detection approaches in time efficiency and detection accuracy. Experiments show that the LDC can better reflect the true abnormal situations of the data for the scattered datasets. And the RCMLQ can be used in parallel with the traditional methods of improving the efficiency of the nearest neighbor search, which can further improve the efficiency of the E2DLOS algorithm by about 16%.

INDEX TERMS Outlier detection, local outlier factor, neighborhood variance, rough clustering, scattered dataset.

I. INTRODUCTION

Outliers are a special subset of the dataset that are significantly distinct from the other data and have a very small percentage, but they often contain real and unimaginable knowledge. Outlier detection is devoted to discovering outliers from a large amount of information data. It is an important part of data mining and a very active branch of information science, which has attracted the attention of researchers in many disciplines including data mining, statistics, and information theory [2], [3]. Now, outlier detection is considered as a critical task in many practical applications, such as network intrusion detection [4]–[8], fraud detection [9]–[11], industrial damage detection [12]–[15] and health care monitoring [16]–[18].

The outliers originally existed as the by-product of many clustering algorithms, and this directly inspired many scholars to improve some clustering algorithms to obtain the early major outlier detection algorithms. Since the clustering algorithms and some other early studies of outlier detection are based on the entire dataset, the early outlier detection methods obtained a set of global outliers. However, due to the complexity and variability of the real world, the instability of data collection and transmission technology, etc., the datasets obtained in practical applications are often incomplete in terms of time and space. Moreover, we only care about the change of things in a local scope in many scenarios. Therefore, the outliers obtained at this time are local outliers. Compared with global outliers, the local outliers only compare the data with one of its smaller nearest neighbors and not the entire dataset. Schubert *et al.* [19] first defined the local outliers factor (LOF), and then for different datasets or application scenarios, many scholars have improved and extended the LOF algorithm from two aspects, i.e., the determination of the local neighbors of the test object and the comparison between the test object and its neighbors, and many of these algorithms have achieved good results. However, the existing local outlier detection approaches only focus on the extent to which the object deviates from its neighbors as a whole when calculating the local outlier factor, and ignore the degree of dispersion between them. Therefore, when these algorithms perform outlier detection on scattered datasets, their accuracy will be seriously affected.

In addition, the existing local outlier detection methods need to perform nearest neighbor search for all objects in the dataset when calculating local outlier factor, which is a very time-consuming process and its time complexity can reach $O(n^2)$, thus the existing detection algorithms are not suitable for local outlier detection in large-scale datasets. Now, there are two main ways to improve the efficiency of the algorithms. One is to simplify the calculation of the local outlier factor. However, this method is limited in the degree of improvement and tends to reduce the accuracy of the algorithms [19], [20]. The other one is to use a various data structures (such as R*-tree [21], KD-tree [22], Cover tree [23], M-tree [24], R-tree [25] etc.) for efficient nearest neighbor search. Now that the research results of data structure are rich and mature, this method is widely used and has achieved a very good performance. According to the definition of outliers, it can be known that the outlier data occupies only a very amount of the dataset relative to normal data [26]. Therefore, we can eliminate the safe internal objects to the maximum extent by preprocessing the dataset, which can minimize the amount of data for calculating the local outlier factor and effectively improve the time efficiency of the outlier detection algorithms. It is obvious that this method can also be applied in parallel with the method of improving the efficiency of nearest neighbor search by adopting a data structure, so this paper focuses on further improving the efficiency of the algorithms by eliminating non-outliers.

Aiming at the limitation of the traditional local outlier detection algorithms, we first redefine a local outlier factor called Local Deviation Coefficient (LDC) by using the expectation and variance of the distance between the object and its neighbors. And then a safe non-outlier objects elimination approach named as Rough Clustering based on Multi-Level Queries (RCMLQ) is studied to preprocess the datasets, which greatly reduces the number of data that needed to quantify the local outlier factor and thus further improves the efficiency of the local outlier detection from a new direction. Finally, we propose an efficient local outlier detection algorithm named as Efficient Density-based Local Outlier Detection for Scattered data (E2DLOS). Expectation and variance are two important indicators that represent the distribution of datasets. Therefore, LDC can better reflect the degree of data dispersion, which can make the local outlier factor quantized by LDC reflects the abnormality of scattered data more fully. And in the process of clustering, the RCMLQ algorithm only needs to traverse the dataset once and perform the nearest neighbor search on a small amount of the dataset. Therefore the RCMLQ algorithm is very efficient. Moreover, the amount of data that is needed to calculate the local outlier factor will be greatly reduced after the dataset is preprocessed by the RCMLQ algorithm, and the RCMLQ can also be used in parallel with the traditional method of improving the nearest neighbor search efficiency to reduce the time complexity of the algorithms, so the E2DLOS algorithm further improves the time efficiency of the local outlier detection from a new direction. In this way, we improve the local outlier detection algorithm from both efficiency and accuracy. In conclusion, we extend the ecosystem of local outlier detection approaches from a new perspective. Finally, the effectiveness and superiority of the proposed approaches are verified by both synthetic and real data. This article is substantially an extension of our early conference paper [1]. In the previous conference paper, a new local outlier coefficient (LDC) was defined for the low accuracy of local outlier detection in scattered datasets. On this basis, a dataset preprocessing method (RCMLQ) is proposed to improve the time efficiency of the local outlier detection algorithm. In this way, we extend the outlier detection approach from the accuracy and time efficiency.

The rest of the paper is organized as follows. Section 2 discusses the related works. Section 3 describes the E2DLOS approach. Section 4 presents the experimental study and the result of experiment evaluation. Finally, the section 5 sates our conclusions.

II. RELATER WORK

A. LOCAL OUTLIER DETECTION APPROACHES

Outliers are also called abnormal points, novel points, deviation points, noise points, etc., which have different definitions in different fields. However, the definition by Hawkings [26] is a good embodiment of the essence of the outliers: "an outlier is an observation point, which deviates from the other observation points so much that it is caused by the suspicion that it is generated by different mechanisms" [13]. Because this definition is from the perspective of the entire dataset, the outliers here refer to global outliers. Relative to the global outliers, the study of local outliers is much later. Breuing *et al.* [27] first defined the concept of local outlier factor and proposed the LOF algorithm for local outlier detection.

Definition 1: Local outliers are the objects that distinguish or deviate from their neighborhood in the dataset.

Density-based outlier detection is developed on the basis of the definition of distance. The main idea of density-based outlier detection is to first define the "density" of the object according to the two parameters of the distance among the objects and the number of objects within a given range, and then use the density to quantify the degree of outlier of each object. The LOF is a density-based parameter that was first introduced by Breuing to measure the degree of local deviation of the objects.

The LOF algorithm can be effectively applied to non-uniform density datasets for outlier detection, and it has been successfully applied in many fields. In addition, LOF is a seminal algorithm, and many scholars have improved or extended it for different datasets or some special application scenarios. And a series of local outlier detection algorithms have been proposed on this basis. The more representative algorithms at present are as follows: LSC (local sparse coefficient) algorithm [28], MDEF (multi granularity deviation factor) algorithm [29], INFLOF (outlier factor based on the symmetric close relation) algorithm [30], COF (connectivity-based outlier factor) algorithm [31], SimplifiedLOF algorithm [19], LoOP (local outlier probabilities) algorithm [32], LDOF(local distance-based outlier factor) algorithm [20], LDF (local density factor) algorithm [33], KDEOS (kernel density estimation outlier score) algorithm [34]. The existing local outlier detection algorithms generally calculate the local outlier factor of all data in the dataset. However, these existing algorithms are less efficient when calculating the local outlier factor of the objects for the reason that the determination of the nearest neighborhood of each object needs to traverse the dataset once. Therefore, the scholars mainly reduce the time complexity of the algorithms by improving the efficiency of nearest neighbor search, and the determination methods of the neighborhood of the objects become a major difference of the existing local outlier detection algorithms. Nowadays, improving the efficiency of nearest neighbor search is mainly using various classic data structures (such as various tree structures) [35]–[37]. Specifically, we will analyze these algorithms in detail in the next section.

B. LOCAL OUTLIER FACTOR QUANTIZATION APPROACHES

The other major difference among the existing local outlier detection algorithms is the calculation of local outlier factor. LOF [24] is the first local outlier factor defined in the academia. For any test object o, the LOF first determines its k nearest neighbor (kNN) set, k-distance, reachable distance ($reach-dist_k$) and reachable density (lrd) by the given threshold number k of nearest neighbor objects, and then the ratio of the average lrd of the kNN set of o to the lrd of o is used to represent the magnitude of the LOF value. LOF can well solve the problem of mining local outliers of the datasets with uneven density distribution. However, the algorithm has the drawback that the detection result is affected by the k value, the time complexity is high, and the accuracy of detection results is relatively low for the distributed sparse datasets.

The proposal of LOF is a groundbreaking work, which leads many scholars to continue in-depth research in this field and has achieved a series of results. Schubert *et al.* [19] directly replaces the reachable distance of LOF with k-nearest neighbor distance, thus the more simple density estimation is obtained. Then, the SimplifiedLOF algorithm is proposed based on this, but it does not improve the time complexity of the LOF algorithm. Tang *et al.* [31] improved the density estimation of the SimplifiedLOF algorithm and proposed the COF algorithm. The COF algorithm first uses the minimum spanning tree (MST) to determine the neighborhood of the object and the average connection distance between the object and its neighbors, and the COF is then represented by the ratio of the average connection distance of the object to its neighbors. The COF algorithm overcomes the shortcomings of the LOF algorithm cannot effectively measure the outliers of sequence data and low-density datasets. However, LOF needs to establish a MST with a time complexity of $O(k^2)$, so its time complexity is higher than the LOF algorithm. Jin *et al.* [30] improved the definition of the neighborhood of the SimplifiedLOF algorithm and proposed the INFLOF algorithm. The INFLOF algorithm uses the union of the object's kNN and RkNN as the k-neighborhood. But the INFLOF algorithm needs to calculate the RkNN of the object with a time complexity of $O(k^2)$. Kriegel *et al.* [32] studied a more robust local density estimation function than the SimplifiedLOF algorithm and proposed the LoOP algorithm. The LoOP algorithm directly uses the reciprocal of the quadratic average distance between the object and its neighbors as the local density of the test object, and its time complexity is similar to LOF.

In addition, Zhang *et al.* [20] redefined the local outlier factor and proposed the LDOF algorithm. For a test object, the LDF algorithm first calculates the average distance between it and its neighbors, then calculates the average of the distance between all its neighbors, and finally uses their ratio as the local outlier. Similar to the COF algorithm, the LDOF algorithm also needs to calculate the distance between every pairs of neighbors, and its time complexity is $O(k^2)$. Latecki *et al.* [33] used the reach distance of the LOF algorithm to replace the original distance of the variable-width Gaussian kernel density estimation (KDE), and then used KDE to represent the density estimate of the tested object. Similar to the LOF algorithm, the value of the LDF is defined by comparing the KDE of the object with its neighbor object. The time complexity of the LDF algorithm is the same as that of the LOF algorithm. Moreover, Schubert *et al.* [34] also used KDE to improve the LOF algorithm and proposed the KDEOS algorithm, but unlike the LDF algorithm, they directly use the mathematical properties of KDE. The KDEOS algorithm first normalizes the KDE density of each object to z-scores respect to the KDE density of the kNN set, then assumes that the resulting intermediate score s is a normal distribution and uses the normal cumulative density function to obtain the final KDEOS score. The KDEOS algorithm considers that the density of objects in the dataset satisfies the normally distribution, which may not be applicable in some applications, and its time complexity is $O(k(k+1))$. The MDEF algorithm [29] sets the multi-level neighborhood of the tested object according to different application requirements, and then uses the change of the number of objects in each neighborhood to define the MDEF value.

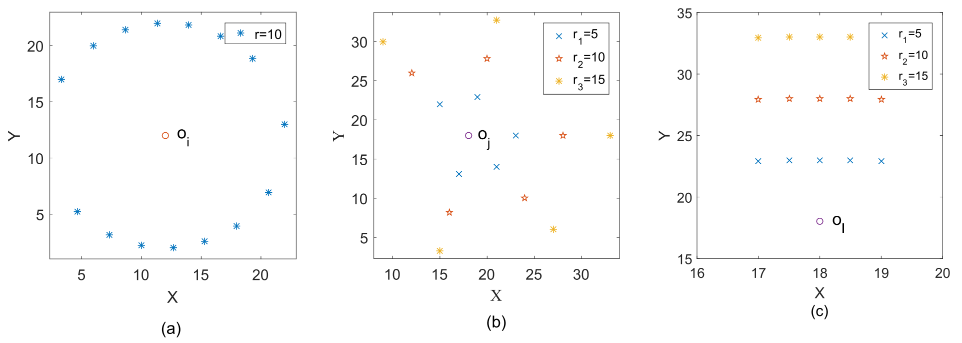


FIGURE 1. The sum of the distances of o_i, o_j, o_l and their respective neighbors are equal.

The MDEF algorithm uses the number of objects to replace the distance between objects, which reduces the time complexity of the outlier detection algorithm. However, the size of the neighborhood is difficult to set accurately, and often requires multiple attempts to achieve better results. Therefore, the detection result of the MDEF algorithm is often directly related to the user's experience. To reduce the time complexity of the LOF algorithm, Kriegel *et al.* [39] proposed the FastABOD algorithm for high-dimensional data sets. The main idea of the FastABOD algorithm is to take each point as the origin, then calculate the angle between it and all its neighbors, and use the weighted variance of these angles as the local outlier. So for each object, the time complexity of the algorithm is $O(k^2)$. A more detailed theoretical analysis of the local outlier detection algorithm can be found in the work of Schubert *et al.* [19], [34], Campos *et al.* [38]. In short, scholars from all walks of life have improved and extended LOF algorithm according to the application needs of different scenarios, and have achieved many impressive scientific research achievements.

III. A NEW LOCAL OUTLIER DETECTION APPROACH BASED ON DATA DISTRIBUTION

A. CALCULATION OF LOCAL OUTLIER FACTOR

From the above analysis, we can find that the existing local outlier detection approaches mainly analyze the extent to which the object deviates from its neighbor objects from a holistic perspective, and ignore the distribution between them. Therefore, when these algorithms perform outlier detection on scattered datasets, their accuracy will be affected to varying degrees. For example, as shown in Fig. 1, the average distances of o_i, o_j and o_l to their respective 15 neighborhood objects are equal, then when $k = 15$, the LOF algorithm, the SimplifiedLOF algorithm, and many of their improved algorithms yield the same local reachability density. Similarly, if the objects have the same quadratic mean distance from their respective neighborhood objects, the local density estimate for the objects obtained by the LoOP algorithm are the same. Obviously, we can see from the Fig. 1 that there is a significant difference in the connectivity between these objects and their respective neighborhood objects, and there is also a clear distinction among the extent to which they deviate from their respective neighborhood objects. Therefore, the accuracy of these algorithms will be affected in the above cases, and we have also proved this in later experiments. In view of the above problems, this paper takes full account for the distribution between the object and its neighbors and redefines the local outliers. And a more detailed analysis of the relevant content of this section can also refer to our previous work [1].

Assuming that the dataset $O = \{o_1, o_2, \cdots, o_n\}$ is composed of n objects, each object $o_i = \{a_1, a_2, \cdots, a_m\}$ $(1 \leq i \leq n)$ is a m-dimensional vector. In order to prevent the over-sized attributes in the data from having a decisive influence on the detection results, we first normalize each dimension of the dataset. The normalized form used in our work is to change the attribute value of the object to [0, 1]. For any dimension i, $A_i = [a_{1i}, a_{2i}, \cdots, a_{ni}](1 \leq i \leq m)$, the normalized result is $f(A_i) = [f(a_{1i}), f(a_{2i}), \cdots, f(a_{ni})]$, and the calculation process of $f(a_{ji})(1 \leq j \leq n)$ is as in EQ.1.

$$f(a_{ji}) = \frac{a_{ji}}{\sum_{j=1}^{n} a_{ji}} \tag{1}$$

Definition 2 (k-Neighborhood of Object o, $N_k(o)$): The set of the k-nearest neighbors of object o in the dataset O, and $N_k(o)$ satisfies the following conditions:

- $|N_k(o)| = k$
- $\forall o_i, o_j$, if $o_i \in N_k(o)$, then $|oo_i| \leq |oo_j|$

Definition 3 (k-Neighborhood Average Distance of Object o, $N_{k-adist}(o)$): The $N_{k-adist}(o)$ equals the average distance from o to all objects in $N_k(o)$:

$$N_{k-adist}(o) = \frac{\sum\limits_{p \in N(o)} |op|}{|N_k(o)|} \tag{2}$$

The size of the expectation can well reflect the overall distribution of the objects in the particular attribute space. As the average of the distance between an object and each of its neighborhood objects, $N_{k-adist}(o)$ can well reflect the extent to which the object o deviates from $N_k(o)$ as a whole. In our work, $|o_i o_j|$ represents the number of objects that $N_k(o)$ contains. In order to improve the accuracy of the detection results, the culling average algorithm [40] is used to eliminate the influence of the extreme distance between the object o and $N_k(o)$ on the calculation of $N_{k-adist}(o)$. Specifically, the distance between the object o and each object in $N_k(o)$ is first sorted in ascending order, and then the maximum and minimum values are eliminated according to the ratio of $\rho\%$. Generally take $\rho = 5 \sim 20$, let $r = \rho\%|N(0)|$, so the formula (2) becomes:

$$N_{k-adist}(o) = \frac{\sum_{i=r+1}^{k-r} |op|}{|N_k(o)| - 2r} \tag{3}$$

It can be known from the characteristics of the local outliers that the distance between the object o and the outliers in $N_k(o)$ are relatively large. After the extremum distance is removed by the equation (3), the influence of the outliers on the detection of the normal object can be avoided. In addition, we normalize the k neighborhood average distance of all data, denoted as:

$$Nn_{k-adist}(o) = \frac{N_{k-adist}(o)}{\sum_{p \in O} N_{k-adist}(p)} \tag{4}$$

Definition 4 (k-Neighborhood Variance of Object o, $N_{k-vari}(o)$): The $N_{k-vari}(o)$ equals the variance of the distance from o to all objects in $N_k(o)$:

$$N_{k-vari}(o) = \frac{\sum_{i=1}^{k} \sqrt{(|oo_i| - N_{k-adist}(o))^2}}{|N_k(o)|} \tag{5}$$

Variance is commonly used to measure the degree of dispersion of a random variable or a set of data in mathematics. Here we use variance to represent the degree of dispersion of the distance between the object and each of its neighborhood objects. Therefore, the $N_{k-vari}(o)$ can well reflect the dispersion between o and $N_k(o)$. Similarly, we normalize the $N_{k-vari}(o)$, recorded as:

$$Nn_{k-vari}(o) = \frac{N_{k-vari}(o)}{\sum_{p \in O} N_{k-vari}(p)} \tag{6}$$

Definition 5 (k-Neighborhood Dispersion of Object o, $N_{k-disp}(o)$): The degree of dispersion of object o with its k neighborhood objects:

$$N_{k-disp}(o) = (Nn_{k-adist}(o) + 1)^{Nn_{k-adist}(o) * Nn_{k-vari}(o)} \tag{7}$$

Expectation and variance in mathematics are two important indicators used to represent the distribution of the datasets or variables. In Equation(4), we normalize the k neighborhood average distance ($N_{k-disp}(o)$) of all data, this will only change the order of magnitude of the $N_{k-disp}(o)$ from the global, and will not change the difference among the $N_{k-disp}(o)$ of the data. It can be concluded in the previous analysis that in order to improve the accuracy of the local outlier detection algorithms, when calculating the local outlier factor of the objects, it is necessary to fully consider not only the overall degree of separation between the object and its neighbors but also the distribution between them. In this paper, we use $N_{k-dist}(o)$ and $N_{k-vari}(o)$ to calculate the $N_{k-disp}(o)$ of the object o, so the $N_{k-disp}(o)$ can not only show the degree of deviation of the object o from $N_k(o)$ as a whole, but also reflect the distribution of o and $N_k(o)$ within the range of k neighborhood. As shown in Fig. 1, $N_{k-adist}(o_i) = N_{k-adist}(o_j) = N_{k-adist}(o_l)$ and $N_{k-vari}(o_i) \neq N_{k-vari}(o_j) \neq N_{k-vari}(o_l)$, that is o_i, o_j and o_l have the same degree of overall deviation from their respective k neighbors, but their connectivity to their neighbors are not the same, so their degree of outlier are also obviously different. In the latter experiments we can see that $N_{k-disp}(o)$ can well reflect these differences between them. As shown in Fig. 1 (a), when the $N_k(o_i)$ of object o_i is on the same circle with radius $r(r \neq 0, r = N_{k-adist}(o))$, the $N_{k-vari}(o) = 0$. At this time, in order not to let the exponent of $N_{k-disp}(o_i)$ be zero, let $Nn_{k-vari}(o) = \frac{1}{|N_k(o)|}$. From the equation (7), we can see that the $N_{k-adisp}(o)$ of the object o has three characteristics: (1) $N_{k-adisp}(o)$ is determined only by the $Nn_{k-adist}(o)$ and $Nn_{k-vari}(o)$. (2) $\forall o_i, o_j$, if $N_{k-adist}(o_i) = N_{k-adist}(o_j)$ and $N_{k-vari}(o_i) \geq N_{k-vari}(o_j)$, then $N_{k-disp}(o_i) \geq N_{k-disp}(o_j)$. (3) The different degrees of deviation of the objects from their respective neighborhoods show an exponential change in the value of k neighborhood dispersion.

Definition 6 (k-Neighborhood Density of Object o, $N_{k-dens}(o)$): The ratio of the k neighborhood dispersion of the object o to its k neighborhood average distance:

$$N_{k-dens}(o) = \frac{N_{k-disp}(o)}{N_{k-adist}(o)} \tag{8}$$

When the object o coincides with all its neighbors, $N_{k-adist}(o) = N_{k-vari}(o) = 0$, then the $N_{k-dens}(o)$ becomes meaningless. Obviously, at this point $N_{k-dens}(o)$ should be the largest in the dataset and the object o is the inners. Therefore, in order to avoid $N_{k-dens}(o)$ meaningless and to ensure that $N_{k-dens}(o)$ is the largest in the dataset, the easiest way is to let $N_{k-dens}(o)$ take a value that is slightly larger than the neighborhood density of other objects or take a large value directly.

Definition 7 (Local Deviation Coefficient of Object o, $LDC(o)$): The local deviation coefficient of object o is defined as:

$$LDC(o) = \frac{\frac{\sum_{p \in N(o)} N_{k-dens}(p)}{|N_k(o)|}}{N_{k-dens}(o)} \tag{9}$$

The LDC is defined by comparing the difference of density between the object and its local neighborhood objects, and this definition also directly restricts its local properties. Therefore, the size of $LDC(o)$ indicates the degree of abnormality of the object o in the local range. The larger the value of $LDC(o)$, the greater degree of abnormality of the object o. $LDC(o)$ quantifies the local anomaly properties of the object o in terms of the expectation and variance of the distance between the object o and its neighbors. Compared with the traditional methods, $LDC(o)$ not only takes full account for the overall density of the object o and its neighbors in the local range, but also focuses on the dispersion of these objects. Therefore, the *LDC* can better quantify the degree of local anomaly of the objects from the inherent distribution law of the local datasets.

B. SECURITY OBJECTS ELIMINATING APPROACH BASED ON THE ROUGH CLUSTERING

The existing local outlier detection algorithms need to perform nearest neighbor search on all objects to calculate their local outlier factor, so the efficiency is relatively low. Currently, scholars mainly improve the efficiency of algorithms by using various data structures to perform nearest-neighbor search. However, the proportion of outlier data in the dataset is very small relative to normal data. Therefore, when outlier detection is performed on a dataset, how to eliminate security non-outlier objects as much as possible is another method to effectively improve the efficiency of the algorithms. As shown in Fig. 2, the non-uniformly distributed dataset consists of two closely distributed clusters C_1, C_2 and some other relatively scattered points $\{o_7, o_8, \ldots, o_{21}\}$. According to the definition of local outlier factor, it can be known that the objects with higher outlier factor are mainly composed of scattered isolated points $\{o_{10}, o_{11}, \ldots, o_{21}\}$ and the points on the edge of clusters $\{o_7, o_8, o_9\}$. In practical applications, only the objects with large outliers are concerned, and the safety points with lower outliers in clusters C_1 and C_2 are not the objects to be obtained by the outlier detection algorithms. Therefore, we can eliminate the security points inside the cluster C_1 and C_2, and complete the outlier detection of the dataset by only calculating the local outlier factor of $\{o_7, o_8, \ldots, o_{21}\}$, which can greatly improve the time efficiency of the algorithms.

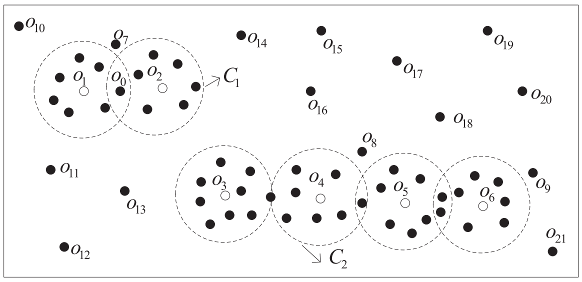


FIGURE 2. Sample dataset.

The outliers initially are the by-product of many clustering algorithms, so it is not difficult to find that clustering can exclude the inner data that occupies most of the data in the dataset. As a classic density-based clustering algorithm, the DBSCAN (density based spatial clustering of applications with noise) [41] algorithm determines the relationships of density-reachable among the data by the preset distance threshold (Eps) and number threshold (MinPts), and then derives all the maximum density-connected datasets (each dataset is a cluster). The DBSCAN algorithm has the advantages of being able to find the clusters of arbitrary shapes and being insensitive to noise data. But the DBSCAN algorithm cannot be used directly in this paper to preprocess the dataset due to the following two limitations: (1) The goal of the DBSCAN algorithm is to get the accurate clusters, and the goal of data preprocessing in this paper is to obtain suspicious outlier dataset. (2) The efficiency of DBSCAN algorithm is relatively low. This is because the DBSCAN algorithm needs to perform nearest neighbor search for all objects. Moreover, the DBSCAN algorithm is difficult to accurately determine the size of the initial threshold, and it needs to re-cluster the dataset when the threshold changes. This paper draws on the idea of the DBSCAN algorithm and the hierarchical clustering method to study an efficient rough clustering method, which can make the data with higher density in the dataset to be absorbed by inner-classes earlier according to continuously expand the distance threshold.

Define 8 (Core Cluster): If the number of objects contained in the NNDis(a given distance threshold) neighborhood of object o is not less than k(a given number threshold), then the object and its neighbourhoods form a core cluster. In this case, the o and its neighbors are security objects.

Define 9 (Directly Reachable-Cluster): Given a set of objects O, for the core clusters C_1 and C_2 in O, if $C_1 \cap C_2 \neq \emptyset$, then C_1 and C_2 are called directly reachable-clusters.

Define 10 (Inner Reachable-Cluster): If there is a core cluster chain $\{C_1, C_2, \cdots, C_i, \cdots, C_j\}$ that satisfies $C_1 = C_p$ and $C_j = Cq$, $C_i + 1$ is a cluster directly accessible from C_i about k and NNDis, then C_q is inner reachable-cluster from C_p about k and NNDis.

In the dataset O, for the given k and NNDis, the object o and its NNDis-neighborhood objects constitute a minimal core cluster. The inner reachable-cluster is a canonical pass and extension of directly reachable-cluster. Specially, if $j = 2$ in $\{C_1, C_2, \cdots, C_i, \cdots, C_j\}$, then $\{C_1, C_2, \cdots, C_i, \cdots, C_j\}$ is two directly reachable-clusters. Obviously, when two core clusters are directly reachable-clusters or inner reachable-clusters, they are largely of the same category. Therefore, the directly reachable-cluster and inner reachable-cluster are important theoretical basis for the inner-class search in this paper.

Define 11 (Inner-Class): Let O be a database of points, a cluster C_i about k and NNDis is a non-empty subset of O. If C_i is composed of one core cluster or more inner reachable-clusters, then C_i is an inner class.

Define 12 (Suspicious Point): Let $\{C_1, C_2, \cdots, C_j\}$ be the inner-classes of the database O for the parameters k and NNDis, for any point $p(p \in O)$, if $p(p \notin C_i(0 \leq i \leq n))$, then p is a suspicious point.

The first goal of our work is to search all the inner-classes and all suspicious data of the dataset for a given neighborhood threshold NNDis when the number threshold k is determined. However, it is difficult to determine an appropriate NNDis value when the users lack prior knowledge of the dataset. If the NNDis value is too large, the accuracy of the detection results will be reduced by the leakage of some outliers, and the efficiency of the algorithm cannot be improved effectively if the NNDis value is too small as the internal object cannot be eliminated to the maximum extent. But when the user changes the value of NNDis, if all the data is re-clustered every time, the time consumed is basically the same. This will cause the algorithm to consume a lot of time when the user modifies the threshold parameter every time, which will undoubtedly seriously affect the time efficiency of the algorithm. In this regard, this paper studies a Rough Clustering algorithm based on Multi-Level Queries (RCMLQ). When the user changes the NNDis value, it is not necessary to re-query all the data, so that the inner-classes with different NNDis values can be quickly obtained. Specifically as shown in Algorithm 1, 1) normalize O by formula (1); 2) search all core clusters in the dataset O under the initial neighborhood distance threshold NNDis and the given number threshold k; 3) merge all directly reachable-clusters and inner reachable-clusters, output all inner-classes and uncategorized data under the neighborhood threshold $NNDis$; 4) Determine whether the number of unclassified data meets the requirements. If yes, the algorithm ends, otherwise the NNDis is expanded and the above operations are repeated for unclassified data until the number of unclassified data meets a predetermined determination condition.

The RCMLQ algorithm requires three initial parameters (NNDis, k and MinN) when preprocessing the datasets. For these parameters, if a set of suitable initial values can be determined, it will undoubtedly simplify the subsequent calculation process and improve the efficiency of RCMLQ algorithm. For the determination of NNDis, this paper uses the local heuristic method, that is, randomly extracts several subsets in the local range from the dataset, and then calculates the average value of k-neighborhood distances of all objects in each data subset, and finally takes a suitable intermediate value from these average values as the initial value of NNDis. In the statistics, the quartile can well be used to represent the degree of dispersion of the sequential data. Therefore, we sort the above average in ascending order, and then take the upper quartile between the minimum value (v_{min}) and the maximum value (v_{max}) as the initial value of NNDis, that is, $NNDis = v_{min} + \frac{3}{4}(v_{max} - v_{min})$. In the clustering process of the RCMLQ algorithm, the distance threshold of the subsequent calculation is doubled on the basis of the previous one. For the setting of the k, we maintain the same size of neighborhood in the full text, that is, the k here is the same as the local size of the LDC. For the size of MinN, it is not only related to the outlier rate of datasets, but also related to the efficiency of E2DLOS algorithm. This is because the suspected outlier data obtained after the preprocessing of the dataset must contain all outlier data. Moreover, the smaller the MinN, the more safety data is excluded by the RCMLQ algorithm, and the amount of data that is needed to calculate the local outlier factor is also smaller. The RCMLQ algorithm continuously excludes the non-outlier data based on the density of the data, so the initial value of MinN is only needs to be slightly larger than the amount of target outlier data. However, it is also necessary to calculate the k neighborhood density of the neighbors of the suspect outlier data. And the objects with similar densities in the same local space in the dataset often have similar neighbors. Therefore, the efficiency of E2DLOo algorithm is relatively stable when MinN takes a little larger than the amount of target outlier data and changes within a small range. We also verified this point in the experiments. In addition, it is often difficult to accurately determine the outlier rate of the datasets in practical applications. Therefore, in order to better analyze the variation of the local outlier factor of the data in the datasets, we generally let MinN take 1.5 to 2.5 times of the amount of target outlier data, and the experiments in this paper take the intermediate value 2.

Theorem 1: Let the dataset $O = \{o_1, o_2, \cdots, o_n\}$, the i inner-classes $\{C_1, C_2, \cdots, C_i\}$ obtained by nearest neighbor search according to the given threshold k and $NNDis$. Then, when the inner-class C_{i+1} is generated, the point belonging to the $(i+1)th$ inner-class C_{i+1} is impossible to belong to the previous i inner-classes $\{C_1, C_2, \cdots, C_i\}$.

Proof: For any point $o_j (1 \leq j \leq n)$ in O belonging to the $xth(x \leq i)$ inner-class C_x, suppose that o_j also belongs to the $(i+1)th$ inner-class C_{i+1}, that is, o_j is in the $NNDis$ neighborhood of a certain point in C_{i+1}. From definition 14, it can be known that when o_j belongs to C_x and C_{i+1} simultaneously, C_x and C_{i+1} are directly reachable-clusters. Therefore, they can be merged into an inner class, that is, C_x and C_{i+1} must belong to the same inner class.

The RCMLQ algorithm proposed in this paper is a process of continuously absorbing data around the core clusters or increasing the number of core clusters according to the distribution density of the data. This process allows the data with higher density in the dataset to be absorbed by inner-classes earlier. Obviously, when the distance threshold is increased to re-cluster the dataset, it is not necessary to re-cluster the data that have been classified, and only the data that have not been classified need to be clustered. And the theorem 1 guarantees that when searching for a new core cluster under the thresholds k and $NNDis$, it only needs to be carried out in the unclassified points, which can greatly improve the efficiency of the algorithm. It is obvious that the RCMLQ algorithm only needs to perform nearest neighbor search for a small amount of data when performing rough clustering, so the RCMLQ is efficient. As shown in Fig. 2, when the RCMLQ performs a search to generate a core cluster C_1, the nearest neighbor search clustering may not be performed again for other data in the cluster C_1 in the following process. As Algorithm 1, the RCMLQ algorithm effectively eliminates the safe data that occupies a larger amount of the dataset by preprocessing the dataset, which can greatly reduce the number of data that is needed to calculate the local outlier factor. Thus, the efficiency of the local outlier detection algorithm is greatly improved. In addition, the RCMLQ algorithm improves the efficiency of local outlier detection by preprocessing the data to reduce the amount of data that is needed to calculate the outlier factor. Therefore, it can be applied as an independent part of other local outlier detection algorithms, and can also be used together with the method of improving outlier detection efficiency by

Algorithm 1 RCMLQ

```
Input: Dataset $O = \{o_1, o_2, \cdots, o_n\}$, $k$, $NNDis$, $MinN$
Output: inner-cluster set $C$, suspicious point set $O_{doubt}$
1:  initialize $C$, $O_{doubt}$, $O_{NNDis}$;
2:  create a $n \times m$ matrix $M = [o_1, o_2, \cdots, o_n]^T$
3:  for $j = 1$ to $m$ do
4:      conduct 0-1 normalization on every column of $M$ according to Eq.(1);
5:  end for
6:  if $O \neq \phi$ do
7:      search the dataset $O_{NNDis}$ in the $NNDis$ neighborhood of any point $o_j$ in $O$;
8:      if $|O_{NNDis}| \geq k$ do
9:          let $o_j$ and its neighbors as a core cluster $C_j$;
10:         $C = C + C_j$ (Add $C_j$ to $C$);
11:         $O = O - C_j$ (Remove $C_j$ from $O$);
12:     else do
13:         $O = O - o_j$ (Remove $o_j$ from $O$);
14: end if
15: Output the dataset $O_{doubt}$ that do not yet have core cluster attributes in $O$;
16: merge all directly reachable-clusters in core cluster set $C$;
17: if $|O_{doubt}| \geq MinN$ do
18:     $NNDis = 2 \times NNDis$;
19:     repeat the calculation of steps 8 to 18 for $O_{doubt}$;
20: else do
21:     return C and $O_{doubt}$;
```

Algorithm 2 E2DLOS

```
Input: Dataset $O = \{o_1, o_2, \cdots, o_n\}$, $k$, $MNDis$, $MinN$, $m_{outlier}$
Output: Outlier series $O_{outlier} = \{o_1, o_2, \cdots, o_{m_{outlier}}\}$
1:  initialize $O_{outlier}$, $C$, $I$, $O_{temp}$, $O_{knn}$, $O_{doubt}$;
2:  using algorithm 1 to preprocess dataset $O$ and get inner-cluster set $C$ and suspicious point set $O_{doubt}$;
3:  for $j = 1$ to $n_{doubt}(n_{doubt} = |O_{doubt}|)$ do
4:      Search k nearest neighbors $O_{knn}$ of $o_j$;
5:      $O_{temp} = O_{temp} \cup O_{knn}$;
6:  end for
7:  output the dataset $O_{temp}(O_{temp} = O_{temp} \cup O_{doubt})$;
8:  for $j = 1$ to $n_1(n_1 = (|O_{temp}|)$ do
9:      calculate $N_{k-adist(o_j)}$ according to Eq.(3);
10:     calculate $N_{k-vari(o_j)}$ according to Eq.(5);
11: end for
12: for j=1 to $n_1$ do
13:     conduct 0-1 normalization on $N_{k-adist(o_j)}$ and $N_{k-vari(o_j)}$ according to Eq.(1);
14:     calculate $N_{k-disp(o_j)}$ according to Eq.(7);
15:     calculate $N_{k-dens(o_j)}$ according to Eq.(8);
16: end for
17: for j=1 to $n_{doubt}$ do
18:     calculate $I_j(I_j = LDC(o_j))$ according to Eq.(9);
19: end for
20: sort objects set $O_{doubt}$ in descending order according to $I$;
21: return $O_{outlier} = \{o_1, o_2, \cdots, o_{m_{outlier}}\}$ (the first $m_{outlier}$ objects with large outliers);
```

improving the efficiency of nearest neighbor search. In summary, the RCMLQ algorithm further improves the efficiency of the local outlier detection approaches from a new direction.

C. EFFICIENT LOCAL OUTLIER DETECTION ALGORITHM

This paper proposes a density-based local outlier detection approach (E2DLOS), which takes full advantage of the distribution characteristics of the dataset itself. Firstly, in order to make the local outlier factor more sensitive to the anomaly degree of the objects, we take full account of the degree of dispersion of the object and its neighbors, and define a new local outlier factor (LDC) of the object according to the expectation and variance of the distance between the object and its neighbors. In addition, we study a rough clustering approach based on multi-level query (RCMLQ), which can efficiently partition the dataset into security dataset and possible outlier dataset. Obviously, after the dataset is preprocessed by RCMLQ, it does not change the neighborhood properties of the data. Therefore, after preprocessing the dataset, it does not change the inherent outlier attribute of the data itself. In this way, we only need to calculate the local outlier factor for the possible outlier data, which can greatly reduce the amount of data that is needed to calculate the outlier factor.

As shown in Algorithm 2, for the given dataset O, the process of the E2DLOS algorithm is as follows: 1) preprocess the dataset O by algorithm 1 to get inner class set C and suspicious point set O_{doubt}; 2) search for the k-neighborhood objects of each data in O_{doubt} and output the set O_{temp} of all these k-neighborhood objects; 3) calculate the k-neighborhood average distance of each object in O_{doubt} and O_{temp} by the formula (3) and normalize them; 4) calculate the k-neighborhood variance of each object in O_{doubt} and O_{temp} by formula (5) and normalize them; 5) calculate the k-neighborhood dispersion of each object in O_{doubt} and O_{temp} by formula (7); 6) calculate the k-neighborhood density of each object in O_{doubt} and O_{temp} by formula (8); 7) calculate the local deviation coefficient LDC of each object in O_{doubt} by formula (9); 8) sort the objects of the dataset O_{doubt} in descending order according to LDC, and the first $m_{outlier}$ objects are the outliers in O.

For the E2DLOS algorithm, we first use the rough clustering algorithm RCMLQ to preprocess it, and then perform the outlier quantification calculation on the suspicious dataset. Finally, the first $m_{outlier}$ objects with the largest outlier factor are obtained as the outlier objects of the dataset. We use the R*-tree to determine the neighborhood of the object, and its computational complexity will be greatly reduced. Assuming that the size of dataset is n, the dimension of each data is m, the number of neighbors of the object is k, s is the minimum number of items of each index node in the R*-tree, then the complexity of normalized attribute value to [0, 1] is $O(mn)$; For the RCMLQ algorithm, it is a process in which the core clusters are continuously merged so that no neighbor search is performed on each data in the dataset during the operation, and its time complexity is $O(n_2(klog_s n))$, where n_2 is the number of objects that is needed to perform nearest neighbor search. The RCMLQ algorithm only needs to traverse the dataset once when performing clustering, and the neighbors of the target object do not need to perform nearest neighbor search in the process of generating the core cluster, so n_2 is much smaller than n. As shown in Algorithm 2, after the RCMLQ algorithm is processed, the number of objects needed to calculate the k-neighborhood average distance is n_1, and the time complexity of this process is $O((klog_s n + km))$, where n_1 is the number of suspicious points combined with their respective neighborhood objects. The complexity of calculating the k neighborhood variance of the test object is $O(kn_1)$; The time complexity of normalized k-neighborhood average distance and k-neighborhood variance are both $O(n_1)$; The complexity of calculating LDC is $O(n_{doubt}(klog_s n_{doubt}))$, where n_{doubt} is the number of suspicious points; Using the bucket sort algorithm with the time complexity is the lowest, the time complexity is $O(n_1)$; For the determination of the outliers, the time complexity of extracting the first $m_{outlier}$ objects is $O(m_{doubt})$; So the total time complexity of N2DLOF is $O(k(n_1 + n_2)log_s n)$. Since $(n_1 + n_2) < n$ in general, the time complexity of the algorithm in this paper is much less than $O(n^2)$ of the LOF algorithm, and also less than $O(knlogn)$ of the algorithm that uses the classical data structure to perform the nearest neighbor search.

IV. EVALUATION

In this section, we perform a series of comparative experiments on multiple synthetic datasets and real datasets to verify the superiority of the E2DLOS algorithm proposed in this paper. The LOF algorithm is the most classical algorithm that is groundbreaking in local outlier detection. The SimplifiedLOF, LoOP, Ldof and COF are the classical extensions of LOF, and the authors in the literature [38] have verified by theory and experiments that they have better time efficiency or higher accuracy. In addition, due to the limitation of the length of the article, this paper compares the five algorithms from the time efficiency and the accuracy of the test results to verify the superiority of the E2DLOS algorithm. All approaches have been implemented in java and the experiments have been conducted on the Intel(R) Core(TM) i5-3470 CPU @ 3.20 GHz Windows 7 machine with 4GB of RAM.

A. SYNTHETIC DATASET

The reason for using synthetic datasets in experiments is that we can better control the distribution of datasets. First, we verify the superiority of the E2DLOS algorithm in accuracy with the synthetic scattered datasets. As shown in Fig. 1, for the three synthesized datasets, each dataset contains 16 objects, each object in the dataset is a two-dimensional vector, and the average distance of o_i, o_j and o_j to all other objects in their respective dataset are equal. It can be clearly seen from the Fig. 1 that o_i and o_l are the points with the largest outliers in their respective datasets. While o_j is at the center of the dataset in which it is located and its degree of outlier is minimal from a global perspective, but the anomaly of the object in its lower right corner may be smaller than it at different k values. In this work, we use the LOF, SimplifiedLOF, LoOP, Ldof, COF and the E2DLOS algorithm proposed in this paper to quantify the outliers of o_i, o_j and o_l, and employ the ranking of the outliers of each object in their respective datasets to show the detection results of the algorithms. When the k takes different values, the experimental results are shown in tables 1-3. It should be noted that in the tables 1-3, the first number in the parentheses indicates the size of the local outlier factor, and the second number indicates its ranking.

It can be found from Table 1 that when k=15, the outliers obtained by each algorithm are global outliers. At this time, in addition to the SimplifiedLOF algorithm, the other four algorithms can accurately quantify the local outlier factor of two of the three tested objects. This is because the SimplifiedLOF algorithm replaces the reachability distance of the LOF algorithm with the kNN distance, which is a more coarser-grained calculation. In addition, except for the LoOP

TABLE 1. Comparison of test results (k=15).

test objects	LOF	SimplifiedLOF	LoOP	LDOF	COF	E2DLOS
o_i	(1.0331,1)	(1.0340,1)	(0.6973,16)	(1.3738,1)	(3.0815,1)	(1.6086,1)
o_j	(1.0158,2)	(1.0226,1)	(0.7468,15)	(1.4088,1)	(1.0583,3)	(1.6086,16)
o_l	(1.3030,1)	(0.9524,16)	(1.7403,1)	(3.737,1)	(9.9759,1)	(1.6086,1)
precision	66.7%	33.3%	66.7%	66.7%	66.7%	66.7%

TABLE 2. Comparison of test results (k=10).

test objects	LOF	SimplifiedLOF	LoOP	LDOF	COF	E2DLOS
o_i	(1.0386,3)	(1.0585,1)	(0.9029,16)	(1.2808,2)	(3.3242,1)	(2.0870,1)
o_j	(0.9396,16)	(0.9586,13)	(0.8458,15)	(1.4550,1)	(1.09,2)	(2.0870,15)
o_l	(1.3664,1)	(0.9544,16)	(1.9815,1)	(3.5771,1)	(9.9668,1)	(2.0870,1)
precision	33.3%	33.3%	66.7%	33.3%	66.7%	100%

TABLE 3. Comparison of test results (k=5).

test objects	LOF	SimplifiedLOF	LoOP	LDOF	COF	E2DLOS
o_i	(1.2198,1)	(1.0549,2)	(1.5083,1)	(1.3902,2)	(3.2194,1)	(3.3554,1)
o_j	(0.9316,15)	(0.9695,13)	(0.8510,16)	(1.2686,1)	(1.1978,1)	(0.8835,15)
o_l	(1.4111,1)	(0.7028,16)	(2.0443,1)	(3.0603,1)	(9.9469,1)	(2.8163,1)
precision	100%	0%	66.7%	33.3%	66.7%	100%

algorithm, the other four algorithms cannot accurately quantify the local outlier factor of o_j in dataset 2. This is because the quadratic mean distance utilized by the LoOP algorithm is sensitive to the scenes where the magnitude of the distance between the object and its neighboring objects exhibits a gradient change. However, the quadratic mean distance is insensitive to the small change in the distance between the object and its neighboring objects, so the LoOP algorithm cannot accurately quantify the local outlier factor of o_i in dataset 1. Nevertheless, the E2DLOS algorithm in this paper can almost accurately obtain the local outlier factor of o_j as its ranking is only one bit worse than the exact ranking. It can be seen from Table 2 that when k=10, the LOF, SimplifiedLOF and Ldof algorithms can only accurately calculate the local outlier factor of one of the three tested objects, the LoOP and COF algorithms can accurately calculate the local outlier factor of two of the three tested objects, and our E2DLOS algorithm can accurately calculate the local outlier factor of these three objects. Therefore, when the value of k decreases, the LoOP and COF algorithms maintain a higher accuracy, and the E2DLOS algorithm proposed in this paper has an ideal detection result. It can be seen from Table 3 that when k=5, the SimplifiedLOF algorithm cannot get any desired result, the LDOF algorithm can only accurately calculate the local outlier factor of o_l in the dataset 3, the LoOP and COF algorithms can accurately calculate the local outlier factor of two of them, while the LOF algorithm and the E2DLOS algorithm proposed in this paper can accurately detect the outliers for the datasets of varying degrees of dispersion.

In summary, the sensitivity of local outlier factor that obtained by each algorithm to the degree of abnormality of the data with different degrees of dispersion is quite different. The effectiveness of the SimplifiedLOF algorithm is generally inferior to the other five algorithms. The LDOF algorithm can have a certain detection effect when the k value is large, but the effectiveness is also low when the k value is small. The LOF algorithm has a large difference in detection results under different k values, but it can obtain ideal detection results when k takes an appropriate value. And the LoOP and COF algorithms are insensitive to the change of k value and achieve acceptable results at different k values, but in general it is not as good as the E2DLOS algorithm proposed in this paper.

B. REAL DATASET

In this section, we chose the following real-world datasets to evaluate the six algorithms. The high dimensional dataset Forest Cover (FC) contains 581,012 records with 54 quantitative attributes. It is available at the UCI KDD Archive [42] and was also used in [43]. TAO [44] contains 575,648 records with 3 attributes (SST, RH, Prec), where SST is the sea surface temperature, measured in units of degrees centigrade at a depth of 1 meter, RH is the relative humidity, measured in units of percent at a height of 3 meters above mean sea level, and Prec is the precipitation, measured in units of millimeters per hour at a height of 3.5 meters above mean sea level. A smaller TAO dataset was used in [43] and [45]. Stock contains 1,048,575 records with 1 attribute. It is available at UPenn Wharton Research Data Services [46]. A similar stock trading dataset was used in [47]. In the experiments of this article, we intercepted 10,000 of the records in each dataset.

First we compare the effectiveness of all algorithms. We take $k = 5$, and list the number of identical objects among the 10 objects with the largest local outlier factor obtained by the all algorithms in each dataset, as shown in tables 4-6. It can be seen from the tables that the differences among these algorithms in the detection results of the three real datasets

TABLE 4. The number of identical objects among the 10 objects with the largest local outlier factor obtained by each algorithm on stock.

algorithms	LOF	SimplifiedLOF	LoOP	LDOF	COF	E2DLOS
LOF	x	2	2	0	2	2
SimplifiedLOF	2	x	1	0	1	0
LoOP	2	1	x	0	2	6
LDOF	0	0	0	x	0	0
COF	2	1	2	0	x	5
E2DLOS	2	0	6	0	5	x

TABLE 5. The number of identical objects among the 10 objects with the largest local outlier factor obtained by each algorithm on TAO.

algorithms	LOF	SimplifiedLOF	LoOP	LDOF	COF	E2DLOS
LOF	x	2	9	0	3	7
SimplifiedLOF	2	x	2	0	1	1
LoOP	9	2	x	0	5	10
LDOF	0	0	0	x	0	0
COF	3	1	5	0	x	5
E2DLOS	7	1	10	0	5	x

TABLE 6. The number of identical objects among the 10 objects with the largest local outlier factor obtained by each algorithm on FC.

algorithms	LOF	SimplifiedLOF	LoOP	LDOF	COF	E2DLOS
LOF	x	1	6	0	2	5
SimplifiedLOF	1	x	1	0	0	0
LoOP	6	1	x	0	4	9
LDOF	0	0	0	x	0	0
COF	2	0	4	0	x	5
E2DLOS	5	0	9	0	5	x

TABLE 7. The first 5 outliers and their neighbors in TAO mined by LOF.

test objects	outliers	k neighbors
4464(1.88, 89.21, 23.305)	13.6308	(0.85, 89.27, 23.353),(0.1, 89.14, 23.272),(0.11, 89.34, 23.603),(0.07, 89.09, 23.334),(0.13, 89.49, 23.755)
4463(0.85, 89.27, 23.353)	11.6204	(0.1, 89.14, 23.272),(0.11, 89.34, 23.603),(0.05, 89.29, 23.37),(0.05, 89.29, 23.33),(0.07, 89.09, 23.334)
9284(1.15, 90.99, 25.438)	9.3292	(0.37, 90.39, 25.645),(0.13, 91.32, 25.511),(0.4, 91.18, 24.666),(0.04, 91.1, 25.433),(0.02, 90.93, 25.387)
9227(0.37, 90.39, 25.645)	8.6075	(0.01, 90.39, 25.622),(0.01, 90.36, 25.671),(0.01, 90.39, 25.584),(0.01, 90.33, 25.603), (0.0, 90.39, 25.619)
7442(0.65, 97.32, 25.094)	7.4841	(0.06, 97.37, 25.113),(0.01, 97.45, 24.948),(0.02, 97.12, 24.923),(-0.01, 97.4, 24.936),(0.02, 97.12, 24.91)

TABLE 8. The first 5 outliers and their neighbors in TAO mined by SIMPLIEDLOF.

test objects	outliers	k neighbors
9284(1.15, 90.99, 25.438)	3.5671	(0.37, 90.39, 25.645),(0.13, 91.32, 25.511),(0.4, 91.18, 24.666),(0.04, 91.1, 25.433),(0.02, 90.93, 25.387)
3492(-0.94, 81.75, 23.954)	3.2307	(-1.07, 81.41, 23.952),(-0.23, 81.59, 23.949),(-0.17, 81.85, 23.955),(-0.18, 82.05, 23.986),(-0.11, 81.8, 23.988)
4464(1.88, 89.21, 23.305)	3.1486	(0.85, 89.27, 23.353),(0.1, 89.14, 23.272),(0.11, 89.34, 23.603),(0.07, 89.09, 23.334),(0.13, 89.49, 23.755)
6621(2.12, 97.07, 25.082)	3.0186	(1.64, 96.2, 25.063),(0.65, 97.32, 25.094),(2.23, 96.68, 23.261),(1.61, 96.66, 23.248),(2.18, 97.83, 23.239)
6622(1.64, 96.2, 25.063)	2.9346	(2.12, 97.07, 25.082),(0.97, 95.41, 25.438),(1.44, 95.11, 25.436),(0.54, 95.27, 25.436),(0.65, 97.32, 25.094)

TABLE 9. The first 5 outliers and their neighbors in TAO mined by LDOF.

test objects	outliers	k neighbors
1(0.04, 84.87, 23.436)	1.5257	(0.01, 84.9, 23.454),(0.01, 84.9, 23.389),(0.02, 84.82, 23.399),(-0.01, 84.82, 23.395),(0.02, 84.95, 23.468)
3(0.14, 84.92, 23.441)	1.0105	(0.04, 84.87, 23.436),(0.02, 84.95, 23.468),(0.01, 84.9, 23.454),(0.01, 84.9, 23.389),(0.0, 84.97, 23.448)
2(0.09, 85.12, 23.438)	0.9791	(0.04, 85.12, 23.456),(0.09, 85.1, 23.494),(0.02, 85.1, 23.395),(0.02, 85.02, 23.414),(-0.04, 85.07, 23.403)
4(0.07, 84.74, 23.443)	0.5155	(0.01, 84.72, 23.412),(0.02, 84.69, 23.463),(0.01, 84.74, 23.366),(0.02, 84.82, 23.399),(-0.04, 84.74, 23.463)
10(-0.19, 82.2, 23.415)	0.3452	(-0.04, 82.23, 23.407),(-0.04, 82.1, 23.406),(-0.05, 82.25, 23.309),(-0.06, 82.15, 23.296),(-0.06, 82.15, 23.277)

TABLE 10. The first 5 outliers and their neighbors in TAO mined by LoOP.

test objects	outliers	k neighbors
4464(1.88, 89.21, 23.305)	13.778	(0.85, 89.27, 23.353),(0.1, 89.14, 23.272),(0.11, 89.34, 23.603),(0.07, 89.09, 23.334),(0.13, 89.49, 23.755)
4463(0.85, 89.27, 23.353)	12.4129	(0.1, 89.14, 23.272),(0.11, 89.34, 23.603),(0.05, 89.29, 23.37),(0.05, 89.29, 23.33),(0.07, 89.09, 23.334)
9227(0.37, 90.39, 25.645)	10.0282	(0.01, 90.39, 25.622),(0.01, 90.36, 25.671),(0.01, 90.39, 25.584),(0.01, 90.33, 25.603), (0.0, 90.39, 25.619)
9284(1.15, 90.99, 25.438)	9.4794	(0.37, 90.39, 25.645),(0.13, 91.32, 25.511),(0.4, 91.18, 24.666),(0.04, 91.1, 25.433),(0.02, 90.93, 25.387)
7442(0.65, 97.32, 25.094)	8.2875	(0.06, 97.37, 25.113),(0.01, 97.45, 24.948),(0.02, 97.12, 24.923),(-0.01, 97.4, 24.936),(0.02, 97.12, 24.91)

TABLE 11. The first 5 outliers and their neighbors in TAO mined by COF.

test objects	outliers	k neighbors
7445(0.18, 97.94, 25.108)	19.7094	(0.01, 98.0, 24.933),(0.0, 98.0, 24.928),(0.01, 98.02, 24.908),(0.01, 97.78, 24.954),(0.01, 97.78, 24.946)
9227(0.37, 90.39, 25.645)	10.0282	(0.01, 90.39, 25.622),(0.01, 90.36, 25.671),(0.01, 90.39, 25.584),(0.01, 90.33, 25.603), (0.0, 90.39, 25.619)
5508(-1.99, 96.28, 23.436)	12.7317	(-0.75, 96.18, 23.307),(-0.74, 96.2, 23.425),(-0.38, 96.38, 23.344),(-0.31, 96.25, 23.356), (-0.26, 96.25, 23.344)
7696(-0.01, 80.65, 25.159)	10.4895	(0.02, 80.68, 24.576),(-0.08, 80.7, 24.571),(0.03, 80.5, 24.573),(-0.04, 80.83, 24.58),(-0.08, 80.8, 24.573)
7442(0.65, 97.32, 25.094)	8.2875	(0.06, 97.37, 25.113),(0.01, 97.45, 24.948),(0.02, 97.12, 24.923),(-0.01, 97.4, 24.936),(0.02, 97.12, 24.91)

TABLE 12. The first 5 outliers and their neighbors in TAO mined by E2DLOS.

test objects	outliers	k neighbors
4464 (1.88, 89.21, 23.305)	13.6262	(0.85, 89.27, 23.353),(0.1, 89.14, 23.272),(0.11, 89.34, 23.603),(0.07, 89.09, 23.334),(0.13, 89.49, 23.755)
4463 (0.85, 89.27, 23.353)	12.5269	(0.1, 89.14, 23.272),(0.11, 89.34, 23.603),(0.05, 89.29, 23.37),(0.05, 89.29, 23.33),(0.07, 89.09, 23.334)
9227(0.37, 90.39, 25.645)	10.3676	(0.01, 90.39, 25.622),(0.01, 90.36, 25.671),(0.01, 90.39, 25.584),(0.01, 90.33, 25.603), (0.0, 90.39, 25.619)
9284(1.15, 90.99, 25.438)	9.6165	(0.37, 90.39, 25.645),(0.13, 91.32, 25.511),(0.4, 91.18, 24.666),(0.04, 91.1, 25.433),(0.02, 90.93, 25.387)
7442(0.65, 97.32, 25.094)	8.8098	(0.06, 97.37, 25.113),(0.01, 97.45, 24.948),(0.02, 97.12, 24.923),(-0.01, 97.4, 24.936),(0.02, 97.12, 24.91)

are similar. Specifically, the detection results of LDOF on the three datasets are different from the other algorithms, and the detection results of simpliedLOF and the other algorithms are a little similar. In addition, the LoOP and COF as well as the LoOP and the E2DLOS have a certain similarity with the test results on the three datasets, while E2DLOS has a high similarity with the LoOP. However, which algorithm is more accurate? It depends on the extent to which the object deviates from its neighbors. In view of the limitation of the article space and the better visualization of the effectiveness of each algorithm, this work selects the three-dimensional dataset TAO to analyze the detection results of each algorithm in detail.

We first take $k = 5$ and list the five points with the largest value of the local outlier factor obtained by each algorithm and their respective neighborhood objects, as shown in tables 7-12. It should be noted that the data in the first column in the tables 7-12, the data in front of the parentheses

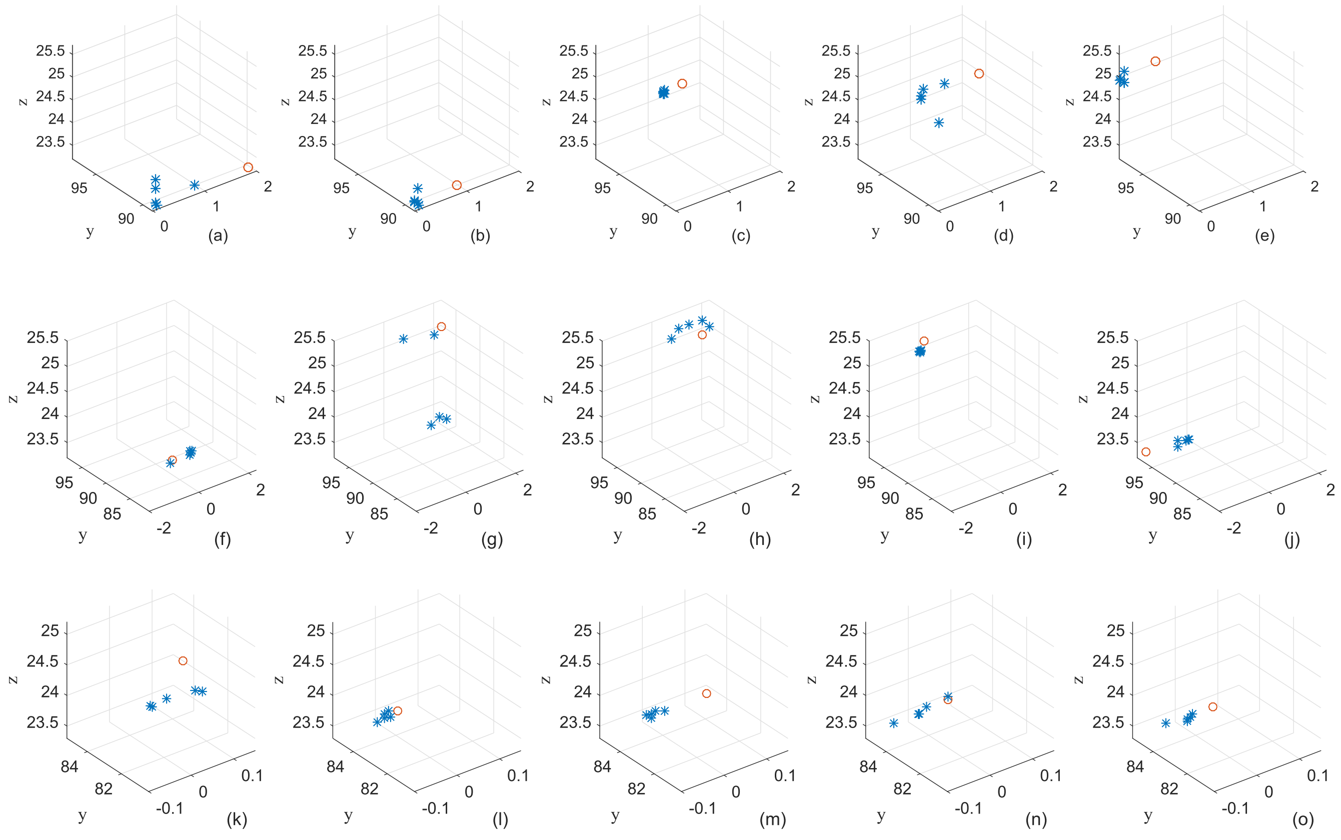


FIGURE 3. **The relationship between the five data with the largest local outlier factor and their respective neighbors mined by each algorithm in TAO.**

represent the location of test data in the original dataset, and the data in the parentheses is the specific test data. Then, we compare the sensitivity of the local outlier factor obtained by each algorithm to the local deviation degree of the objects according to the distribution law between the objects and their respective neighbors. In order to more intuitively compare the sensitivity of the local outlier factor of each algorithm to the local anomalies degree of objects with different distribution laws, we use the three-dimensional space to plot the top five outlier data and their respective neighbors obtained by each algorithm. As shown in Fig. 3, where (a, b, c, d, e) are the first five outlier objects and their respective neighbors obtained by E2DLOS algorithm or LoOP algorithm, (d, f, a, g, h), (i, c, j, k, e) and (a, b, d, c, e) are the first five outlier objects and their respective neighbors for the simplifiedLOF algorithm, COF algorithm and the LOF algorithm. Since the first five objects obtained by the Ldof algorithm are not the same as other algorithms, we only list the first four objects (l, m, n, o) and their neighbors.

As can be seen from the tables, the E2DLOS algorithm and the LoOP algorithm have the same 5 points with the largest local outlier factor and the same arrangement order. However, they appear different for the first time to the eighth objects and there are some differences in the ordering of subsequent outlier data. The E2DLOS algorithm and the LOF algorithm find the 5 same points with the largest local outlier factor but the ordering has a certain difference. Compared with the E2DLOS algorithm, the SimpliedLOF algorithm and COF algorithm have only two common points, and the five points obtained by the LDOF algorithm are completely different. This is because the LOF algorithm and the LoOP algorithm do not take into account the distribution of the object and its neighborhood objects like the E2DLOS algorithm, so the E2DLOS algorithm has better sensitivity to scattered data. In addition, the simpliedLOF algorithm is more sensitive to the dataset with large kNN distance, the COF algorithm is more sensitive for datasets with connectivity characteristics in the data patterns, the LDOF algorithm is more sensitive to the objects with large distance among the neighborhood objects. From Fig. 3, it can be seen clearly that the five data points obtained by the E2DLOS algorithm are indeed outliers, and the E2DLOS algorithm is more sensitive to the degree of deviation of the data within the local range. Therefore, the outlier factor obtained by the E2DLOS can better represent the degree of abnormality of the object in the scattered data, which makes it have higher accuracy in the anomaly detection of the scattered data.

Here we verify the efficiency of the E2DLOS algorithm. We take full account of the influence of various factors on the algorithms and test their time efficiency by changing various conditions. The experimental contents include the effect of the RCMLQ algorithm in improving the efficiency of the E2DLOS algorithm, the RCMLQ algorithm and the classical data structures are independent in improve the efficiency of the E2DLOS algorithm, and the effect of the size of the data dimension and the size of the local neighborhood on the time efficiency of all algorithms. Therefore, we set up the following combinations of conditions to perform local outlier detection on the dataset. Specifically, 1) use LDC alone; 2) use LDC and RCMLQ together; 3) combine LDC, RCMLQ and R*-tree; 4) use LDC and R*-tree together; 5) apply R*-tree to RCMLQ (RCMLQ (R*-tree)) and combined with LDC and R*-tree. Finally, we change the size of k for different dimensional test datasets and use E2DLOS (combined with LDC and RCMLQ) algorithm to perform local outlier detection, and then compare it with the traditional LOF algorithm, SimplifiedLOF algorithm, LoOP algorithm, Ldof algorithm and COF algorithm. The experimental results are shown in figures 4-6.

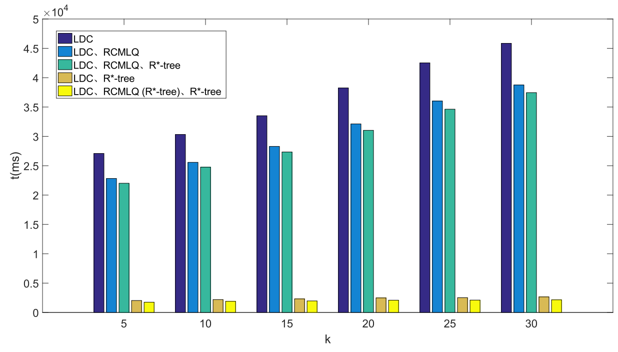


FIGURE 4. The running time comparison of local outlier detection on TAO under different conditions.

As shown in Fig. 4, when R*-tree is used for nearest neighbor search, it has a significant effect on improving the efficiency of the local outlier detection algorithms. When using only the RCMLQ algorithm, the time of outlier detection can be reduced by 16 percent to 18 percent at different k values. Furthermore, after the RCMLQ algorithm is used to preprocess the dataset, the efficiency of local outlier detection algorithm is not significantly improved when the R*-tree is applied to the nearest neighbor search of suspicious outlier data. This is because the nearest neighbor search of the RCMLQ algorithm determines the efficiency of local outlier detection, and the nearest neighbors of many suspicious outlier data have already been determined in the preprocessing. Similarly, when the R*-tree is used for the nearest neighbor search of the RCMLQ algorithm and suspected outlier data, the time efficiency of the algorithm can be further improved on the basis of the use of R*-tree alone, which can reach 15 percent to 18 percent at different k. Therefore, the RCMLQ studied in this paper effectively improves the time efficiency of the local outlier detection algorithm from a new perspective.

As shown in Fig. 5, we compare the running time of E2DLOS algorithm, LOF algorithm, simpliedLOF algorithm, LoOP algorithm, COF algorithm and LDOF algorithm in the case of different k values. It can be seen that the running time of the LDOF algorithm is relatively low and little affected by the change of k, which is because the LDOF algorithm does not need to calculate the density ratio of the tested object to its neighborhood objects. Compared with the other algorithms, LDOF is a coarse-grained local outlier factor. The runtime of the LoOP algorithm and the COF algorithm are similar at different k values. And the running time of the E2DLOS algorithm is slightly smaller than that of the LoOP algorithm and the COF algorithm, but it is much smaller than the LOF algorithm and the simpliedLOF algorithm. Therefore, the E2DLOS algorithm presented in this paper is efficient.

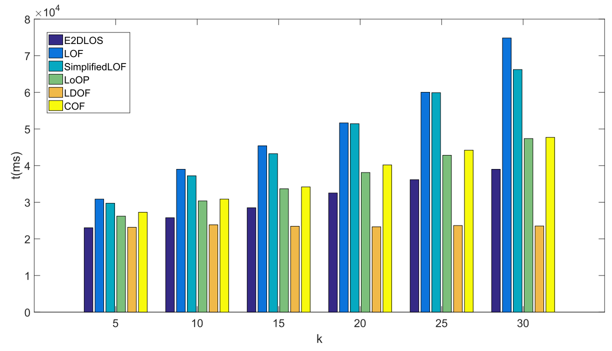


FIGURE 5. The running time of each algorithm on TAO at different k.

As shown in Fig. 6, we take $k = 5$ and compare the running times of E2DLOS algorithm, LOF algorithm, simplifiedLF algorithm, LoOP algorithm, COF algorithm and LDOF algorithm on the datasets of different dimensions. It can be found that the time efficiency of all algorithms is reduced when the dimensions of the datasets are increased, and the LDOF algorithm is the most obvious when the dataset is increased from one-dimensional to three-dimensional. In addition, when the neighborhood search range is small ($k = 5$), in addition to the high efficiency of the LDOF algorithm on the dataset stock, the E2DLOS algorithm proposed in this paper has some advantages in time efficiency on the three datasets with different dimensions.

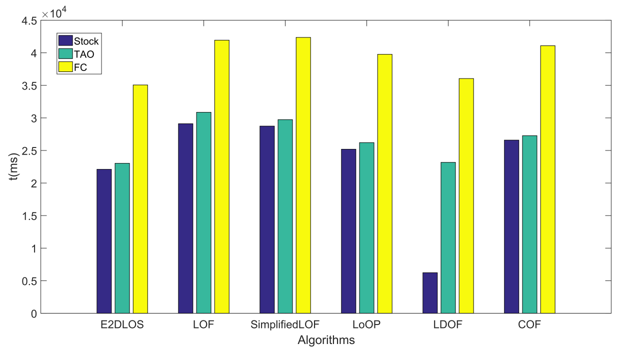


FIGURE 6. The running time of each algorithm on the datasets of different dimensions ($k = 5$).

V. CONCLUSIONS

In this paper, we take full advantage of the distribution of the dataset to redefine a local outlier factor (LDC), and study a rough clustering algorithm (RCMLQ) based on the multi-level queries to preprocess the dataset. Based on these, we propose an efficient local outlier detection algorithm (E2DLOS). Compared with the traditional local outlier factor quantification methods, the LDC is more sensitive to scattered data and this improves the accuracy of the local outlier detection. Moreover, the RCMLQ greatly reduces the amount of data that is needed to be quantified for local outlier factor according to the characteristics of local outlier detection, and it can improve the efficiency of local outlier detection algorithm in parallel with the traditional method of improving the efficiency of nearest neighbor search. The experimental results show that the E2DLOS algorithm has higher detection accuracy for the scattered dataset, and RCMLQ has a significant effect on improving the time efficiency of the E2DLOS algorithm. In the experiments we can also find out that for the more closely distributed dataset, the RCMLQ algorithm is more effective in improving the efficiency of the local outlier detection. Moreover, the RCMLQ algorithm can also be applied to various local outlier detection methods to improve the efficiency by preprocessing the dataset. In the future, we will apply the above algorithms to various practical and complex environments for anomaly detection.

ACKNOWLEDGMENT

This paper was presented at the IEEE HPCC 2017 Conference [1].

REFERENCES

[1] S. S. Su *et al.*, "N2DLOF: A new local density-based outlier detection approach for scattered data," in *Proc. IEEE Int. Conf. High Perform. Comput. Commun., IEEE Int. Conf. Smart City, IEEE Int. Conf. Data Sci. Syst.*, Dec. 2017, pp. 458–465.

[2] E. M. Knorr and R. T. Ng, "A unified approach for mining outliers," in *Proc. Conf. Centre Adv. Stud. Collaborative Res.*, 1997, p. 11.

[3] E. M. Knorr, R. T. Ng, and V. Tucakov, "Distance-based outliers: Algorithms and applications," *VLDB J.*, vol. 8, nos. 3–4, pp. 237–253, 2000.

[4] S. Mehnaz and E. Bertino, "Ghostbuster: A fine-grained approach for anomaly detection in file system accesses," in *Proc. Conf. Data Appl. Secur. Privacy*, 2017, pp. 3–14.

[5] M. Iturbe, I. Garitano, U. Zurutuza, and R. Uribeetxeberria, "Towards large-scale, heterogeneous anomaly detection systems in industrial networks: A survey of current trends," *Secur. Commun. Netw.*, vol. 2017, no. 6, Nov. 2017, Art. no. 9150965.

[6] Y. Wang, Z. Wu, Y. Zhu, and P. Zhang, "Research on anomaly detection algorithm based on generalization latency of telecommunication network," *Future Gener. Comput. Syst.*, vol. 85, pp. 9–18, Aug. 2018.

[7] P. Gogoi, D. K. Bhattacharyya, B. Borah, and J. K. Kalita, "A survey of outlier detection methods in network anomaly identification," *Comput. J.*, vol. 54, no. 4, pp. 570–588, Apr. 2011.

[8] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Survey on incremental approaches for network anomaly detection," *Int. J. Commun. Netw. Inf. Secur.*, vol. 3, no. 3, pp. 226–239, Dec. 2012.

[9] D. Agarwal, "An empirical Bayes approach to detect anomalies in dynamic multidimensional arrays," in *Proc. IEEE Int. Conf. Data Mining*, Nov. 2005, pp. 1–8.

[10] M. A. F. Pimentel, D. A. Clifton, L. Clifton, and L. Tarassenko, "A review of novelty detection," *Signal Process.*, vol. 99, pp. 215–249, Jun. 2014.

[11] V. Avdiienko, K. Kuznetsov, I. Rommelfanger, A. Rau, A. Gorla, and A. Zeller, "Detecting behavior anomalies in graphical user interfaces," in *Proc. Int. Conf. Softw. Eng. Companion*, May 2017, pp. 201–203.

[12] E. Keogh, J. Lin, S.-H. Lee, and H. van Herle, "Finding the most unusual time series subsequence: Algorithms and applications," *Knowl. Inf. Syst.*, vol. 11, no. 1, pp. 1–27, 2010.

[13] L. Cai, N. Thornhill, S. Kuenzel, and B. C. Pal, "Real-time detection of power system disturbances based on k-nearest neighbor analysis," *IEEE Access*, vol. 5, pp. 5631–5639, 2017.

[14] A. Mccarren, S. Mccarthy, C. O. Sullivan, and M. Roantree, "Anomaly detection in agri warehouse construction," in *Proc. Australas. Comput. Sci. Week Multiconf.*, 2017, pp. 1–10.

[15] N. Stojanovic, M. Dinic, and L. Stojanovic, "A data-driven approach for multivariate contextualized anomaly detection: Industry use case," in *Proc. IEEE Int. Conf. Big Data*, Dec. 2018, pp. 1560–1569.

[16] G. Vidmar and R. Blagus, "Outlier detection for healthcare quality monitoring—A comparison of four approaches to over-dispersed proportions," *Qual. Rel. Eng. Int.*, vol. 30, no. 3, pp. 347–362, 2014.

[17] K. Yan, X. You, X. Ji, G. Yin, and F. Yang, "A hybrid outlier detection method for health care big data," in *Proc. IEEE Int. Conf. Big Data Cloud Comput.*, Oct. 2016, pp. 157–162.

[18] F. Gu, J. Niu, S. K. Das, Z. He, and X. Jin, "Detecting breathing frequency and maintaining a proper running rhythm," *Pervasive Mobile Comput.*, vol. 42, pp. 498–512, Dec. 2017.

[19] E. Schubert, A. Zimek, and H.-P. Kriegel, "Local outlier detection reconsidered: A generalized view on locality with applications to spatial, video, and network outlier detection," *Data Mining Knowl. Discovery*, vol. 28, no. 1, pp. 190–237, 2014.

[20] K. Zhang, M. Hutter, and H. Jin, "A new local distance-based outlier detection approach for scattered real-world data," in *Proc. Pacific–Asia Conf. Adv. Knowl. Discovery Data Mining*, 2009, pp. 813–822.

[21] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger, "The R*-tree: An efficient and robust access method for points and rectangles," in *Proc. ACM Int. Conf. Manage. Data (SIGMOD)*, Atlantic City, NJ, USA, vol. 19, no. 2, 1990, pp. 322–331.

[22] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Commun. ACM*, vol. 19, no. 9, pp. 509–517, 1975.

[23] A. Beygelzimer, S. Kakade, and J. Langford, "Cover trees for nearest neighbor," in *Proc. 23rd Int. Conf. Mach. Learn. (ICML)*, Pittsburgh, PA, USA, 2006, pp. 97–104.

[24] P. Ciaccia, M. Patella, and P. Zezula, "M-tree: An efficient access method for similarity search in metric spaces," in *Proc. 23rd Int. Conf. Very Large Data Bases (VLDB)*, Athens, Greece, 1997, pp. 426–435.

[25] S. T. Leutenegger, M. A. Lopez, and J. M. Edgington, "STR: A simple and efficient algorithm for R-tree packing," in *Proc. Int. Conf. Data Eng. (ICDE)*, Birmingham, U.K., Apr. 1997, pp. 497–506.

[26] D. M. Hawkins, *Identification of Outliers*. London, U.K.: Chapman & Hall, 1980

[27] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "LOF: Identifying density-based local outliers," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, Dallas, TX, USA, May 2000, pp. 93–104.

[28] M. Agyemang, "Local sparsity coefficient-based mining of outliers," Ph.D. dissertation, School Comput. Sci., Univ. Windsor, Windsor, ON, Canada, 2002.

[29] S. Papadimitriou, H. Kitagawa, P. B. Gibbons, and C. Faloutsos, "LOCI: Fast outlier detection using the local correlation integral," in *Proc. 19th Int. Conf. Data Eng.*, Mar. 2003, pp. 315–326.

[30] W. Jin, A. K. H. Tung, J. Han, and W. Wang, "Ranking outliers using symmetric neighborhood relationship," in *Proc. Pacific–Asia Conf. Adv. Knowl. Discovery Data Mining*, Singapore, 2006, pp. 577–593.

[31] J. Tang, Z. Chen, A. W. Fu, and D. W. Cheung, "Capabilities of outlier detection schemes in large datasets, framework and methodologies," *Knowl. Inf. Syst.*, vol. 11, no. 1, pp. 45–84, 2007.

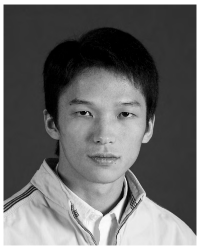
[32] H.-P. Kriegel, P. Kröger, E. Schubert, and A. Zimek, "LoOP: Local outlier probabilities," in *Proc. Conf. Inf. Knowl. Manage. (CIKM)*, Hong Kong, Nov. 2009, pp. 1649–1652.

[33] L. J. Latecki, A. Lazarevic, and D. Pokrajac, "Outlier detection with kernel density functions," in *Proc. Int. Conf. Mach. Learn. Data Mining Pattern Recognit. (MLDM)*, Leipzig, Germany, Jul. 2007, pp. 61–75.

[34] E. Schubert, A. Zimek, and H. P. Kriegel, "Generalized outlier detection with flexible kernel density estimates," in *Proc. SIAM Int. Conf. Data Mining*, 2014, pp. 542–550.

[35] T. D. Vries, S. Chawla, and M. E. Houle, "Density-preserving projections for large-scale local anomaly detection," *Knowl. Inf. Syst.*, vol. 32, no. 1, pp. 25–52, 2012.

[36] E. Schubert, A. Zimek, and H. P. Kriegel, "Fast and scalable outlier detection with approximate nearest neighbor ensembles," in *Proc. Int. Conf. Database Syst. Adv. Appl.* Cham, Switzerland: Springer, 2015, pp. 19–36.
[37] H.-P. Kriegel, E. Schubert, and A. Zimek, "The (black) art of runtime evaluation: Are we comparing algorithms or implementations?" *Knowl. Inf. Syst.*, vol. 52, no. 2, pp. 341–378, 2017.
[38] G. O. Campos *et al.*, "On the evaluation of unsupervised outlier detection: Measures, datasets, and an empirical study," *Data Mining Knowl. Discovery*, vol. 30, no. 4, pp. 891–927, 2016.
[39] H.-P. Kriegel, M. Schubert, and A. Zimek, "Angle-based outlier detection in high-dimensional data," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2008, pp. 444–452.
[40] T. Hu and S. Y. Sung, "A trimmed mean approach to finding spatial outliers," *Intell. Data Anal.*, vol. 8, no. 1, pp. 79–95, 2004.
[41] M. Ester, H. P. Kriegel, J. Sander, and X. Xu, "A density based algorithm for discovering clusters in large spatial data sets with noise," in *Proc. 2nd Int. Conf. Knowl. Discovery Data Mining*, 1996, pp. 226–231.
[42] S. Hettich and S. D. Bay, *The UCI KDD Archive*. Irvine, CA, USA: Univ. of California, Department of Information and Computer Science, 1999. Accessed: May 28, 2018. [Online]. Available: http://kdd.ics.uci.edu
[43] M. Kontaki, A. Gounaris, A. N. Papadopoulos, K. Tsichlas, and Y. Manolopoulos, "Continuous monitoring of distance-based outliers over data streams," in *Proc. IEEE Int. Conf. Data Eng.*, Apr. 2011, pp. 135–146.
[44] Pacific Marine Environmental Laboratory. *Pacific Ocean TAO*. Accessed: May 26, 2018. [Online]. Available: http://www.pmel.noaa.gov/tao/
[45] F. Angiulli and F. Fassetti, "Detecting distance-based outliers in streams of data," in *Proc. 16th ACM Conf. Inf. Knowl. Manage. (CIKM)*, Lisbon, Portugal, Nov. 2007, pp. 811–820.
[46] *UPenn Wharton Research Data Services, Stock*. Accessed: May 26, 2018. [Online]. Available: https://wrds-web.wharton.upenn.edu/wrds/
[47] L. APACao, D. Yang, Q. Wang, Y. Yu, J. Wang, and E. A. Rundensteiner, "Scalable distance-based outlier detection over high-volume data streams," in *Proc. IEEE Int. Conf. Data Eng.*, Mar. 2014, pp. 76–87.

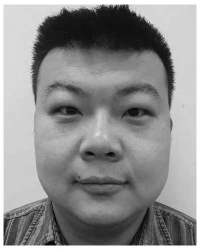


SHUBIN SU was born in Fujian, China, in 1987. He received the B.S. degree in communication engineering from Quanzhou Normal University, Quanzhou, Fujian, in 2010, and the M.S. degree in computer technology from the Jiangxi University of Science and Technology, Ganzhou, Jiangxi, in 2014. He is currently pursuing the Ph.D. degree with the School of Computer Science and Technology, Beihang University, Beijing. His research interests include data mining, artificial intelligence, cloud computing, computer architecture, and so on.

LIMIN XIAO received the B.S. degree from Tsinghua University, Beijing, China, in 1993, and the M.S. and Ph.D. degrees from the Institute of Computing, Chinese Academy of Sciences, Beijing, in 1996 and 1998, respectively, all in computer science. He is currently a Professor with the School of Computer Science and Engineering, Beihang University. His main research areas are computer architecture, computer system software, high performance computing, virtualization, and cloud computing. He is a Senior Member of the China Computer Federation.

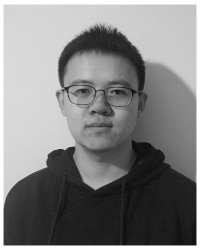
LI RUAN received the Ph.D. degree in computer science from the Institute of Software Chinese Academy of Sciences, Beijing, China, in 2009. She was a Visiting Scholar with the Illinois Institute of Technology, Chicago, IL, USA, in 2012. She is currently a Lecturer with the School of Computer Science and Engineering, Beihang University, China. Her main research areas are virtualization and cloud computing, computer system software, and high performance computer. She is a Senior Member of the China Computer Federation.

FEI GU received the B.S. degree in software engineering from Yangzhou University, in 2011, and the M.S. degree in computer science and technology from the Nanjing University of Aeronautics and Astronautics, in 2014. He is currently pursuing the Ph.D. degree in computer science with Beihang University, Beijing, China. His research focuses on sensor network, healthcare system, data mining, and so on.



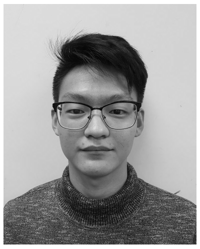
SHUPAN LI received the B.S. degree in computer science from National Chung Cheng University, Kaifeng, Henan, China, in 2008, and the M.S. degree in computer science from the University of Science and Technology of China, Hefei, Anhui, China, in 2011. He is currently pursuing the Ph.D. degree in computer science with Beihang University, Beijing, China.

His research focuses on system virtualization, file system, computer architecture, and information. He is a member of the China Computer Federation.



ZHAOKAI WANG received the B.S. degree in software engineering from the University of Electronic Science and Technology of China, Sichuan, China, in 2017. He is currently pursuing the M.S. degree in computer science with Beihang University, Beijing, China.

His research interests include the machine learning and computer vision.



RONGBIN XU was born in Shandong, China, in 1995. He received the B.S. degree in computer science and technology from the University of Electronic Science and Technology of China.

He is currently pursuing the M.S. degree in computer science and technology with Beihang University, China. His research interests include the machine learning and computer architecture.