IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received January 8, 2019, accepted January 17, 2019, date of publication January 25, 2019, date of current version February 12, 2019.

*Digital Object Identifier 10.1109/ACCESS.2019.2895094*

# Applying Anomaly Pattern Score for Outlier Detection

**CHAO WANG**[1,2], **ZHEN LIU**[1,2], **HUI GAO**[1,2], **AND YAN FU**[1,2]
[1]School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China
[2]Big Data Research Center, University of Electronic Science and Technology of China, Chengdu 611731, China

Corresponding authors: Chao Wang (chaosimpler@gmail.com) and Zhen Liu (quake.liu0625@gmail.com)

This work was supported by the National Natural Science Foundation of China under Grant 61673085 and Grant 21673034.

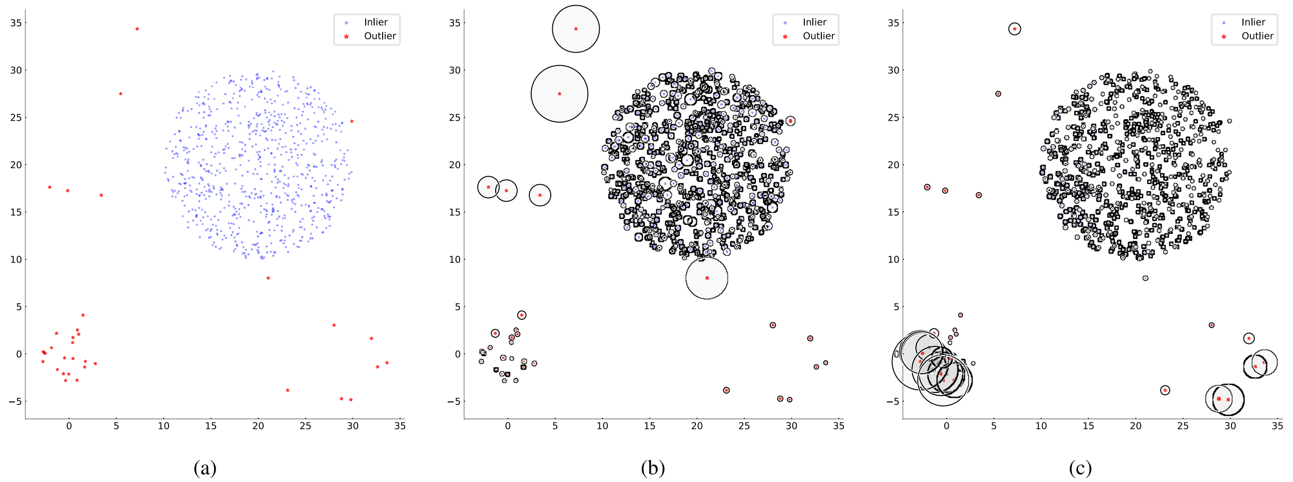
**ABSTRACT** Outlier detection is an important sub-field of data mining and studied intensively by researchers in the past decades. For neighborhood-based outlier detection methods like KNN and LOF, different settings in the number of neighbors (indicated by a parameter $k$) would greatly affect the model's performance. Thereby, there are some recent studies which focus on identifying the optimal value of $k$ by analyzing the global or local structure of the dataset. But, we argue that neighborhood-based outlier detection model could obtain an improvement in performance without parameter tuning. In this paper, from a novel angle of view, we adopt a uniform sampling strategy to generate a series of local proximity graphs and propose a new adaptive outlier detection model named anomaly pattern score which does not rely on the $k$ tuning. In addition, the theoretical analysis of the effectiveness of the proposed model is conducted as well. The extensive experiments on both synthetic and real-world datasets show that the proposed model outperforms the state-of-the-art algorithms on most datasets.

**INDEX TERMS** Adaptive outlier detection, adaptive anomaly detection, neighborhood-based model, Markov random walk, local proximity graph.

## I. INTRODUCTION

Outlier detection as one of the key branch of data mining is designed to automatically discover rare observations, events or patterns hidden in the dataset, which are dissimilar with the majority of the data [1]. It has been applied in a wide range of domains such as finding opinion spam in online review systems [2]–[4], detecting anomalies in video surveillance [5], [6], revealing suspicious patterns in medical images [7], [8], and many others.

Due to the broadly applications in both industry and academia, a large number of anomaly detection methods have emerged in the past few decades. Model based approaches [9], [10] construct a statistical model to represent the majority of data objects, while the observations that does not fit the model are considered as the outliers. This type of methods become incapable when the patterns hidden in the data change frequently. Cluster based approaches [11]–[14] take the outliers as the byproducts of a clustering process. However, as an unexpected result, these methods would wrongly identify a set of outlier objects with higher similarities as the normal ones since they will be grouped into a cluster. Neighborhood based approaches utilize a specific proximity measure (e.g. distance, density etc.) on the neighborhood around each object to design a scoring model, which calculates an outlier score to discriminate different types of objects. Methods under this schema attract the most attention because of their simplicity and flexibility. Knox and Ng [15] first modify the $k$NN framework to adapt to the problem of outlier detection. The distance from a specific object to its $k$-th nearest neighbor is directly used to represent its outlier-ness. This method is simple but efficient to find the outliers in evenly distributed datasets, but it is incapable to find an appropriate $k$ value to capture the outliers in datasets with different densities. Breunig *et al.* [16] proposed the Local Outlier Factor to solve this issue by formulating a density definition on the neighborhood around each object. An outlier object will be assigned with a LOF score far larger than 1. In contrast, the objects lie in a evenly distributed area will get a LOF score near 1, which indicates their lower chance to be outlier. COF [17] utilized an average chain distance to estimate the local density, which solves the problem that an outlier object could not be distinguished if it is close to a sparsely distributed inlier cluster. INFLO [18] uses a enlarged neighborhood definition including the $k$NN and reverse $k$NN compared with the LOF model. RDOS [19] extended the neighborhood definition with a



(a)

(b)

(c)

**FIGURE 1.** **A synthetic dataset with 750 inlier objects clustering in a circle and 35 outlier objects scattered around. The outlier score for each data object is calculated by LOF algorithm with different values of *k*. The circles around the objects represent their outlier scores. The scatter plot of the dataset is provided in Fig 1(a), and Fig. 1(b) shows the outlier scores computed with the parameter *k* set to 3. Fig. 1(c) shows the outlier scores computed by using *k* set to 770. (a) A synthetic dataset. (b) $k = 3$. (c) $k = 770$.**

shared neighborhood set, furthermore, it also adopts a kernel density estimation to estimate the local density.

On the other hand, graph-based approaches are receiving increasing attention, owing to their robust expressiveness for various type of data [20], and their capabilities to effectively capture the potential connections implicit in the dataset [21]. ODIN [22] determines the outlier-ness of an object by the in-degree of its corresponding node in a directed unweighted graph, which is constructed by using the $k$-nearest neighbors of each object in the dataset. OutRank [23], [24] employs a Markov random walk process on a fully connected graph. After the stochastic process reaches equilibrium, the values in the stationary vector are taken as the outlier scores. A small score represents that the related object has a lower possibility to be visited by the random walker, thus indicates it has a higher chance to be an outlier. HCOD [25] uses a similar strategy with OutRank, the difference between them is that it iteratively split the graph by using the Fiedler vector of the corresponding transition matrix, then the random walk process is applied on each of the subgraphs, which are naturally treated as the local information. LIGRW [26] explicitly constructs an asymmetric directed weighted graph to capture the local information, moreover two different types of restart vector are devised to ensure that an outlier object would get more weight to be visited by the random walker.

One of the major challenges for algorithms using local information is the problem of how to choose an appropriate neighbor size (usually denoted by a parameter $k$). The value for the parameter $k$ greatly affects the performance of the related algorithms. We apply the LOF algorithm on a synthetic dataset to demonstrate this issue. In Fig. 1, we could see that with a small value setting to the parameter $k$, the algorithm could not detect the outlier objects that form a cluster (Fig. 1(b)). While if it is set too large, the outlier objects near the inlier cluster could not get enough scores to be detected (Fig. 1(c)). The choice for the value of the parameter $k$ depends highly on the priori knowledge of the dataset, and it is never an easy task for even an experienced user to choose the appropriate value.

To solve this particular problem, Ha *et al.* [27] proposed a new model (INS) using the instability factor to calculate the outlier score for each object, which aims to reduce the sensitivity of the algorithm to the parameter $k$. Bhattacharya *et al.* [28] utilized the Daubechies wavelets to search the optimal parameter. While Zhu *et al.* [29], [30] and Ning *et al.* [31] try to acquired the appropriate value of the parameter $k$ by exploring either the natural neighborhood or stable state of the neighborhood.

To the best of our knowledge, currently, there has no studies bringing the characteristics of the random walk based graph models into account to determine the suitable local information. Inspired by the different change patterns on the probabilities for different types of objects being visited by the random walker on local information graphs constructed by various $k$ values, in this study, we proposed an adaptive outlier detection model named Anomaly Pattern Score (APS). The proposed APS model constructs a set of local proximity graphs from the original dataset by using a sequence of automatically determined neighbor size. A Markov random walk process is performed on the predefined graphs utilizing the different aspects of local information to calculate the stationary distribution vector, which represents the visiting probability for each object. Then the anomaly pattern score is deduced from the multiple stationary vectors. Unlike those methods that aiming to search for the optimal value of the parameter $k$, the proposed APS model analyzes multiple local information and utilizes the differences between them to characterize the outlier-ness. Experimental results on both synthetic and real-world datasets shows that the proposed model obtains improvements simultaneously on the measures of ROC AUC and Precision against the state-of-the-art approaches. The main contributions of this study could be summarized as following:

1) We analyze the characteristics of the random walk based approaches, and find out that the visiting probabilities of different types of objects will show different patterns as the proximity measure and the neighbor size are chosen differently to construct the transition matrix.
2) We design a uniform sampling strategy to automatically generate a sequence of values for constructing various local proximity graphs. By applying the Markov random walk process on multiple graphs, we deduce the anomaly pattern score for each object to discriminate the outliers from inlier objects, and give the theoretical analysis on the effectiveness of the proposed model as well.
3) We conduct extensive experiments on both synthetic and real-world datasets (57 datasets in total) , from which the results show that the proposed APS model outperforms state-of-the-art approaches.
4) We analyze the flexibility of the proposed APS model on five high-dimensional datasets. Moreover, the stability of the proposed model are also analyzed. It shows that the experimental results are quite stable and not sensitive to the number of local proximity graphs used in the model, representing an outstanding advantage of our model.

The rest of the paper is organized as follows: in section 2 we introduce the related works and some preliminaries. Then the proposed APS model is described in section 3. The corresponding algorithm is introduced in section 4. Empirical study and analysis work are presented in section 5. The conclusion is given in the last section.

## II. PRELIMINARIES AND RELATED WORKS

In this section, we bring the brief introduction about the related conceptions in random walk based graph models, which forms the basis of the proposed APS model. Furthermore, several algorithms that aim to search for the optimal value for the parameter $k$ are also introduced, which will be used in the later section to compare with the proposed APS model.

OutRank firstly proposed to use the stationary distribution of a random walk process to represent the outlier-ness of each object. It constructed a undirected fully connected graph by computing the pairwise similarities of each object, then a Markov random walk process is applied to calculate the stationary distribution. The values in the stationary distribution vector are utilized to denote the visiting probabilities of a random walker on each object. The lower probability that an object is visited by the random walker indicates it has a higher chance to be an outlier. This process is defined from a global perspective, which can be thought as using all of the neighbors to form the local information.

HCOD introduces the Fieldler vector to iteratively split the original fully connected graph into a sequence of subgraphs, which are then used as the local information. Although in theory, it does not depend on any parameters of neighbor size to form the local information, in practical applications, it is often difficult to achieve satisfying results.

LIGRW uses Heat Kernel to calculate the similarities between each of the object, based on which a local information graph is defined to capture the asymmetric relationships between different types of data objects. Then a customized random walk process along with two individual restart vector is applied on the predefined local information graph to ensure that the outlier objects could get more chance to be visited by the random walker. Unlike OutRank and HCOD, in LIGRW, a larger value in the stationary distribution vector indicates the related object has a higher chance to be an outlier.

Given a dataset $D$ with $d$ features and $n$ objects, $D = \{x_1, x_2, \cdots, x_n\}, x_i \in \mathbb{R}^d$, the adjacent matrix of the $k$-neighborhood graph is defined as following:

$$A_{kng} = \begin{bmatrix} w(1,1) & w(1,2) & \dots & w(1,n) \\ w(2,1) & w(2,2) & \dots & w(2,n) \\ \vdots & \vdots & \ddots & \vdots \\ w(n,1) & w(n,2) & \dots & w(n,n) \end{bmatrix} \tag{1}$$

where $w(i,j)$ denotes the weight on the edge directed from node $j$ to $i$, which can be calculated as:

$$w(i,j) = \begin{cases} \mathrm{sim}(i,j), & \texttt{if} \quad \texttt{j} \in \texttt{knn(i)} \\ 0, & \texttt{otherwise} \end{cases} \tag{2}$$

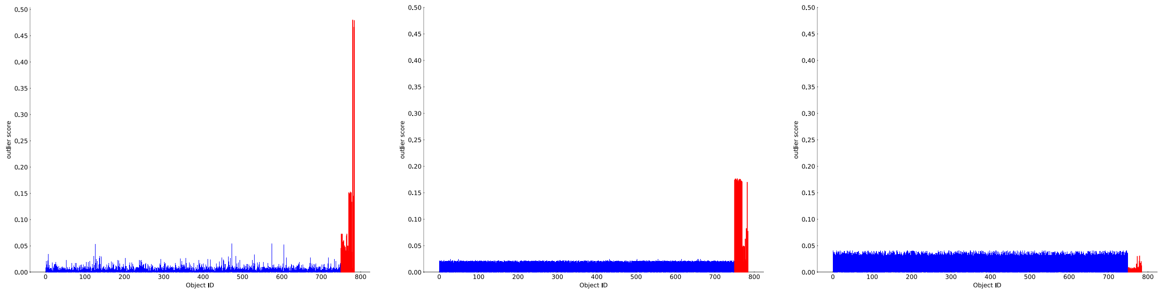
where $\mathrm{sim}(i,j)$ denotes the similarity between objects $i$ and $j$.

A neighborhood graph describes the local information around each object. With a Markov random walk process defined on it, the LIGRW model utilized the values in stationary distribution of the stochastic process to represent the outlier score for each object. That is to say, LIGRW combines the local information with a global/local restart vector to effectively detect the outliers implicit in the dataset. However, it still suffers the problem of choosing the appropriate parameter of neighbor size. Without a proper value for the parameter $k$, the performance of the model will deteriorate drastically.

INS [27] was proposed to alleviate the sensitivity of parameter selection in neighborhood-based methods. It first defines a center of gravity for a given neighborhood of a specific data object. Then the change of gravity centers with different $k$ values are integrated to form a instability factor to represent the outlier-ness for each object.

*Definition 1: Gravity Center: the gravity center of object i is defined as the centroid of the objects which belong to the k-th nearest neighbors of i.*

Given a parameter $k$, the instability factor of $i$ is defined as the sum of gravity center changes for the different neighborhoods of object $i$ by using a set of values with the range from 1 to $k$. If the value of $k$ is large enough, the performance of the INS tends to be stable. In this way, the INS model achieves the goal of alleviating the sensitivity to the parameter of neighbor size. However, the algorithm still needs to set the



(a)

(b)

(c)

**FIGURE 2. Applying random walk based model on the synthetic dataset with different *k* settings, which is introduced in Fig. 1. Values in the x-axis represent the ID of the objects, and the bars in the plot represent the outlier score for each of the object. Scores for the outlier objects are denoted by the red bars, and the inlier objects are denoted by the blue bars. (a) $k = 10$. (b) $k = 400$. (c) $k = 785$.**

parameter manually. Besides, it is hard for the algorithm to find an appropriate parameter to detect both of the local and global outliers simultaneously.

Different from the idea of INS, NOF [32] calculated the Natural Outlier Factor by replacing the k-nearest neighborhood in the typical LOF model with a *nature influence space*, which equals to the union of the nearest neighborhood and the reverse nearest neighborhood according to an automatic determined *Nature Value*. Similarly, NaNE [30] used a *Nature Neighbor Eigenvalue* to estimate the optimal $k$ value, which could be reached by calculating the stable searching state. MNGApk [31] utilize a mutual neighbor graph based approach to solve the neighbor size selection problem. Their algorithm builds a collection of mutual neighbor graphs using $k$ values ranging from 1 to N-1. As the value of $k$ increases monotonically, when the *Mutual Neighbor Graph* reaches its first stable state (i.e. the number of complete subgraphs does not change), the corresponding value of $k$ is considered as the appropriate $k$. After the value is obtained, a neighbor-based outlier detection algorithm can be applied to calculate the outlier score.

All of the above methods focus on searching for the optimal neighbor size by investigating the inner relationships of the dataset, which is achieved by constructing the corresponding models (e.g. Nature Value, Nature Eigenvalue, Mutual Graph etc.). Unlike the aforementioned studies, our model is free of $k$ parameter and, as a result, not necessary to find the optimal value of the $k$. Instead, we focus on using the multiple proximity graphs to generate the model which can also guarantee the performance of outlier detection.

## III. THE ANOMALY PATTERN SCORE MODEL

In this section, we first investigate the characteristics of the random walk based approach, then we describe the details of the proposed APS model. The effectiveness of the model is also analyzed from a theoretical perspective.

### *A. THE PARTICULAR CHARACTERISTICS OF THE RANDOM WALK BASED APPROACHES*

Outlier detection methods based on random walk process are obviously different from other types of methods, in which the structure of the graph related to the stochastic process directly affects its stationary distribution. This means that, the method used to construct the graph determines the limiting behavior of random walk process, thus the outputting graphs could have a direct impact on the probabilities of objects being visited by the random walker.

The first major problem is which proximity measure should used to characterize the interrelationships between data objects. In some real-wold applications, different proximity measures may be needed according to the specific scenarios. For example, Euclidean distance can effectively capture the differences between objects in most cases. However, when the dimension of the dataset increases, Kriegel *et al*. [33] have shown that the variances in angle between high-dimensional feature vectors are more sensitive than directly using Euclidean distances. In this case, cosine similarity can achieve better results. In order to ensure the effectiveness of the results, most random walk based methods use a fixed proximity measure when constructing the transition matrix, which limits the flexibility of the method.

On the other hand, applying the random walk process on a properly constructed local information graph can be expected to get excelling results than directly using the global information for the task of outlier detection [26]. Using different neighbors when constructing the local information graphs can have a significant impact on the outputting outlier scores. We applied the LIGRW algorithm on the synthetic dataset shown in Fig. 1(a), using different neighbor sizes. The results are shown in Fig. 2.

From the results, we can see that as the value of $k$ increases, the scores of the inlier objects will increase, while the scores of the outlier objects shows a trend of decline. According to [26], when the local information graph is constructed with a small $k$, in most cases, there are only inliers existing in the neighborhood of a inlier object, while there may be some inliers exist in the neighborhood of an outlier object. In another word, for an inlier object, there may exists some outlier objects would take it as their neighbors. On the contrary, there are few inlier objects would take an outlier object as their neighbors. Therefore, the random walker could jump from a inlier object to an outlier object. But when the random walker visits an outlier object, the probability that it jumps to an outlier is much higher than that it jumps to an inlier. In this case, outliers are visited at relatively high frequencies, their corresponding scores in the stationary distribution increase sharply (Fig. 2(a)).

In contrast, as the value of $k$ increases, the connectivity in the local information graph begins to increase, especially the edges between the inlier objects. This results in a higher chance for the random walker to wander between inlier objects. Therefore, the scores of inlier objects in the stationary distribution will increase together, meanwhile the scores of outlier objects will decrease. And the increases for the scores of the inlier objects will be constrained (Figs. 2(b) and 2(c)).

Using a specified proximity measure, the visiting probabilities of different types of objects demonstrate different change patterns as the value of $k$ changes, which constitutes the main idea of our proposed APS model.

### B. THE ANOMALY PATTERN SCORE

According to the definition of outlier detection [1], in most real-world applications, the number of outliers is far smaller than that of the inliers. Combined with the characteristic of the visiting probability changes for different types of objects, we could infer that when the probability of each inlier object increases as the value of $k$ changes, its growth rate will be constrained due to the large number of inliers. Conversely, when the visiting probabilities of most inlier objects begin to decrease, those for the outlier objects will increase very fast for their small number compared with the vast number of inliers.

When applying the random walk process on graphs constructed with different neighbor sizes, the visiting probabilities of different types of objects show different change patterns. In the following we will describe how to model these patterns in detail.

*Definition 2: Local Proximity Graph: a local proximity graph is weighted directed graph, which is constructed under the assumption that if and only if object j lies in the neighborhood of i, there is an edge directed from i to j.*

It is worth noticing that the Local Proximity Graph can be seen as a super-set of the local distance graph and the local similarity graph. It does not depend on a specific proximity measure, which guarantees its flexibility to various application scenarios.

In order to characterize the changes of visiting probabilities under different neighbor sizes, in this study, a uniform sampling strategy is adopted to automatically generate a set of values for the parameter $k$, which is formulated as following:

$$k_\alpha = \begin{cases} 2, & \alpha = 0 \\ k_{\alpha-1} + \lceil \frac{n}{m} \rceil, & 1 \leq \alpha < \mathtt{m} \end{cases} \tag{3}$$

where $m$ represents the number of proximity graphs, $0 < \alpha < m$, and $n$ is the number of objects in the datasets.

As long as the number of graphs is specified, the sampling process described above could help to automatically obtain $m$ values for the parameter of neighbor size, which are evenly distributed across the sample spaces.

Let $G_\alpha$ denotes the local proximity graph constructed by using the parameter $k_\alpha$, $A_\alpha$ represents its adjacent matrix, then the transition matrix can be calculated by normalizing the adjacent matrix by column:

$$T_\alpha = A_\alpha \cdot D^{-1} \tag{4}$$

where $D$ is a diagonal matrix, and $D(i,i) = \sum_j A_\alpha(j,i)$.

Using local information may cause the proximity graph to be split into several isolated subgraphs, which is known as the *dangling link problem*. Therefore, a random walk with restart process is adopted to solve this issue. The probability of the object $i$ being visited after the stochastic process reaching equilibrium on proximity graph $G_\alpha$ can be computed by using the following iterative method:

$$p_\alpha^{(t+1)}(i) = \gamma \cdot \frac{1}{n} + (1-\gamma) \sum_{j \in X} T_\alpha(i,j) \cdot p_\alpha^{(t)}(j) \tag{5}$$

where $p_\alpha^{(t)}(i)$ denotes the probability which is calculated for object $i$ at step $t$ of the process, $\gamma$ is a restart factor satisfying $0 < \gamma < 1$, and usually setting to the value of 0.15. $n$ represent the number of objects in the dataset.

*Definition 3: Deviation Value: The deviation value of an object on a specific local proximity graph is defined as the difference between its corresponding value and the minimum value in the stationary distribution of a random walk, which is performed on the graph.*

The deviation value of object $i$ on proximity graph $G_\alpha$ can be computed as:

$$\mathcal{D}_\alpha(i) = p_\alpha(i) - \min_{j \in X}(p_\alpha(j)) \tag{6}$$

where $p_\alpha(i)$ denotes the $i$-th value in the stationary distribution vector of a random walk process, which is performed on the local proximity graph $G_\alpha$.

*Definition 4: Anomaly Pattern Score: the anomaly pattern score of an object is defined as the average of the deviation values it obtains on all of the local proximity graphs.*

The anomaly pattern score for object $i$ on $m$ automatically generated local proximity graphs can be calculated as following:

$$S(i) = \frac{\sum_{\alpha \in m} \mathcal{D}_\alpha(i)}{\sum_{j \in X} \sum_{\alpha \in m} \mathcal{D}_\alpha(j)} \tag{7}$$

where $X$ denotes the original dataset.

According to the above definition, the proposed APS model is based on the local proximity graph, which ensures its adaptiveness on different proximity measures. Furthermore, it automatically generate a sequence of neighbor sizes instead of specifying the parameter $k$ manually. The anomaly pattern score of an object represents the average deviation values of its visiting probabilities on various random walk processes, which are applied on different local proximity graphs. In most case, as the number of outlier objects are much smaller than the inlier objects, their anomaly pattern scores will be much larger than the inlier objects. Therefore, an object with a larger anomaly pattern score represents that its visiting probabilities on different local proximity graphs change drastically, which indicates it has a higher chance to be an outlier.

### C. THEORETICAL ANALYSIS

*Assumption 1: For most applications, compare to the significant differences between the outlier objects, the similarities between inlier objects are relatively large.*

According to the above assumption, there would exists high similarities between the inlier objects, each inlier object will be visited by the random walker with an approximate chance. Therefore, after the random walk process reaches equilibrium, the visiting probabilities for the inlier objects should roughly near the same values.

*Theorem 1: For a given inlier object $I$ and an outlier object $O$, the proposed APS model will assign a relatively larger anomaly pattern score to object $O$ than $I$.*

*Proof:* When the neighbor size $k$ is set to a small value $\mu$, there may exists an edge directed from $I$ to $O$ in the local proximity graph $G_\mu$. Since the similarities between inlier objects is relative higher, the random walker could only transit from another inlier object to node $I$. As a result, the visiting probability of $I$ is smaller than the outlier object $O$, which can be formulated as following:

$$p_\mu(O) \geq p_\mu(I) \tag{8}$$

On the contrary, when a large value $\xi$ is set to the neighbor size, there may also exist some edges directed from an outlier node to a inlier node. Under this circumstance, once the random walker jumps to a inlier node, it will wander between inlier objects for a long time, thus the visiting probabilities for the inlier objects will increase.

$$p_\xi(O) \leq p_\xi(I) \tag{9}$$

As mentioned above, the visiting probabilities for all inlier objects should roughly reach the same value. Therefore, when the visiting probabilities for the inlier objects increase, their growth are constrained, which makes the following equation hold:

$$p_\xi(I) - p_\xi(O) \leq p_\mu(O) - p_\mu(I) \tag{10}$$

According to the definition of the anomaly pattern score in Eq. 7,

$$\begin{aligned}
&\text{APS}(O) - \text{APS}(I)\\
&= \frac{\sum \mathcal{D}_\alpha(O)}{\sum_{j\in X}\sum \mathcal{D}_\alpha(j)} - \frac{\sum \mathcal{D}_\alpha(I)}{\sum_{j\in X}\sum \mathcal{D}_\alpha(j)}\\
&= \frac{\sum \mathcal{D}_\alpha(O) - \sum \mathcal{D}_\alpha(I)}{\sum_{j\in X}\sum \mathcal{D}_\alpha(j)}\\
&= \frac{\sum [p_\alpha(O) - \min p_\alpha(i)] - \sum [p_\alpha(I) - \min p_\alpha(i)]}{\sum_{j\in X}\sum \mathcal{D}_\alpha(j)}\\
&= \frac{\sum [p_\alpha(O) - p_\alpha(I)]}{\sum_{j\in X}\sum \mathcal{D}_\alpha(j)}\\
&= \frac{\sum p_\alpha(O) - \sum p_\alpha(I)}{\sum_{j\in X}\sum \mathcal{D}_\alpha(j)}\\
&= \frac{[p_\mu(O) + \cdots + p_\xi(O)] - [p_\mu(I) + \cdots + p_\xi(I)]}{\sum_{j\in X}\sum \mathcal{D}_\alpha(j)}
\end{aligned}$$

According to Eq. 10, we could infer that

$$p_\mu(O) + \cdots + p_\xi(O) \geq p_\mu(I) + \cdots + p_\xi(I)$$

In addition, by the definition of deviation value in Eq. 6,

$$\sum_{j\in X}\sum \mathcal{D}_\alpha(j) \geq 0$$

Therefore,

$$\text{APS}(O) - \text{APS}(I) \geq 0$$

which indicates that the anomaly patter score for an outlier object will be greater than or equal to that for an inlier object. □

## IV. ALGORITHM

**Algorithm 1** APS Algorithm

```
Input: Dataset $X \in \mathbb{R}^{N\times d}$, $\beta$ represents the restart probability of the Markov random walk process, default set to 0.15, $m$ represents the number of graphs, default set to 50.
Output: $\vec{S} \in \mathbb{R}^N$: a vector contains the anomaly pattern score for each object.
for $\alpha = 1$ to $m$ do
    calculate $k_\alpha$ using Eq. 3.
    calculate the adjacent matrix $A_\alpha$ of the local proximity graph.
    calculate the transition matrix $T_\alpha$ using Eq. 4

    ▷ calculate the visiting probability of each object
    $\vec{\eta} \leftarrow [1/N, \cdots, 1/N]$
    $\vec{\mu} \leftarrow [1/N, \cdots, 1/N]$
    while not converged do
        $\vec{p} \leftarrow \beta \cdot \vec{\eta} + (1-\beta) \cdot T_\alpha \cdot \vec{\mu}$
        $\vec{\mu} \leftarrow \frac{\vec{p}}{\|\vec{p}\|_1}$
    end while

    ▷ calculate the deviation value of each object
    $\mathcal{D}_\alpha(i) = \vec{p}(i) - \min(\vec{p})$
end for

▷ calculate the outlier score
for $i = 1$ to $N$ do
    $\vec{S}(i) \leftarrow$ calculating the anomaly pattern score using Eq. 7.
end for
Return $\vec{S}$
```

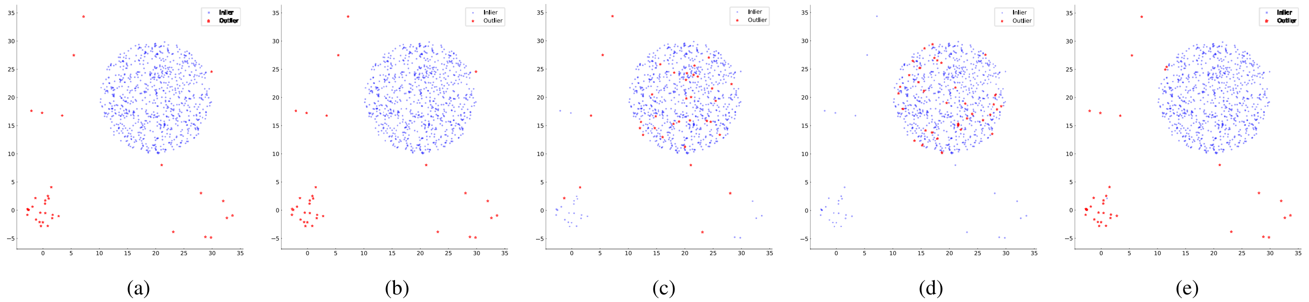


**FIGURE 3.** Comparison on synthetic dataset 1. The top 35 objects with largest outlier scores are marked as outliers. (a) APS. (b) MNGApk(LOF). (c) NaNE(LIGRW). (d) HCOD. (e) OutRank.

The proposed APS algorithm consists of two major steps. The first step is to construct different local proximity graphs using a specific proximity measure and various $k$ values, then apply a Markov random walk process on these graphs to calculate the stationary visiting probability. Next, we calculate the deviation values on each of the local proximity graphs for every object. In the second step, we combine the deviation values of each object on all of the graphs to obtain the anomaly pattern scores.

For a specified local proximity graph, the time complexity for calculating its stationary distribution is $O(N^2)$, thus, the time complexity for the first major step is $O(m \cdot N^2)$. The second step has a time complexity of $O(N)$. To sum these up, the overall time complexity for the APS model is: $O(m \cdot N^2)+O(N) \approx O(m \cdot N^2)$. In addition, there is no sequential dependencies when calculating the deviation values on different local proximity graphs, therefore, the proposed APS algorithm could easily extended to the parallel paradigm to accelerate the calculation process.

## V. EMPIRICAL STUDY AND ANALYSIS

In this section, we conduct experiments on two synthetic datasets and 57 real-world datasets to compare the proposed APS model against four state-of-the-art algorithms. Then, the adaptiveness of the APS model by using different proximity measures, and its sensitivity to the number of the local proximity graphs adopted are analyzed, respectively.

### A. EXPERIMENT SETUP

Both MNGApk and NaNE are proposed to search for the optimal value of neighbor size, which could be used to combine with a neighborhood based algorithm. In the following experiments, we combine the result from MNGApk with LOF, which is one of most distinguished neighborhood based algorithm, to form a new method MNGApk(LOF). The value returns from NaNE is chosen as the neighbor size of the LIGRW algorithm, which can be seen as a representative random walk based graph model. This constitutes another new algorithm NaNE(LIGRW). With this setup, neither MNGApk(LOF) nor NaNE(LIGRW) depend on a user-specified parameter of neighbor size. In addition, HCOD which automatically calculates the local information from the original dataset, and OutRank which can be thought of as using all of the neighbors of the object as local information, are also being selected to compare the proposed APS model.

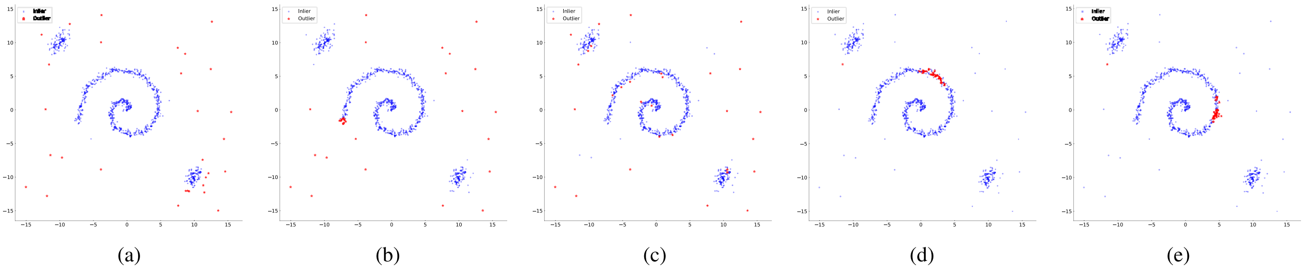
It is worth noticing that both HCOD and OutRank take the objects with smaller scores as outliers. For the sake of comparison, we used a linear transformation to convert their outlier scores so that a larger score indicates the related object has a higher chance to be an outlier. Let $\vec{S}_o$ be a vector with the same size as the number of objects in the dataset, which contains the original outlier score for each object. $\vec{S}_o(i)$ denotes the original score of object $i$, then the converted score $\vec{S}_n(i)$ can be calculated as following:

$$\begin{aligned} \vec{T} &= -1 \times \vec{S}_o \\ \vec{S}_n(i) &= \frac{\vec{T}(i) - \min \vec{T}}{\max \vec{T} - \min \vec{T}} \end{aligned} \tag{11}$$

### B. SYNTHETIC DATASETS

The first dataset is introduced in Fig. 1, which contains a circular-shape dense cluster with 750 inlier objects, and 35 randomly distributed outlier objects. 21 outlier objects in the left bottom corner form a sparse cluster. The aforementioned five algorithms are applied on the dataset without need of any parameter to indicate the neighbor size. We select the top 35 objects with largest scores as outliers. The results are shown in Fig. 3. From the results, we can see that the proposed APS as well as the MNGApk(LOF) algorithm correctly detected all of the outlier objects. The OutRank algorithm wrongly classifies two inlier objects in the upper left of the cluster as outliers. Both NaNE(LIGRW) and HCOD do not correctly detect the outlier objects within the sparse cluster. Besides, some inliers in the middle of the inlier cluster are also mistakenly recognized as outliers.

The second dataset consists of three inlier clusters, including a spiral-shape cluster with 550 inlier objects and two Gaussian clusters centered at (−10,10) and (10, −10), respectively, each of which contains 100 inlier objects. There are 30 outlier objects distributed in a field of (−15,15), (−15,15). We mark the top 30 objects with largest score as outliers, the results are shown in Fig 4. From the results, we can see that MNGApk(LOF) incorrectly marked the objects in the tail of the spiral cluster as outliers. NaNE(LIGRW) recognizes some objects within the inlier clusters as outliers while ignoring some outlier objects close to inlier clusters. HCOD improperly identifies a segment of the spiral cluster consisting of inlier objects as outliers. OutRank makes the same mistake with HCOD, and ignores the majority of the



(a) (b) (c) (d) (e)

**FIGURE 4. Comparison on synthetic dataset 2. The top 30 objects with largest outlier scores are marked as outliers. (a) APS. (b) MNGApk(LOF). (c) NaNE(LIGRW). (d) HCOD. (e) OutRank.**

true outliers. Compared with these algorithms, the proposed APS model gets a better satisfying result. It basically identified all of the outlier objects.

### C. REAL-WORLD DATASETS

The lack of widely accepted benchmark datasets to compare the performance of the emerging algorithms has long been an open issue in the field of outlier detection. Many of the related studies either use some synthetic datasets, or select class(es) with a small number of samples as outliers in the datasets to evaluate different anomaly detection algorithms. Emmott *et al.* [34] proposed a new method to systematically construct the outlier datasets from the real-world datasets. First, each dataset is converted into a binary problem. Then, each outlier object is assigned with a *difficulty score* according to the probability it belongs to the outlier class estimated by a kernel logistic regression process. Afterwards, each dataset is transformed into several variants by combing all of the inlier objects and the outlier objects under the same *difficulty level*.

In this study, we adopt 57 real-world datasets constructed following the Emmott's method, and the detail preprocessing process refers to [35]. The characteristics of the datasets are described in Table 1.

Normally, the unsupervised outlier detection methods could return a collection of scores indicating the outlier-ness of each object in the dataset. A small subset of the data objects with larger outlier scores are commonly interested from the user's perspective. Let $n$ be the number of outlier objects a user expect to get from the outlier detection algorithm, TP and FP denote the number of true outliers and true inliers in the top-$n$ objects with largest outlier scores, respectively. FN represents the number of outliers whose scores are not ranked in top-$n$. The Precision, Recall and F1 can be calculated as following:

$$\text{Precision} = \frac{\text{TP}}{\text{TP}+\text{FP}} \quad \text{Recall} = \frac{\text{TP}}{\text{TP}+\text{FN}}$$

$$\text{F1} = 2 \cdot \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

In particular, if $n$ is set to the number of true outliers in the dataset, the following relationship holds: $n = \text{TP}+\text{FP}$. Under this circumstance, we could infer that the values of Precision, Recall and F1 will be the same.

**TABLE 1. The characteristics of the selected 57 real-world datasets.**

| id | Datasets | # of records | # of features | # of outliers | ratio of outliers |
|---|---|---|---|---|---|
| 1 | Blood-Transfusion-Easy | 391 | 4 | 7 | 1.79 % |
| 2 | Blood-Transfusion-Medium | 393 | 4 | 9 | 2.29 % |
| 3 | Blood-Transfusion-Hard | 433 | 4 | 49 | 11.32 % |
| 4 | Blood-Transfusion-Very-Hard | 468 | 4 | 84 | 17.95 % |
| 5 | Breast-Cancer-Wisconsin-Easy | 545 | 30 | 188 | 34.50 % |
| 6 | Breast-Cancer-Wisconsin-Medium | 375 | 30 | 18 | 4.80 % |
| 7 | Breast-Cancer-Wisconsin-Hard | 362 | 30 | 5 | 1.38 % |
| 8 | Breast-Tissue-Easy | 83 | 9 | 17 | 20.48 % |
| 9 | Breast-Tissue-Medium | 71 | 9 | 5 | 7.04 % |
| 10 | Breast-Tissue-Hard | 69 | 9 | 3 | 4.35 % |
| 11 | Breast-Tissue-Very-Hard | 81 | 9 | 15 | 18.52 % |
| 12 | Cardiotocography-Easy | 1974 | 27 | 143 | 7.24 % |
| 13 | Cardiotocography-Medium | 1917 | 27 | 86 | 4.49 % |
| 14 | Cardiotocography-Hard | 1879 | 27 | 48 | 2.55 % |
| 15 | Cardiotocography-Very-Hard | 1849 | 27 | 18 | 0.97 % |
| 16 | Ecoli-Easy | 286 | 7 | 81 | 28.32 % |
| 17 | Ecoli-Medium | 232 | 7 | 27 | 11.64 % |
| 18 | Ecoli-Hard | 214 | 7 | 9 | 4.21 % |
| 19 | Ecoli-Very-Hard | 219 | 7 | 14 | 6.39 % |
| 20 | Glass-Easy | 189 | 10 | 75 | 39.68 % |
| 21 | Glass-Medium | 134 | 10 | 20 | 14.93 % |
| 22 | Glass-Hard | 117 | 10 | 3 | 2.56 % |
| 23 | Glass-Very-Hard | 116 | 10 | 2 | 1.72 % |
| 24 | Haberman-Easy | 229 | 3 | 4 | 1.75 % |
| 25 | Haberman-Medium | 236 | 3 | 11 | 4.66 % |
| 26 | Haberman-Hard | 244 | 3 | 19 | 7.79 % |
| 27 | Haberman-Very-Hard | 272 | 3 | 47 | 17.28 % |
| 28 | Ionosphere-Easy | 261 | 33 | 36 | 13.79 % |
| 29 | Ionosphere-Medium | 311 | 33 | 86 | 27.65 % |
| 30 | Ionosphere-Hard | 228 | 33 | 3 | 1.32 % |
| 31 | Iris-Easy | 144 | 4 | 44 | 30.56 % |
| 32 | Iris-Medium | 102 | 4 | 2 | 1.96 % |
| 33 | Iris-Hard | 102 | 4 | 2 | 1.96 % |
| 34 | Iris-Very-Hard | 102 | 4 | 2 | 1.96 % |
| 35 | Libras-Easy | 331 | 90 | 115 | 34.74 % |
| 36 | Libras-Medium | 244 | 90 | 28 | 11.48 % |
| 37 | Multiple-Features-Easy | 1263 | 649 | 63 | 4.99 % |
| 38 | Multiple-Features-Medium | 1937 | 649 | 737 | 38.05 % |
| 39 | Parkinsons-Easy | 178 | 22 | 31 | 17.42 % |
| 40 | Parkinsons-Medium | 160 | 22 | 13 | 8.13 % |
| 41 | Parkinsons-Hard | 151 | 22 | 4 | 2.65 % |
| 42 | Pima-Indians-Easy | 601 | 8 | 101 | 16.81 % |
| 43 | Pima-Indians-Medium | 576 | 8 | 76 | 13.19 % |
| 44 | Pima-Indians-Hard | 545 | 8 | 45 | 8.26 % |
| 45 | Pima-Indians-Very-Hard | 546 | 8 | 46 | 8.42 % |
| 46 | Sonar-Easy | 166 | 60 | 55 | 33.13 % |
| 47 | Sonar-Medium | 153 | 60 | 42 | 27.45 % |
| 48 | Statlog-Vehicle-Easy | 675 | 18 | 46 | 6.81 % |
| 49 | Statlog-Vehicle-Medium | 715 | 18 | 86 | 12.03 % |
| 50 | Statlog-Vehicle-Hard | 694 | 18 | 65 | 9.37 % |
| 51 | Statlog-Vehicle-Very-Hard | 649 | 18 | 20 | 3.08 % |
| 52 | Synthetic-Control-Chart-Easy | 597 | 60 | 197 | 33.00 % |
| 53 | Synthetic-Control-Chart-Medium | 403 | 60 | 3 | 0.74 % |
| 54 | Vertebral-Column-Hard | 478 | 6 | 68 | 14.23 % |
| 55 | Vertebral-Column-Very-Hard | 552 | 6 | 142 | 25.72 % |
| 56 | Wine-Easy | 172 | 13 | 65 | 37.79 % |
| 57 | Wine-Medium | 113 | 13 | 6 | 5.31 % |

Please note that, when the number of true outliers is extremely small, the precision measure becomes less valuable to evaluate the performance of the algorithm. The Receiver

**TABLE 2.** The ROC AUC scores and Precision scores of the APS model against four other algorithms.

| | | AUC Score | | | | | Precision Score | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| id | Datasets | APS | MNGApk(LOF) | NaNE(LIGRW) | OutRank | HCOD | APS | MNGApk(LOF) | NaNE(LIGRW) | OutRank | HCOD |
| 1 | Blood-Transfusion-Easy | **0.980** | 0.945 | 0.623 | 0.859 | 0.056 | **0.286** | **0.286** | 0.143 | 0.000 | 0.000 |
| 2 | Blood-Transfusion-Medium | 0.804 | 0.747 | 0.692 | **0.836** | 0.188 | **0.444** | **0.444** | 0.222 | 0.000 | 0.000 |
| 3 | Blood-Transfusion-Hard | 0.547 | 0.503 | 0.470 | **0.719** | 0.600 | **0.122** | 0.061 | 0.061 | 0.000 | 0.020 |
| 4 | Blood-Transfusion-Very-Hard | 0.452 | 0.482 | 0.438 | 0.469 | **0.523** | 0.167 | 0.107 | **0.179** | 0.131 | 0.000 |
| 5 | Breast-Cancer-Wisconsin-Easy | **0.893** | 0.677 | 0.619 | 0.735 | 0.077 | **0.755** | 0.527 | 0.431 | 0.564 | 0.000 |
| 6 | Breast-Cancer-Wisconsin-Medium | 0.860 | **0.939** | 0.791 | 0.868 | 0.127 | 0.500 | **0.667** | 0.333 | 0.444 | 0.000 |
| 7 | Breast-Cancer-Wisconsin-Hard | 0.791 | **0.923** | 0.540 | 0.897 | 0.087 | **0.000** | **0.000** | **0.000** | **0.000** | **0.000** |
| 8 | Breast-Tissue-Easy | **0.989** | 0.840 | 0.757 | 0.501 | 0.485 | **0.882** | 0.647 | 0.471 | 0.059 | 0.000 |
| 9 | Breast-Tissue-Medium | **0.824** | 0.691 | 0.770 | 0.382 | 0.485 | **0.400** | 0.200 | 0.200 | 0.000 | 0.000 |
| 10 | Breast-Tissue-Hard | 0.227 | 0.374 | 0.417 | 0.333 | **0.485** | **0.000** | **0.000** | **0.000** | **0.000** | **0.000** |
| 11 | Breast-Tissue-Very-Hard | 0.376 | 0.444 | **0.510** | 0.383 | 0.485 | 0.000 | **0.200** | 0.133 | 0.000 | 0.000 |
| 12 | Cardiotocography-Easy | **0.747** | 0.566 | 0.548 | 0.685 | 0.481 | 0.063 | **0.105** | 0.098 | 0.000 | 0.000 |
| 13 | Cardiotocography-Medium | **0.753** | 0.621 | 0.625 | 0.677 | 0.483 | 0.105 | 0.070 | **0.140** | 0.058 | 0.000 |
| 14 | Cardiotocography-Hard | 0.631 | 0.516 | 0.555 | **0.644** | 0.591 | **0.021** | 0.000 | 0.000 | 0.000 | **0.021** |
| 15 | Cardiotocography-Very-Hard | **0.662** | 0.558 | 0.538 | 0.641 | 0.591 | 0.000 | 0.000 | 0.000 | 0.000 | **0.056** |
| 16 | Ecoli-Easy | **0.753** | 0.482 | 0.556 | 0.729 | 0.528 | 0.432 | 0.284 | 0.358 | **0.494** | 0.321 |
| 17 | Ecoli-Medium | **0.773** | 0.644 | 0.695 | 0.731 | 0.455 | 0.259 | 0.222 | **0.296** | 0.222 | 0.185 |
| 18 | Ecoli-Hard | **0.793** | 0.538 | 0.528 | 0.573 | 0.132 | **0.111** | 0.000 | 0.000 | 0.000 | 0.000 |
| 19 | Ecoli-Very-Hard | **0.617** | 0.339 | 0.424 | 0.573 | 0.352 | 0.000 | 0.000 | 0.000 | 0.000 | **0.071** |
| 20 | Glass-Easy | 0.594 | 0.505 | 0.475 | **0.765** | 0.646 | 0.360 | 0.400 | 0.360 | **0.547** | 0.520 |
| 21 | Glass-Medium | **0.961** | 0.731 | 0.528 | 0.589 | 0.291 | **0.650** | 0.400 | 0.300 | 0.300 | 0.100 |
| 22 | Glass-Hard | 0.959 | **0.988** | 0.652 | 0.322 | 0.763 | **0.667** | **0.667** | 0.333 | 0.000 | 0.000 |
| 23 | Glass-Very-Hard | 0.991 | **1.000** | 0.746 | 0.294 | 0.833 | 0.500 | **1.000** | 0.000 | 0.000 | 0.500 |
| 24 | Haberman-Easy | **0.953** | 0.843 | 0.780 | 0.863 | 0.713 | 0.000 | 0.000 | 0.000 | 0.000 | **0.250** |
| 25 | Haberman-Medium | 0.873 | **0.884** | 0.594 | 0.780 | 0.538 | 0.364 | 0.182 | 0.273 | **0.455** | 0.000 |
| 26 | Haberman-Hard | **0.883** | 0.844 | 0.631 | 0.645 | 0.783 | **0.368** | 0.158 | 0.105 | 0.263 | 0.053 |
| 27 | Haberman-Very-Hard | 0.458 | 0.490 | **0.505** | 0.458 | 0.439 | 0.106 | **0.191** | 0.170 | 0.149 | 0.043 |
| 28 | Ionosphere-Easy | 0.716 | 0.850 | 0.719 | **0.867** | 0.605 | 0.250 | 0.389 | **0.500** | 0.444 | 0.250 |
| 29 | Ionosphere-Medium | 0.852 | 0.771 | **0.919** | 0.897 | 0.475 | 0.593 | 0.628 | 0.767 | **0.802** | 0.267 |
| 30 | Ionosphere-Hard | 0.739 | 0.573 | **0.757** | 0.545 | 0.049 | **0.000** | **0.000** | **0.000** | **0.000** | **0.000** |
| 31 | Iris-Easy | 0.320 | 0.437 | 0.462 | 0.001 | **0.543** | 0.182 | 0.227 | **0.295** | 0.000 | **0.295** |
| 32 | Iris-Medium | 0.735 | 0.815 | **0.932** | 0.120 | 0.555 | 0.000 | 0.000 | **0.500** | 0.000 | 0.000 |
| 33 | Iris-Hard | 0.575 | **0.650** | 0.625 | 0.095 | 0.450 | **0.000** | **0.000** | **0.000** | **0.000** | **0.000** |
| 34 | Iris-Very-Hard | 0.345 | 0.425 | 0.670 | 0.180 | **0.800** | **0.000** | **0.000** | **0.000** | **0.000** | **0.000** |
| 35 | Libras-Easy | 0.513 | 0.443 | 0.440 | **0.517** | 0.481 | 0.365 | 0.287 | 0.313 | **0.400** | 0.348 |
| 36 | Libras-Medium | 0.454 | 0.516 | **0.689** | 0.420 | 0.484 | 0.071 | **0.143** | **0.143** | 0.036 | 0.000 |
| 37 | Multiple-Features-Easy | **0.761** | 0.535 | 0.359 | 0.068 | 0.516 | 0.095 | 0.143 | 0.000 | 0.032 | **1.000** |
| 38 | Multiple-Features-Medium | 0.494 | 0.471 | **0.500** | 0.188 | 0.499 | 0.292 | 0.372 | **0.395** | 0.096 | 0.000 |
| 39 | Parkinsons-Easy | **0.739** | 0.686 | 0.492 | 0.727 | 0.459 | **0.290** | 0.161 | 0.161 | 0.226 | 0.000 |
| 40 | Parkinsons-Medium | 0.460 | **0.582** | 0.543 | 0.549 | 0.386 | 0.000 | **0.077** | **0.077** | **0.077** | **0.077** |
| 41 | Parkinsons-Hard | 0.410 | 0.509 | **0.731** | 0.497 | 0.486 | 0.000 | 0.000 | 0.000 | 0.000 | **0.250** |
| 42 | Pima-Indians-Easy | **0.800** | 0.770 | 0.597 | 0.635 | 0.187 | **0.416** | 0.307 | 0.257 | 0.297 | 0.000 |
| 43 | Pima-Indians-Medium | 0.689 | **0.727** | 0.557 | 0.633 | 0.277 | **0.276** | 0.250 | 0.211 | 0.197 | 0.000 |
| 44 | Pima-Indians-Hard | 0.530 | 0.449 | 0.395 | **0.538** | 0.389 | 0.000 | **0.044** | 0.022 | 0.022 | 0.022 |
| 45 | Pima-Indians-Very-Hard | 0.460 | 0.439 | 0.507 | **0.518** | 0.420 | 0.043 | 0.022 | **0.087** | 0.043 | 0.000 |
| 46 | Sonar-Easy | 0.483 | 0.499 | 0.413 | **0.518** | 0.425 | 0.255 | 0.273 | 0.291 | **0.309** | 0.255 |
| 47 | Sonar-Medium | 0.671 | 0.672 | **0.778** | 0.652 | 0.346 | 0.405 | 0.381 | **0.548** | 0.310 | 0.119 |
| 48 | Statlog-Vehicle-Easy | **0.735** | 0.350 | 0.473 | 0.659 | 0.299 | 0.000 | 0.000 | **0.065** | 0.000 | 0.000 |
| 49 | Statlog-Vehicle-Medium | **0.629** | 0.483 | 0.516 | 0.520 | 0.289 | 0.070 | 0.116 | **0.128** | 0.047 | 0.000 |
| 50 | Statlog-Vehicle-Hard | **0.565** | 0.486 | 0.525 | 0.480 | 0.359 | 0.015 | 0.077 | **0.092** | 0.015 | 0.000 |
| 51 | Statlog-Vehicle-Very-Hard | **0.534** | 0.434 | 0.407 | 0.483 | 0.442 | **0.000** | **0.000** | **0.000** | **0.000** | **0.000** |
| 52 | Synthetic-Control-Chart-Easy | **0.867** | 0.626 | 0.595 | 0.761 | 0.220 | **0.680** | 0.518 | 0.416 | 0.619 | 0.051 |
| 53 | Synthetic-Control-Chart-Medium | 0.886 | **0.921** | 0.795 | 0.588 | 0.380 | 0.000 | **0.333** | 0.000 | 0.000 | 0.000 |
| 54 | Vertebral-Column-Hard | **0.747** | 0.553 | 0.336 | 0.627 | 0.176 | 0.206 | 0.191 | 0.000 | **0.279** | 0.000 |
| 55 | Vertebral-Column-Very-Hard | **0.570** | 0.538 | 0.408 | 0.513 | 0.366 | **0.310** | 0.303 | 0.007 | 0.289 | 0.000 |
| 56 | Wine-Easy | 0.311 | 0.414 | 0.414 | **0.521** | 0.302 | 0.169 | 0.277 | 0.323 | **0.400** | 0.015 |
| 57 | Wine-Medium | 0.609 | 0.614 | **0.745** | 0.642 | 0.369 | 0.000 | 0.167 | 0.333 | **0.500** | 0.000 |
| | Average | **0.673** | 0.620 | 0.584 | 0.560 | 0.425 | **0.220** | 0.219 | 0.185 | 0.160 | 0.089 |

Operating Characteristics (ROC) curves capture the trade-off between true positive rate and false positive rate by using different threshold. The Area Under Curve (AUC) summarize the ROC curve into a single value by calculating the area under the ROC curve, which ranges from 0 to 1. An algorithm with a larger ROC AUC score indicates that it has a higher probability to rank the true outliers before the inlier objects, hence is more preferable.

To better demonstrate the performance of the proposed model, in this study, both of the Precision and ROC AUC score measures are utilized to evaluate the detection result. Particularly, the value of $n$ is set to the number of true outliers when calculating the Precision.

For the proposed APS model, the Euclidean distance is adopted to generate the proximity graph on each of the dataset. For the number of the proximity graphs, we use 50 for all datasets. The rest of the algorithms use their default settings. The ROC AUC scores and Precision scores on all of the 57 datasets are shown in Table 2.

Taking the dataset *Glass-Easy* for example, we select the top 75 objects with largest outlier scores as the outlier objects, then the corresponding Precision score (equals to the Recall

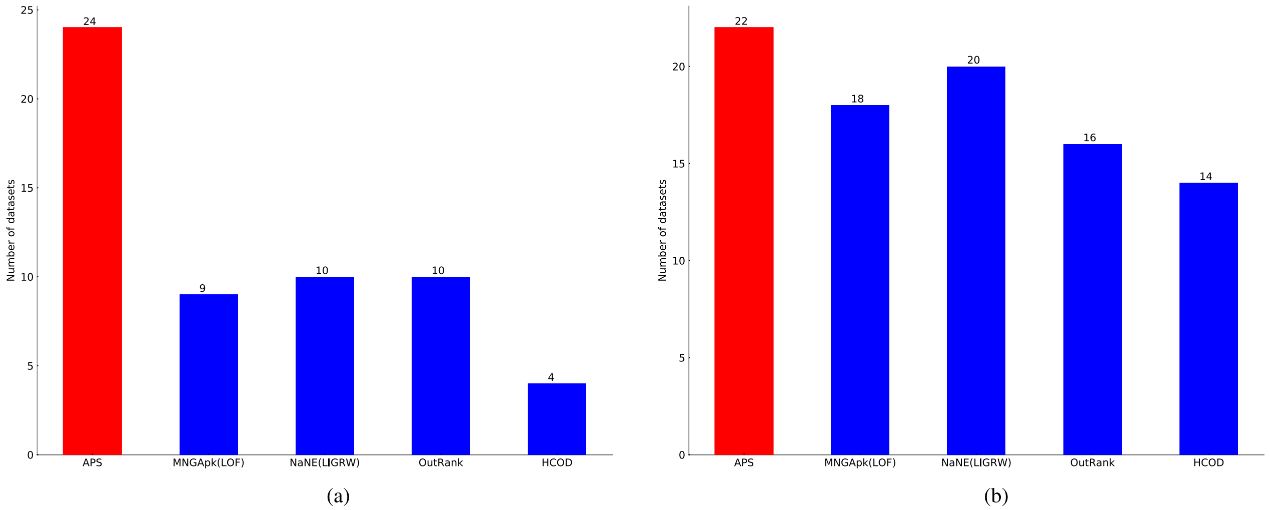


**FIGURE 5.** Comparison of the number of winners for the algorithms in outlier detection on all of the real-world datasets. The values in each bar indicate the number of datasets when the corresponding algorithm outperforms the others. (a) Comparison by ROC AUC. (b) Comparison by Precision.

and F1 measure) and ROC AUC score are computed, separately. From Table 2, we can see that, generally, the performance of each algorithm decreases on dataset with a higher *difficulty level*. whether ROC AUC or Precision is used, there is no algorithm can outperform the others on all of the 57 datasets. However, the proposed APS model has the largest average ROC AUC score and Precision score on all datasets, which means that the it has better average performance than the compared algorithms. In addition, on the measure of ROC AUC, the proposed APS model achieves the best on 24 (42.11%) datasets. And on the measure of Precision, it outperforms the compared algorithms on 20 (38.6%) datasets, which is demonstrated in Fig. 5.

### D. THE EFFECT OF THE PROXIMITY MEASURE

Many graph-based outlier detection algorithms rely on a specific proximity measure to construct the neighborhood graph. Taken OutRank for example, it either uses cosine similarity or uses the RBF kernel to calculate the similarities between the objects. Once the proximity measure changes, the performance of the algorithm could not be guaranteed.

On the contrary, the proposed APS model does not directly use the visiting probability of the random walker as the outlier score. Instead, the visiting probabilities obtained on different graphs are combined together to calculate an anomaly pattern score, which indicates the outlier-ness of the related object. Therefore, the proposed APS model can accommodate different proximity measures, which can be freely chosen according to the specific features of the datasets. We provide an example to illustrate the above idea. It is recommended by Kriegel *et al.* [33] that using the variances of angles between the feature vectors in high dimensional datasets could achieve a better results than the distance criteria. We equip the proposed APS model with different proximity measures, and apply it on several real-world datasets to demonstrate its adaptiveness.

The original KDDCup99 dataset[1] contains different types of network attacks, which has 60,839 inlier objects, and 246 objects belong to the following attacks: buffer_overflow, ftp_write, imap, load_module, multihop, nmap, perl, phf, pod, rootkit, and teardrop are taken as outliers. After using One Hot Encoding to map the categorical features into numerical ones, there are 79 features for each of the data object. We randomly selected 1000 inlier objects, and some random outlier objects to construct five different datasets. The characteristics of the datasets are shown in Table. 3.

**TABLE 3.** The characteristics of the KDDCup99 datasets.

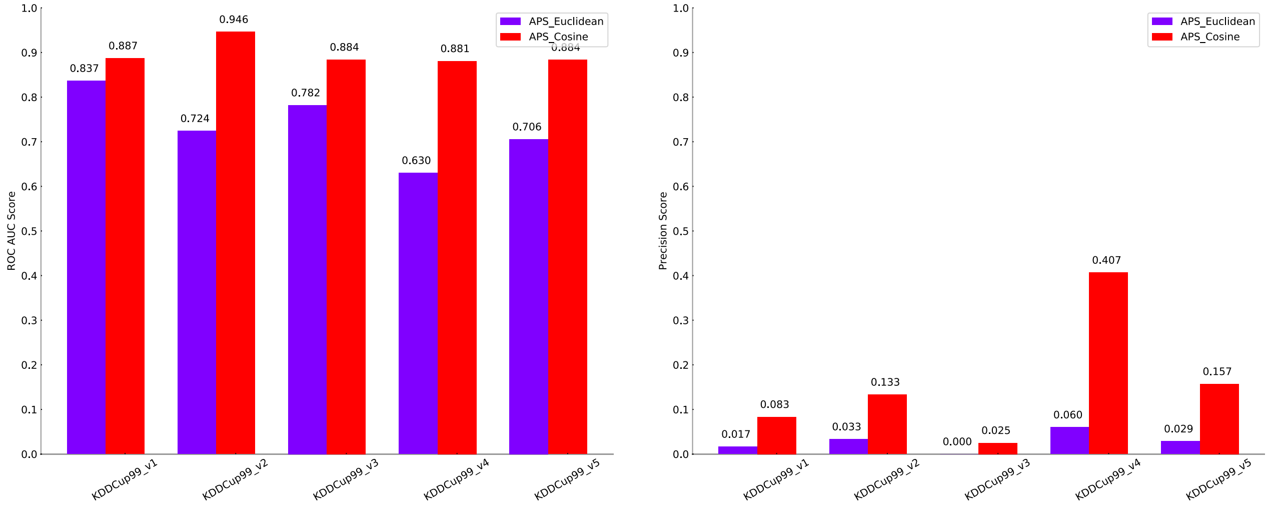
| id | Datasets | # of records | # of features | # of outliers | ratio of outliers |
|---|---|---|---|---|---|
| 1 | KDDCup99_v1 | 1060 | 79 | 60 | 5.66 % |
| 2 | KDDCup99_v2 | 1030 | 79 | 30 | 2.91 % |
| 3 | KDDCup99_v3 | 1040 | 79 | 40 | 3.85 % |
| 4 | KDDCup99_v4 | 1150 | 79 | 150 | 13.04 % |
| 5 | KDDCup99_v5 | 1070 | 79 | 70 | 6.54 % |

We constructed the local proximity graphs using Euclidean distance and Cosine similarity, separately. Then we apply the proposed APS model. The results are shown in Fig. 6. From the results, we can see that on the high-dimensional KDDCup dataset, the performance of the APS model constructed by using cosine similarity outperforms its counterparts using the Euclidean distance by both of the ROC AUC measure and the Precision measure. This suggests that the proposed APS model has the flexibility to use different proximity measures when facing datasets with different characteristics.

### E. THE EFFECT OF GRAPH COUNT

Another important parameter in the proposed APS model is the number of proximity graphs it uses to capture the change

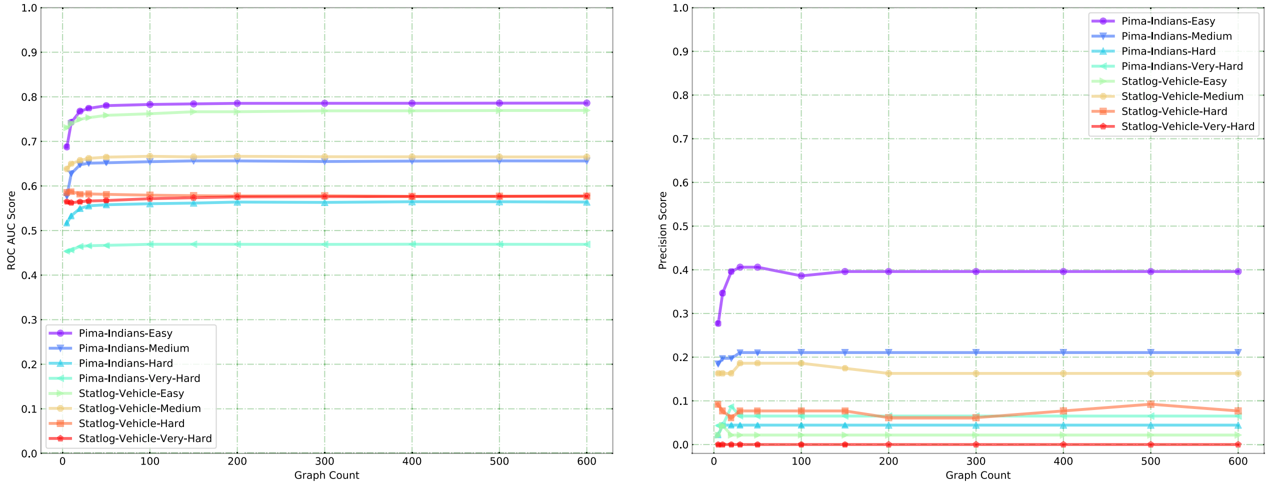
[1] http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html



(a)

(b)

**FIGURE 6.** APS model adopting different metrics (Euclidean distance vs. Cosine similarity). (a) ROC AUC Score. (b) Precision Score.



(a)

(b)

**FIGURE 7.** Comparison of the APS model using various number of proximity graph on several real-world datasets. (a) ROC AUC Score. (b) Precision Score.

pattern for each of the object. In this subsection, we conduct experiments to analyze the effect of this parameter. Without lose of generality, we randomly selected eight real-world datasets with different *difficulty levels*. A sequence of values [5, 10, 20, 30, 50, 100, 150, 200, 300, 400, 500, 600] are utilized to represent the number of proximity graphs used in the APS model, respectively. The results are demonstrated in Fig. 7.

From the results, we can see that as the number of graphs increases, the performance of the algorithm begins to increase. When this value reaches up to 50 or so, the performance tends to be stable. Moreover, the trend nearly holds for all tested datasets. This means that our proposed APS model is not sensitive to the number of graphs.

## VI. CONCLUSION

After analyze the particular characteristics of the random walk based graph models, in this study, we proposed a new outlier detection model named Anomaly Pattern Score. Unlike the former methods, it does not depend on a specific proximity measure, which ensures its adaptiveness on different application scenarios. Moreover, the proposed APS model does not rely on a user-specified parameter of neighbor size, which is achieved by applying multiple random walk processes on various local proximity graphs. Besides, it is not sensitive to the number of graphs adopted. Extensive experiments were conducted on synthetic and real-world datasets, and the results suggested that the proposed APS model outperforms the state-of-the-art algorithms by the measures of average ROC AUC score and Precision score.

## REFERENCES

[1] A. Zimek and E. Schubert, *Outlier Detection*. New York, NY, USA: Springer, 2017, pp. 1–5, doi: 10.1007/978-1-4899-7993-3_80719-1.

[2] S. Rayana and L. Akoglu, "Collective opinion spam detection: Bridging review networks and metadata," in *Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*. New York, NY, USA: ACM, 2015, pp. 985–994, doi: 10.1145/2783258.2783370.

[3] H. Li, Z. Chen, A. Mukherjee, B. Liu, and J. Shao, "Analyzing and detecting opinion spam on a large-scale dataset via temporal and spatial patterns," in *Proc. ICWSM*, 2015, pp. 634–637.

[4] S. Rayana and L. Akoglu, *Collective Opinion Spam Detection using Active Inference*. Philadelphia, PA, USA: SIAM, 2016, pp. 630–638, doi: 10.1137/1.9781611974348.71.

[5] M. J. V. Leach, E. P. Sparks, and N. M. Robertson, "Contextual anomaly detection in crowded surveillance scenes," *Pattern Recognit. Lett.*, vol. 44, pp. 71–79, Jul. 2014.

[6] T. Xiao, C. Zhang, and H. Zha, "Learning to detect anomalies in surveillance video," *IEEE Signal Process. Lett.*, vol. 22, no. 9, pp. 1477–1481, Sep. 2015.

[7] C. Bowles *et al.*, "Brain lesion segmentation through image synthesis and outlier detection," *NeuroImage, Clin.*, vols. 13–16, pp. 643–658, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2213158217302164

[8] X. Chen and E. Konukoglu. (2018). "Unsupervised detection of lesions in brain MRI using constrained adversarial auto-encoders." [Online]. Available: https://arxiv.org/abs/1806.04972

[9] V. Barnett and T. Lewis, *Outliers In Statistical Data*. New York, NY, USA: Wiley, 1994.

[10] T. Johnson, I. Kwok, and R. T. Ng, "Fast computation of 2-dimensional depth contours," in *Proc. KDD*, 1998, pp. 224–228.

[11] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. KDD*, 1996, vol. 96, no. 34, pp. 226–231.

[12] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: An efficient data clustering method for very large databases," *ACM SIGMOD Rec.*, vol. 25, pp. 103–114, Jun. 1996.

[13] S. Guha, R. Rastogi, and K. Shim, "Cure: An efficient clustering algorithm for large databases," in *ACM SIGMOD Rec.*, vol. 27, no. 2, pp. 73–84, 1998.

[14] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, no. 6191, pp. 1492–1496, Jun. 2014.

[15] E. M. Knorr and R. T. Ng, "Finding intensional knowledge of distance-based outliers," in *Proc. VLDB*, vol. 99, 1999, pp. 211–222.

[16] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: Identifying density-based local outliers," *SIGMOD Rec.*, vol. 29, no. 2, pp. 93–104, 2000, doi: 10.1145/335191.335388.

[17] J. Tang, Z. Chen, A. W.-C. Fu, and D. W. Cheung, "Enhancing effectiveness of outlier detections for low density patterns," in *Proc. Pacific-Asia Conf. Knowl. Discovery Data Mining*. Taipei, Taiwan: Springer, 2002, pp. 535–548.

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

[19] B. Tang and H. He, "A local density-based approach for outlier detection," *Neurocomputing*, vol. 241, pp. 171–180, Jun. 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0925231217303302

[20] L. Akoglu, H. Tong, and D. Koutra, "Graph based anomaly detection and description: A survey," *Data Mining Knowl. Discovery*, vol. 29, no. 3, pp. 626–688, 2015.

[21] S. Ranshous, S. Shen, D. Koutra, S. Harenberg, C. Faloutsos, and N. F. Samatova, "Anomaly detection in dynamic networks: A survey," *Wiley Interdiscipl. Rev., Comput. Stat.*, vol. 7, no. 3, pp. 223–247, 2015.

[22] V. Hautamaki, I. Karkkainen, and P. Franti, "Outlier detection using k-nearest neighbour graph," in *Proc. 17th Int. Conf. Pattern Recognit. (ICPR)*, vol. 3, Aug. 2004, pp. 430–433.

[23] H. D. K. Moonesignhe and P. N. Tan, "Outlier detection using random walks," in *Proc. 18th IEEE Int. Conf. Tools Artif. Intell. (ICTAI)*, Nov. 2006, pp. 532–539.

[24] H. D. K. Moonesinghe and P.-N. Tan, "Outrank: A graph-based outlier detection framework using random walk," *Int. J. Artif. Intell. Tools*, vol. 17, no. 1, pp. 19–36, 2008, doi: 10.1142/S0218213008003753.

[25] X. Wang and I. Davidson, "Discovering contexts and contextual outliers using random walks in graphs," in *Proc. 9th IEEE Int. Conf. Data Mining (ICDM)*, Dec. 2009, pp. 1034–1039.

[26] C. Wang, H. Gao, Z. Liu, and Y. Fu, "A new outlier detection model using random walk on local information graph," *IEEE Access*, vol. 6, pp. 75531–75544, 2018.

[27] J. Ha, S. Seok, and J.-S. Lee, "Robust outlier detection using the instability factor," *Knowl.-Based Syst.*, vol. 63, pp. 15–23, Jun. 2014. [Online]. Available:r http://www.sciencedirect.com/science/article/pii/S0950705114000744

[28] G. Bhattacharya, K. Ghosh, and A. S. Chowdhury, "Outlier detection using neighborhood rank difference," *Pattern Recognit. Lett.*, vols. 60–61, pp. 24–31, Aug. 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167865515001130

[29] Q. Zhu, J. Feng, and J. Huang, "Weighted natural neighborhood graph: An adaptive structure for clustering and outlier detection with no neighborhood parameter," *Cluster Comput.*, vol. 19, no. 3, pp. 1385–1397, Sep. 2016, doi: 10.1007/s10586-016-0598-1.

[30] Q. Zhu, J. Feng, and J. Huang, "Natural neighbor: A self-adaptive neighborhood method without parameter K," *Pattern Recognit. Lett.*, vol. 80, pp. 30–36, Sep. 2016.

[31] J. Ning, L. Chen, C. Zhou, and Y. Wen, "Parameter k search strategy in outlier detection," *Pattern Recognit. Lett.*, vol. 112, pp. 56–62, Sep. 2018.

[32] J. Huang, Q. Zhu, L. Yang, and J. Feng, "A non-parameter outlier detection algorithm based on natural neighbor," *Knowl.-Based Syst.*, vol. 92, pp. 71–77, Jan. 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0950705115004013

[33] H.-P. Kriegel, M. Schubert, and A. Zimek, "Angle-based outlier detection in high-dimensional data," in *Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*. New York, NY, USA: ACM, 2008, pp. 444–452, doi: 10.1145/1401890.1401946.

[34] A. F. Emmott, S. Das, T. Dietterich, A. Fern, and W.-K. Wong, "Systematic construction of anomaly detection benchmarks from real data," in *Proc. ACM SIGKDD Workshop Outlier Detection Description*. New York, NY, USA: ACM, 2013, pp. 16–21.

[35] T. Pevný, "Loda: Lightweight on-line detector of anomalies," *Mach. Learn.*, vol. 102, no. 2, pp. 275–304, Feb. 2016, doi: 10.1007/s10994-015-5521-0.

**CHAO WANG** received the B.Sc. degree in computer science from Southwest Petroleum University, China, in 2007. He is currently pursuing the Ph.D. degree with the School of Computer Science and Engineering, University of Electronic Science and Technology of China. His research interests include data mining, outlier detection, and machine learning.

**ZHEN LIU** received the Ph.D. degree in technology of computer application from the University of Electronic Science and Technology of China (UESTC), in 2007. He was a Visiting Scholar with the Data Mining Lab, Minnesota University, from 2012 to 2013. He has been an Associate Professor with the School of Computer Science and Engineering, UESTC, since 2011. He has published more than 30 peer-reviewed papers in his academic career. His current research interests include data mining and analysis, machine learning, and social network analysis.

**HUI GAO** received the Ph.D. degree in computing science from the University of Groningen, The Netherlands, in 2005. In 2006, he joined the University of Electronic Science and Technology of China, where he is currently a Professor in the field of data mining with the School of Computer Science and Engineering. His main research interests include distributed data mining, privacy preserving, and artificial intelligence.

**YAN FU** received the master's degree in computing science from the University of Electronic Science and Technology of China, in 1988, where she is currently a Professor in the field of data mining with the School of Computer Science and Engineering. Her main research interests include data mining, complex networks, and artificial intelligence.