Hindawi
Mobile Information Systems
Volume 2020, Article ID 5068731, 12 pages
https://doi.org/10.1155/2020/5068731

Research Article

An Area-Context-Based Credibility Detection for Big Data in IoT

Bo Zhao, Xiang Li, Jiayue Li, Jianwen Zou, and Yifan Liu

Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University, Wuhan 430072, China

Correspondence should be addressed to Xiang Li; lixiang950629@whu.edu.cn

Received 29 September 2019; Accepted 6 November 2019; Published 25 January 2020

Academic Editor: Marco Picone

In order to improve the credibility of big data analysis platform's results in IoT, it is necessary to improve the quality of IoT data. Many detection methods have been proposed to filter out incredible data, but there are certain deficiencies that performance is not high, detection is not comprehensive, and process is not credible. So this paper proposes an area-context-based credibility detection method for IoT data, which can effectively detect point anomalies, behavioral anomalies, and contextual anomalies. The performance of the context determination and the data credibility detection of the device satisfying the area characteristics is superior to the similar algorithms. As the experiments show, the proposed method can reach a high level of performance with more than 97% in metrics, which can effectively improve the quality of IoT data.

1. Introduction

With the rapid development of information technology, IoT (Internet of Things) products have been widely used in various aspects of agriculture, industry, and social life, bringing a huge market to the world [1, 2]. Combining IoT technology with the artificial intelligence and big data analysis, the new service model, big data analysis platform in IoT, has been created in recent years. Using the IoT as data source, the big data as the object of analysis, and the artificial intelligence as the technology [3, 4], the big data analysis platform's credibility depends on the data collected by the IoT devices. However, existing IoT devices are prone to malfunction and security issues, resulting in erroneous data [5, 6] and incorrect analysis results.

Due to the analysis of incredible data by the big data analysis platform in IoT, there have been many security incidents all around the world leading to serious consequences recently. In June 2018, Alibaba Cloud Computing launched an automatic operation and maintenance function but triggered a bug that generated erroneous data when identifying the internal normal IP, which caused denial of service when users accessing to the Alibaba Cloud Computing. In December 2018, an accident occurred in the Amazon Warehouse in New Jersey, USA. Due to errors in the robot detection data, it was mistaken that a beast broke into the warehouse and pierced a can of bear-spraying spray in the warehouse, causing 24 Amazon employees to be taken to the hospital for treatment. How to improve the credibility of IoT data, detect and filter out incredible data, and enable the big data analysis platform to process credible data and produce trusted analysis results has become a key issue in IoT research [7, 8].

Related research has been a focus of many researchers in diverse areas. In terms of IoT device protection, Trusted Computing Technology [9, 10] adds a TPM (trusted platform module) to hardware architecture of the device, transfering trust from TPM to the application layer, to protect the integrity of device and predictability of behavior; TNC (Trusted Network Connection) [11] builds a trusted network transmission process by extending trust from the device itself to the network, which can effectively prevent more and more complex network attacks [12]. However, trusted computing needs to change the hardware architecture of the device, which will bring excessive cost in the deployment of IoT devices. In terms of IoT anomaly detection, device vulnerability detection [13], Web attack detection [14], configuration file detection [15], chip radiation detection [16], lateral movement detection of the network environment [17], botnet detection [18], dynamic

attack detection for Internet of Vehicles [19], routing abnormal behavior analysis [20], and other methods have achieved good results, but these methods are all aimed at IoT devices or networks and do not take data into account which is the object of the big data analysis platform. To fundamentally solve this problem, it is necessary to detect the credibility of IoT data itself, filter out the untrusted data, and ensure the credibility of analysis results.

In view of the abovementioned problems, this paper proposes a credibility detection method for IoT data based on area-context. Our method introduces the concept of area-shared-context and uses the characteristics of devices in the same area sharing the same context in the IoT environment to determine the devices' working state. For cases that do not meet this characteristic, we use the same detection of single device as the existing method. Then, the probabilistic detector based on the sliding window is used to detect the credibility of IoT data itself. The feasibility of the above scheme is verified by experiments, and the experimental results show that the proposed method has good performance.

2. Related Works

In order to improve the credibility of IoT data and ensure the security of the analysis process, it is necessary to make credibility detection on IoT data, filter out untrusted data, and retain high availability data. Researchers around the world have proposed many solutions to this problem.

The method of credibility detection of the IoT data is mainly to judge whether the data itself is abnormal. The anomalies discussed in this paper can be divided into three types: (1) Point anomalies: the data at one time is significantly different from the value of most other data. For example, the normal value of a thermometer ranges from 35°C to 42°C, and if a data of 45°C appears, it is called a point anomaly. (2) Behavioral anomalies, or collective anomalies: the way data changes in a period of time is different from the way credible data changes. For example, the gear speed of a factory equipment ranges in 0–180 r/min and changes slowly and continuously, and if a 10 r/min mutation to 160 r/min occurs at a certain time, the changing pattern is abnormal, which is called behavioral anomaly. (3) Contextual anomalies: IoT devices have different working states in different contexts, and if the data do not correspond to the context in which the device should be currently, we call it a contextual anomaly. For example, the car is accelerating, and the data displayed by the speedometer is decreasing. Many of the existing anomaly detection methods focus on the above three types of anomalies.

References [21] and [22] use Gaussian detectors for anomaly detection. They construct a Gaussian distribution by using the mean and variance of training data and then determine whether the new sample is abnormal according to the threshold. However, this method which determines outliers as anomalies can only detect point anomalies. Maxion and Tan [23] and Cuzzocrea et al. [24] use the Markov detector to detect the security state of the embedded system. This method constructs a Markov matrix based on transition probability of the state sequence. The probability of the current state is calculated according to the previous state value and the probability in the Markov matrix. But they only target point anomalies. Yan et al. [25] propose a hybrid method of using hidden Markov and STIDE detectors for intrusion detection. It mimics the way the immune system works by matching the test data by constructing a template of the normal data, storing all occurrences of the state sequence in the predefined length n into a buffer, and then matching the test data. However, this method consumes a lot of computational and storage resources. The above solutions can detect point anomalies, but they are difficult to detect behavioral abnormalities. Therefore, they are not suitable for the IoT environment.

The detection algorithm using the sliding window mechanism [26] can detect behavioral anomalies very well. Zandrahimi et al. [27] propose an anomaly detector based on sliding window and buffer, which stores the normal data information extracted during the training phase in the buffer. If the data stream of the test data does not exist in the buffer, a miss occurs. If the hit rate does not reach the predefined threshold, then there is an anomaly in the test data. This method is similar to STIDE and also requires large computational and storage resources. Summerville et al. [28] develop an ultralightweight deep packet anomaly detection method that uses a mask and bit pattern matching data in the sliding window for network deep packets to effectively distinguish between normal and abnormal payloads. It only requires a bitwise "AND" operation, so it is suitable for running on resource-constrained IoT devices. Zandrahimi et al. [27] also propose a probabilistic detector method, which samples two pieces of data with different distances in the sliding window, counts the probability of the corresponding state, and detects the anomaly by calculating the overall probability in the sliding window. It has the characteristics of small calculation and high accuracy. There are also many algorithms based on the sliding window [29–31] that are used in a variety of areas for anomaly detection and credibility detection. The sliding window can reflect the behavior of the measured data well but without considering the context. When facing the diversity of IoT devices, we need to consider the context of data to meet the requirements of credibility in the IoT environment.

In order to detect contextual anomalies in data or device, researchers have already begun some work. Hayes and Capretz [32] propose a postprocessing context-aware anomaly detection algorithm based on a sensor profile, which uses a well-defined context anomaly detection algorithm to perform on large sensor data, then uses k-means clustering algorithm to divide different contexts, and finally uses the Gaussian detector to perform anomaly detection on the data that have determined the context. However, the core part of the method uses a Gaussian detector with low accuracy and cannot detect behavioral anomalies. In Reference [33], a context-aware anomaly detection method based on the probabilistic detector is proposed. The scheme takes the probabilistic detector algorithm with good performance as the core, introduces the KNN algorithm to determine the context of data, and loads the corresponding probabilistic

matrix for credibility detection. The detection rate of this method is high, but it does not consider the situation that multiple devices can affect each other when they are related in an area, and the KNN algorithm increases the cost of calculation.

According to the above description and analysis, existing solutions have certain deficiencies when applied to IoT: (1) traditional methods can only detect point anomalies; (2) simple sliding window mechanisms cannot cope with context; (3) existing context-aware detection method process is not credible, and the performance needs to be improved. In order to improve the credibility of IoT data and ensure the correct and effective results from the analysis platform, a credible decision method that can comprehensively detect point anomalies, behavioral anomalies, and contextual anomalies should be proposed for the IoT environment, and the detection process must be trusted.

3. Methods

In view of the demand for data credibility of the big data analysis platform in IoT and the shortcomings of existing methods, this paper proposes a credibility detection method for IoT data based on area-context, as shown in Figure 1. Our method can effectively detect point anomalies, behavioral anomalies, and contextual anomalies and determine the context of data based on the area, which solves the untrustworthy problem of the contextual anomaly detection process.

3.1. Threat Model. The data credibility detection in the IoT environment is quite different from the traditional embedded device anomaly detection. The IoT system studied in this paper has multiple types, quantities, and context states, so the following hypotheses are proposed to construct the threat model of this paper:

Hypothesis 1. IoT data may be credible before it arrives at the analysis platform. That is, the reason of incredibility may be that the IoT device is attacked or fails or it may be wrong during transmission.

There are many reasons why the IoT data is not credible. This paper starts from the perspective of data, only determines whether the data itself is credible, and filters out incredible data without locating the anomaly.

Hypothesis 2. If an attacker controls an IoT device, it can only obtain data from the device that has been attacked but cannot obtain any information about other devices.

If the attacker forges false data, the incredible data will be detected by the credibility detection center. If the attacker hides itself, its behavior will show a difference when compared with other devices in the same area, so it will be detected by the context determination module.

Hypothesis 3. The number of incredible data is a minority of all data analyzed at the same time.

There are lots of devices in IoT. If most of the data is incredible, data analysis will lose its meaning, and it is very difficult to cause such a large number of abnormalities. Therefore, the abnormal data in this paper always accounts for a small part of the total data.

In summary, the constructed threat model conforms to the characteristics of the IoT environment. Our method can solve the security problem in this threat model by taking the data itself as the analysis object, the model trained by credible data as the static analysis part, and the area-shared-context determination of different devices in the actual runtime as the dynamic analysis part.

3.2. Sliding Window and Improved Probabilistic Detector

3.2.1. Principle of Sliding Window. Sliding windows have proven to be a good method for detecting behavioral anomalies in embedded devices and IoT devices [26–28]. A sliding window of fixed length is set on a time series. It slides with the direction of newly generated data over time, and attention is only paid to the sequence in the window at a time. For a piece of data, we represent it as the sequence $\text{Data} = \{d_1, d_2, d_3, \ldots\}$, where d_i is one data in the sequence, and the larger i is, the newer the data is. We set a sliding window SW of length w on Data, each time $\text{SW}_i = \{d_i, d_{i+1}, d_{i+2}, \ldots, d_{i+w-1}\}$ represents a sequence of length w starting from d_i. A probability threshold p and an incredible sequence size threshold k are defined for the sequence in the sliding window, where $p \in (0, 1)$ and k is a positive integer.

Based on the above definition, we believe it is incredible when a sequence of data meets the following conditions:

$$\begin{cases} \text{Data} = \{d_1, d_2, d_3, \ldots, d_n\}; \\ \text{SW}_i = \{d_i, d_{i+1}, d_{i+2}, \ldots, d_{i+w-1}\}; \\ P\left(\text{SW}_{i+j}\right) < p, \quad j = 0, 1, 2, \ldots, t; \\ t + 1 \geq k. \end{cases} \tag{1}$$

For the sequence that satisfies the above formula, we think that there is a $t + 1$ length anomaly event starting from d_i in Data; that is, this piece of data is incredible and needs to be filtered out, as shown in Figure 2.

When $w = 3, k = 3$, and $\text{SW}_{i+j} < p$, $j = 0, 1, 2, 3, 4, 5$, it is considered that an abnormal event of length 6 has occurred from d_i in this data sequence.

3.2.2. Probabilistic Matrix Construction. The probabilistic detector [27, 33] is a good anomaly detection algorithm, which can effectively detect point anomalies and behavioral anomalies. Before training, the original data is represented by size and block and converted into a state sequence $\text{TrainingData} = \{d_1, d_2, d_3, \ldots, d_n\}$, where d_i is no longer one data but the symbol of the block in which the data is located. For example, $\text{Data} = \{5, 11, 9, 22, 27, 18\}$ can be divided into three blocks and expressed as $\text{TrainingData} = \{A, B, A, C, C, B\}$. A sliding window SW is

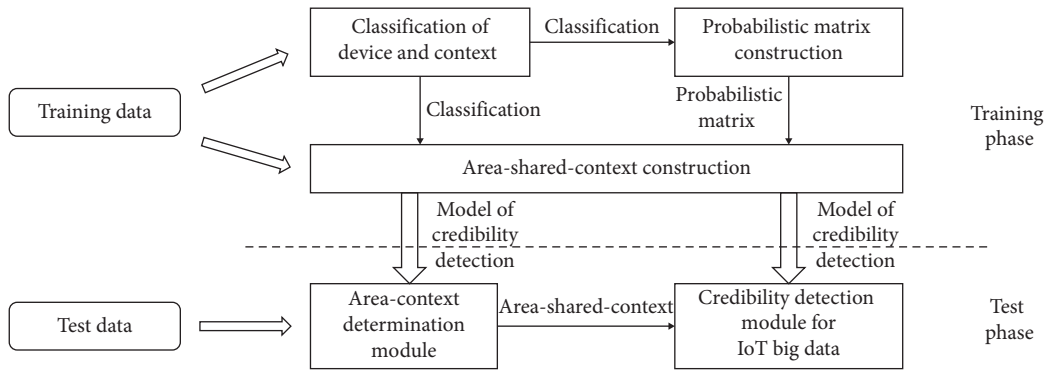


Figure 1: Credibility detection method for IoT data based on area-context.

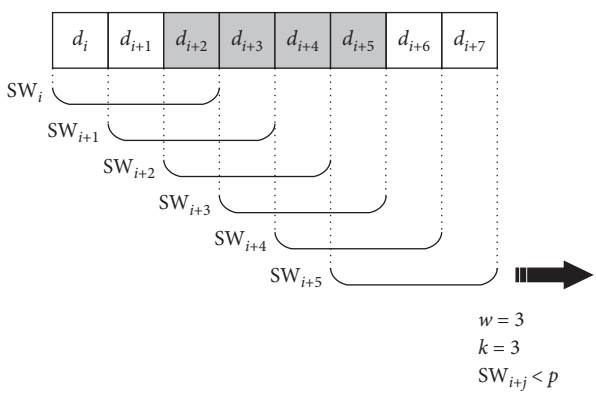


Figure 2: Sliding window mechanism detects abnormal events.

set, and each time the frequency of occurrence of all possible two data pairs in the SW at various distances is counted. When $w = 3$, at some time, $SW_i = \{d_i, d_{i+1}, d_{i+2}\}$; the state pairs are d_i, d_{i+1} and d_{i+1}, d_{i+2} when the distance is 1, and the state pair is d_i, d_{i+2} when the distance is 2. When the length of SW is w and the data pair may be $\{AA, AB, AC, BA, BB, BC, CA, CB, CC\}$, the state matrix constructed by this TrainingData can be expressed as a matrix with 9 rows and $w - 1$ columns, of which each item Num (distance, StatePair) counts the total number of times the StatePair at the distance appearing in the sliding window when it slides across the entire TrainingData. Finally, the probability is calculated by distance to obtain the final probabilistic matrix, that is, the feature matrix. Figure 3 and equation (2) show a probabilistic matrix:

$$P(\text{distance, StatePair}) = \frac{\text{Num}(\text{distance, StatePair})}{\Sigma_{\text{StatePair}}\text{Num}(\text{distance, StatePair})},$$
$$\text{distance} = 1, 2, \ldots, w - 1. \tag{2}$$

It represents the probability of occurrence of each state pair at each distance as a feature matrix for the training data.

The probability matrix construction is expressed in Algorithm 1:

	1	2	…	$w - 1$
AA	$P(1, AA)$	$P(2, AA)$		$P(w - 1, AA)$
AB	$P(1, AB)$	$P(2, AB)$		$P(w - 1, AB)$
…		$P(distance, StatePair)$		
CC	$P(1, CC)$	$P(2, CC)$		$P(w - 1, CC)$

Figure 3: An example of a probabilistic matrix.

3.2.3. Improved Probabilistic Detector. The credibility detection for IoT data also needs to meet the real-time requirement of IoT. A large number of multiplications in the actual calculation process of the original algorithm consume a lot of computing resources, and the result of multiplying the P values is too small and makes it difficult to set threshold. And when a small P but not 0 appears, it may affect the probability calculation of the entire sliding window, leading to misjudgment.

We improve the probabilistic detector. The test data is divided into blocks as the same way of training data to form TestData $= \{d_1, d_2, d_3, \ldots, d_n\}$. Setting a sliding window $SW_i = \{d_i, d_{i+1}, d_{i+2}, \ldots, d_{i+w-1}\}$, we each time extract the corresponding P (distance, StatePair) of distance $1 \sim w - 1$ in SW from the probabilistic matrix and calculate $P(SW_i) = \Sigma_{\text{distance}}\Sigma_{\text{StatePair in distance}}P(\text{distance, StatePair})$. For example, for $SW = \{A, B, C, B\}$, $P(SW) = P(1, AB) + P(1, BC) + P(1, CB) + P(2, AC) + P(2, BB) + P(3, AB)$.

When a certain P is small and the surrounding value is large, since $P > 0$, that is, it appears in the training data, so we determine this situation as credible data; if $P = 0$ or the data value is outside the block range, it is considered that a point anomaly occurs at this time, and we should set $P(SW_i) = 0$. The improved probability detector proposed in this paper can reduce the overhead of computing resources, adapt to the real-time requirements of IoT, and be convenient to set threshold. Experiments show that it can maintain a high detection rate of over 97%.

The threshold p is selected based on the performance of the algorithm on the test data. We use four metrics, accuracy, precison, recall, and $F1$ to evaluate the performance of our algorithm. The improvement of some metrics will lead to the decline of the other metrics. p is the most effective one when the four metrics are all higher. That is, the

```
Input: TrainingData
Output: ProbMatrix
(1) for SW slides in TrainingData
(2)     calculate Num(distance, StatePair);
(3) for distance from 1 to w − 1
(4)     calculate P(distance, StatePair);
(5) construct ProbMatrix;
```

Algorithm 1: ProbMatConstruct.

```
Input: TestData, Probabilistic Matrix
Output: credible or incredible
(1) for SW_i slides in TestData
(2)     for distance from 1 to w − 1
(3)         for StatePair in distance
(4)             if P(distance, StatePair) == 0
(5)                 P(SW_i) = 0;
(6)                 calculate SW_{i+1};
(7)             else
(8)                 P(SW_i) += P(distance, StatePair);
(9) for i from 1 to TestData
(10)    if (∃P(SW_{i+j}) < p, j = 0, 1, 2, ..., t) ∧ (t + 1 ≥ k)
(11)        return incredible;
(12) return credible;
```

Algorithm 2: Credibility Detect.

position where the four metrics intersect corresponds to the best p, which we will describe in detail in Section 4.1.

For a piece of test data, the following equation is determined as incredibility:

$$\begin{cases} P(\mathrm{SW}_i) = \begin{cases} \Sigma_{\text{distance}} \Sigma_{\text{StatePair in distance}}, & \forall P > 0; \\ 0, & \exists P = 0, \end{cases} \\ P\left(\mathrm{SW}_{i+j}\right) < p, \quad j = 0, 1, 2, \ldots, t; \\ t + 1 \geq k. \end{cases} \tag{3}$$

The improved probabilistic detector is expressed in Algorithm 2.

3.3. Area-Shared-Context. A large number of studies [34–37] have shown that consideration of context is necessary for anomaly detection. The context of IoT data means that the working state of IoT devices will change in different contexts, and the generated IoT data will exhibit different behavioral characteristics.

For a single IoT device A, we define Context(A) as all of its existing contexts and $C(A)$ as the current context. In this case, we use training data to construct the probabilistic matrix as the feature matrix for each context and then use the test data to construct probabilistic matrix to compare with each feature matrix to determine the context. However, the credibility of data before the determination is unknown, so this way is an incredible process, and the contextual anomalies may be missed.

For a plurality of IoT devices of same type A, devices which have the same context as the surrounding devices in the time stamp are divided into one area, and others without such characteristics are divided into different areas. The IoT environment often requires multiple similar devices to work together, and these IoT devices often work in the same state as other devices around them. For example, when the workers in downhole operation safety monitoring system are at −200 m, −450 m, and −700 m depth, the temperature, air volume, humidity, dust concentration, lighting status, and other data will show a coordinated behavior pattern. We construct a credible context determination process based on Hypothesis 3 in the threat model, which constructs the probabilistic matrix of data from all devices A and average it, and then use the result to compare with feature matrix of each context to determine $C(A)$ of the area. If a device's context is offset, it has little effect on the area-context and can be detected.

For many types of IoT devices, we define AreaSharedContext to represent the correspondence between all area-shared-contexts and each type of device's context, ASC(area) to represent the area-shared-context of the current area. For example, Context(A) = $\{1, 2, 3\}$, Context(B) = $\{1, 2\}$, AreaSharedContext = $\{(1, 1), (1, 2), (2, 1), (2, 2)\}$, ASC(area) = 1, which indicates that there are four possibilities for the area-shared-context of the area within device types A and B, and the current area-shared-context is 1, corresponding to $C(A) = 1$ and $C(B) = 1$. For an area with only two device types, we discuss the following two situations:

(1) A and B are not relevant. That is, the contexts of two device types have no effect on each other. For example, when A has 3 contexts, device B has 2 contexts, and there are $2 \times 3 = 6$ area-shared-contexts in this area.

(2) A and B are relevant. That is, the context of device type A can have an impact on B, and the same for B to A. It is very common in IoT, for example, the greenhouse environmental intelligent monitoring system can remotely monitor the temperature, humidity, light, ground temperature, soil moisture, and other parameters in the greenhouse in real time by configuring IoT sensors; inside the greenhouse, light and ground temperature are related to time and weather; and air humidity and soil moisture are positively correlated. Figure 4 shows a case of relevant context and its AreaSharedContext.

For the case described in Figure 4, there are only four area-shared-contexts:
AreaSharedContext = {(1, 1), (2, 1), (2, 2), (3, 1)}, and we can see that when A's context is 1, B's context cannot be 2. In this case, the contexts of A and B interact with each other and are mutually constrained in the determination process of area-shared-context.

Based on all the above situations, this paper proposes the concept of area-shared-context for the IoT environment, which divides unrelated devices into different areas, so that all devices in each area have the same area-shared-context and each area-shared-context has a context value for each type of device.

After defining the area-shared-context, we need to construct AreaSharedContext from the devices to be detected. (1) If the contexts of all device types are clearly defined or can be visually seen, it can be directly generated. (2) If the context definition is not clear, based on the idea of the buffer detector, we provide the **algorithm AreaSharedContextConstruct**: the context of each device type is recorded in the buffer on the timestamp. After storing a large amount of test data's contexts, the records with higher frequency are reserved as an item in AreaSharedContext, and others will be deleted, as shown in Figure 5.

The algorithm in Figure 5 has a high time complexity but needs to be executed only once during the training phase, so it will not affect real-time performance.

3.4. Determination of Area-Shared-Context. This section describes the credibility detection methods in a single area.

We define DeviceType = {DType | Dtype ∈ all device types}, where DType represents the device type.

Define Num(DType) as the amount of devices with type DType in the area.

Define Data(DType, number) to represent a piece of IoT data generated by the device of which type is DType and ID is number.

Define PMatrix(DType, number) to represent the probabilistic matrix constructed from Data(DType, number) by using the **algorithm ProbMatConstruct**, that is, PMatrix(DType, number) = **ProbMatConstruct**(Data(DType, number)).

Define CMatrix(DType, C(DType)) to represent the feature matrix of device DType in C(DType) context. The CMatrix is constructed by the training data, which is the probabilistic matrix corresponding to the training data.

Define a function Difference(m_1, m_2) to calculate the Euclidean distance between matrix m_1 and m_2.

We propose a function F(ASC(area)), which indicates the possibility that the current area's area-shared-context is ASC(area). The specific method is averaging the probabilistic matrices constructed from the data generated by each type of device in the area and then calculating the distance between them and all the feature matrices corresponding to contexts of each device type in ASC(area). The smaller the sum of the differences is, the more the area-shared-context tends to be ASC(area), which is expressed as the following equations:

$$F(\mathrm{ASC}(\mathrm{area})) = \frac{\Sigma_{D\mathrm{Type}\in\mathrm{DeviceType}}[\mathrm{Num}(\mathrm{DType}) \times \mathrm{Differences}]}{\Sigma_{D\mathrm{Type}\in\mathrm{DeviceType}}\mathrm{Num}(\mathrm{DType})}, \tag{4}$$

$$\mathrm{Differences} = \mathrm{Difference}\left(\frac{\sum_{\mathrm{number}=1}^{\mathrm{Num}(D\mathrm{Type})} P\mathrm{Matrix}(\mathrm{DType}, \mathrm{number})}{\mathrm{Num}(\mathrm{DType})}, C\mathrm{Matrix}\right), \tag{5}$$

$$C\mathrm{Matrix} = C\mathrm{Matrix}(D\mathrm{Type}, \mathrm{ASC}(\mathrm{area}) \longrightarrow C(D\mathrm{Type})), \tag{6}$$

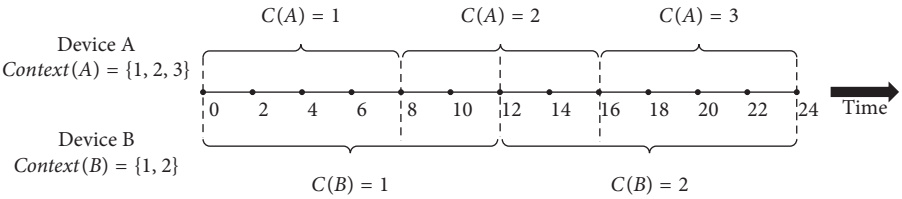
where ASC(area) $\longrightarrow C(D\mathrm{Type})$ means the contexts of each device type in ASC(area).

We take the minimum value of F(ASC(area)) to determine the area-shared-context, expressed as follows:

$$\begin{cases} F(x) = \min(F(\mathrm{ASC}(\mathrm{area}))), \mathrm{ASC}(\mathrm{area}) \in \mathrm{AreaSharedContext}; \\ \mathrm{ASC}(\mathrm{area}) = x. \end{cases} \tag{7}$$

Finally, we extract each C(DType) according to the determined value ASC(area) from the AreaSharedContext and load the corresponding CMatrix to detect the credibility of IoT data by using the **algorithm Credibility Detect**.

The area-shared-context determination is expressed in Algorithm 3.



	A	B
$ASC(area) = 1$	$C(A) = 1$	$C(B) = 1$
$ASC(area) = 2$	$C(A) = 2$	$C(B) = 1$
$ASC(area) = 3$	$C(A) = 2$	$C(B) = 2$
$ASC(area) = 4$	$C(A) = 3$	$C(B) = 2$

Figure 4: Relevant context of multi-type devices and its AreaSharedContext.

3.5. Process of Credibility Detection for IoT Data. Based on the method proposed in the above three sections, we define the data packet from each device as DataPackage = (area, DType, Data) and give the processing flow as follows:

(1) Classify training data by device type and context, and use **algorithm ProbMatConstruct** to construct feature matrices of all contexts.

(2) Construct the AreaSharedContext by using the **algorithm AreaShare dC ontextConstruct** on training data.

(3) Collect DataPackage of all devices to be detected at the same time, and classify the data packets according to area. Each area separately performs credibility detection.

(4) Use the **algorithm ASC De termine** on all IoT data in each area to calculate the area-shared-context.

(5) According to the area-shared-context, load the corresponding feature matrix, and use **algorithm Cre di bility De tect** to detect each piece of data.

(6) Send credible data to the big data analysis platform, and send incredible data to the Incredible Data Processing Module.

4. Results and Discussion

We implement our method in MATLAB R2016a and design the following experiments to evaluate the performance of the method. We run different programs to collect power data on two demoboards: ZYNQ7350 (model: XC7Z035-2FFG676) and ZYNQ7020 (model: XC7Z020-2CLG400I), to build the data set required for the experiment. Programs with different computational loads are executed on demoboards to simulate different contexts of one device. Different demoboards correspond to different devices, and the area-shared-context is simulated by the program designed on the timestamp. After preprocessing, the final data set includes four types of devices, where device A is relevant to B and device C is relevant to D, forming 16 area-shared-contexts. Our incredible data are divided into point abnormal data, behavioral abnormal data, and contextual abnormal data, of which point anomalies are generated by injecting outliers, behavioral anomalies are generated by interference, and contextual anomalies are generated by running a program on the timestamp that does not belong to the current area-shared-context.

We evaluate the performance of the proposed method and compare it with two similar methods of contextual anomaly detection. Finally, the time cost of our method is evaluated. The results show that our method has a higher detection rate with lower false positives, and the time cost is acceptable.

4.1. Performance Evaluation. We first evaluate the performance of context determination in this paper to select the sliding window size parameter and data length parameter of the algorithm. We set the WinSize from 4 to 8 and the data length from 10 to 100 and calculate the accuracy rate of context determination in each case. We extract 471 groups of data with length 10 to 100 from the dataset and mark the context. The context of each piece of data is determined by the method proposed in this paper, and then the accuracy rate of context determination is calculated by comparing it with the actual context. The effect of the size parameter and the length parameter on the accuracy of context determination is shown in Figure 6.

As shown in Figure 6, when the data length is 100 and the sliding window size is 5, the accuracy rate of context determination reaches the highest level of 98.73%. Therefore, the subsequent experiments are performed with a sliding window size of 5 and a data length of 100.

Then, we evaluate the effect of parameter k on the performance of the improved probabilistic detector when detecting a single device. We select 287 normal and abnormal data with a length of 100 and use false negatives and false positives to evaluate the performance of the algorithm. Table 1 shows the effect of parameter k.

As can be seen from Table 1, when $k = 3$, the algorithm has the best performance on a single device because its false negatives and false positives are lower than others. Therefore, the subsequent experiments are performed with k set to 3.

Finally, we evaluate the effect of the number of data blocks and the threshold p on the performance of credibility

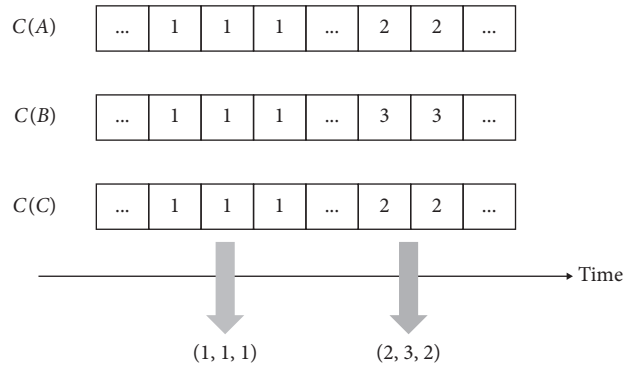


Figure 5: Algorithm for area-shared-contexts' construction.

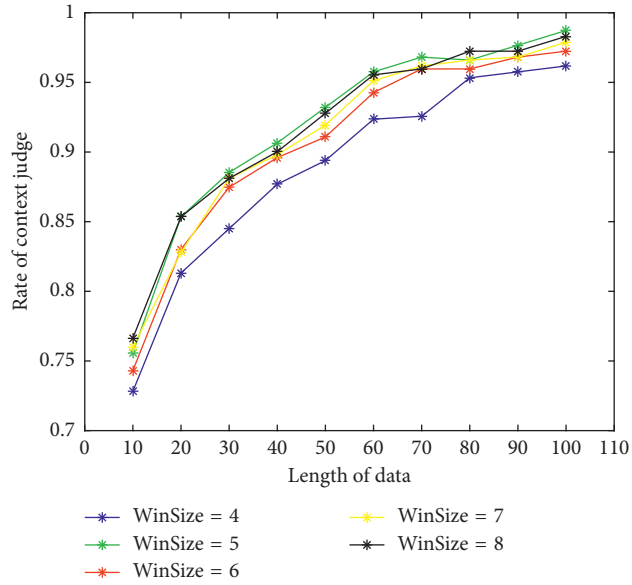


Figure 6: The effect of size and length on the accuracy of context determination.

Input: area, Data from all devices in an area
Output: ASC(area)
(1) for ASC(area) ∈ AreaSharedContext
(2) extract *C*Matrix corresponding to *C*(DType) in ASC(area);
(3) calculate the difference between *C*Matrix and average *P*Matrix;
(4) calculate *F*(ASC(area));
(5) for x = all ASC(area)
(6) calculate x = ASC(area) of minimum *F*;
(7) return x;

Algorithm 3: ASC Determine.

Table 1: Effect of k on the performance of the improved probabilistic detector.

k	2		3		4		5	
p	False negatives	False positives	False negatives	False positives	False negatives	False positives	False negatives	False positives
0.015	17	5	31	3	47	0	87	0
0.020	5	8	7	5	35	0	74	0
0.025	3	12	4	5	31	3	66	0
0.030	3	14	4	6	28	3	63	2

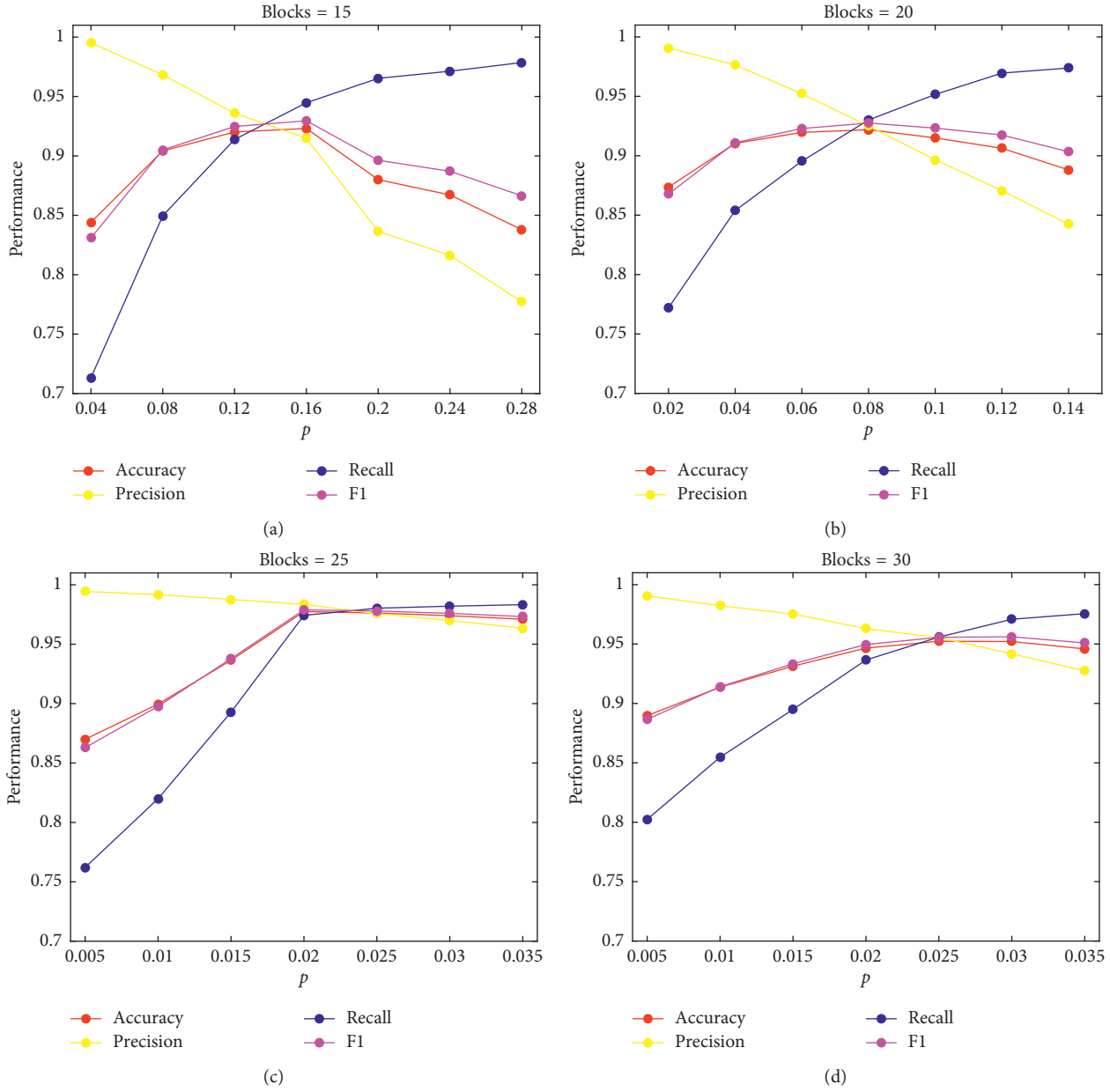


Figure 7: Effect of threshold p and number of data blocks on the performance of credibility detection. (a) Blocks = 15. (b) Blocks = 20. (c) Blocks = 25. (d) Blocks = 30.

detection. In this paper, we choose four metrics, accuracy, precison, recall, and $F1$ constructed by four basic values (TP, TN, FP, and FN) to evaluate the performance, expressed as equations (8)–(11), respectively:

$$\text{accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}, \tag{8}$$

$$\text{precison} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{9}$$

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{10}$$

$$F1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precison} + \text{recall}}. \tag{11}$$

When these four metrics are all high, it reflects the better performance of the model. This experiment used $67 * 30$ credible data and 629 groups of abnormal data, and each group contained small amount of incredible data. Finally there are more than 2000 pieces of data within credible and incredible data. Figures 7(a)–7(d) separately show the effect of threshold p on the performance of credibility detection when the data are divided into 15, 20, 25, and 30 blocks. The improvement of some metrics will lead to the decline of the other metrics. p is the most effective when the four metrics

Table 2: Performance of three methods for three types of anomalies.

	Our method (%)	KNN probabilistic (%)	k-means-Gaussian (%)
Point anomaly	98.91	95.18	98.76
Behavioral anomaly	96.51	91.07	73.09
Contextual anomaly	97.28	78.50	73.96
Total	97.43	88.04	80.45

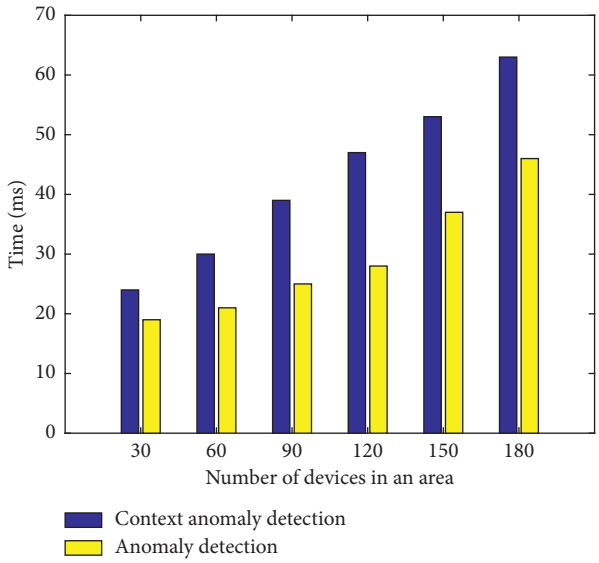


Figure 8: Comparison of time cost in two ways.

are all higher. Because the different blocks will lead to the changes of the setting of best p, the horizontal axis of Figures 7(a)–7(d) takes different values in order to show the best performance.

As can be seen from Figure 7, when the number of data blocks is 25, the performance of the four metrics is the highest when they are aggregated. In this case, when $p = 0.02$, accuracy = 97.74%, precison = 98.36%, recall = 97.43%, and $F1 = 97.89\%$. Therefore, considering the four metrics, the data block number is set to 25, and the threshold p is set to 0.02, which is most suitable for the data set of this paper.

4.2. Comparison Study. In order to evaluate the performance of three types of anomalies in this paper, point anomalies, behavioral anomalies, and contextual anomalies, we choose two similar algorithms of contextual anomaly detection to compare with our method. One is a KNN-based context determination combined with a probabilistic detector proposed in Reference [33], and the other is a k-means-based context determination combined with a Gaussian detector proposed in Reference [32]. We evaluate the performance of three methods when detecting 643 point anomalies, 918 behavioral anomalies, and 772 contextual anomalies and calculate (the number of incredible data detected by the method)/(actual anomalies). Table 2 shows the performance of the three methods for three types of anomalies.

We can see from Table 2 that, for point anomaly, all three methods have good performance, but the accuracy rate of context determination in our method is higher, so our final detection rate is the highest; for behavioral anomalies, the Gaussian detector used by the k-means-Gaussian method cannot detect behavioral anomalies effectively, so its performance is much lower than the other two methods; for contextual anomaly, the KNN probabilistic method will cause context misjudgment when facing longer anomalies, resulting in a lower detection rate, but our method can still show good performance due to the characteristics of area. In summary, our method is superior to other two methods for three types of anomalies with a high level of performance.

4.3. Time Cost Analysis. Finally, we evaluate the performance in time cost of our method. We set the number of devices in an area from 30 to 180 and record the time cost of two ways that using the context determination of this paper or directly detecting without determining the context. Figure 8 shows the results of the evaluation.

The time overhead of the proposed method in this paper is less than 35%. In the case of 180 devices, the time used in this method is only 0.063 seconds, which indicates that the time cost of our method is acceptable.

5. Conclusions and Future Work

Aiming at the problem that IoT is easy to generate incredible data and the analysis platform makes untrusted results, this paper proposes a credibility detection based on area-context for IoT data. This method introduces the concept of area-shared-context and combines it with the probabilistic detector, so it can effectively detect point anomalies, behavioral anomalies, and contextual anomalies. The credibility of the context determination process of the context-relevant device that satisfies the area characteristics is higher, making the overall detection performance better. Experiments show that our method can reach a high level of performance with more than 97% in all kinds of metrics, which is better than the similar methods. And time cost of our method is acceptable, which meets the real-time requirements of the IoT system. Because the proposed method in this paper can detect multiple devices in an area at the same time, it is suitable for the IoT environment with a large number of devices. And the characteristics of area are more suitable for the edge computing architecture, so we can further improve the efficiency when deploying the credibility detection center to edge nodes.

Future work will need to detect the running status of the sensor device in the IoT system and the security of data transmission. It will also focus on the situation awareness of the overall IoT system and the prediction of the future system security status.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was funded by Wuhan Frontier Program of Application Foundation (grant number 2018010401011295), Joint Funds of the National Natural Science Foundation of China (grant number U1936122), the Fundamental Research Funds for the Central Universities (grant number 2042017kf024), Program of Chinese Academy of Engineering (grant number 212000005) and Science and Technology Program of the Headquarters of State Grid Corporation of China (Research on Key Technologies of Energy Internet Mobile and Interconnection Security).

References

[1] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (IoT): a vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[2] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: a survey," *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[3] M. Mehdi, A. F. Ala, S. Sameh, and M. Guizani, "Deep learning for IoT big data and streaming analytics: a survey," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 2923–2960, 2018.

[4] J. Qiu, L. Du, D. Zhang, S. Su, and Z. Tian, "Nei-TTE: intelligent traffic time estimation based on fine-grained time derivation of road segments for smart city," *IEEE Transactions on Industrial Informatics*, vol. 99, 2019.

[5] H. S. Hassanein and S. M. A. Oteafy, "Big sensed data challenges in the internet of things," in *Proceedings of the 13th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, Ottawa, ON, Canada, June 2017.

[6] W. Ding, X. Jing, Z. Yan, and L. T. Yang, "A survey on data fusion in internet of things: towards secure and privacy-preserving fusion," *Information Fusion*, vol. 51, pp. 129–144, 2018.

[7] H. Shin, H. K. Lee, H. Cha, S. W. Heo, and H. Kim, "IoT security issues and light weight block cipher," in *Proceedings of the International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, pp. 381–384, Okinawa, Japan, February 2019.

[8] Z. Tian, M. Li, M. Qiu, Y. Sun, and S. Su, "Block-DEF: a secure digital evidence framework using blockchain," *Information Sciences*, vol. 491, pp. 151–165, 2019.

[9] H. Zhang and B. Zhao, *Trusted Computing*, Wuhan, China, 2011.

[10] B. Zhao, M. Ni, Y. Shi et al., "A review on the security of embedded system," *Journal of Wuhan University (Natural Science Edition)*, vol. 64, no. 2, pp. 95–108, 2018.

[11] Z. Bo, Z. Xiangyu, X. Shuang et al., "C-TNC: trusted cloud access protocol for openstack," *Journal of Huazhong University of Science and Technology (Natural Science Edition)*, vol. 44, no. 3, pp. 83–88, 2016.

[12] Q. Tan, Y. Gao, J. Shi, X. Wang, B. Fang, and Z. Tian, "Toward a comprehensive insight into the eclipse attacks of tor hidden services," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1584–1593, 2019.

[13] J. L. Flores and I. Mugarza, "Runtime vulnerability discovery as a service on industrial internet of things (IIoT) systems," in *Proceedings of the IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA)*, pp. 948–955, Turin, Italy, September 2018.

[14] Z. Tian, C. Luo, J. Qiu, X. Du, and M. Guizani, "A distributed deep learning system for web attack detection on edge devices," *IEEE Transactions on Industrial Informatics*, vol. 99, 2019.

[15] A. Gupta, O. J. Pandey, M. Shukla et al., "Computational intelligence based intrusion detection systems for wireless communication and pervasive computing networks," in *Proceedings of the IEEE International Conference on Computational Intelligence & Computing Research*, Enathi, India, December 2014.

[16] M. Ni, B. Zhao, F. Wu et al., "CREBAD: Chip radio emission based anomaly detection scheme of IoT devices," *Journal of Computer Research and Development*, vol. 55, no. 7, pp. 1451–1461, 2018.

[17] Z. Tian, W. Shi, Y. Wang et al., "Real-time lateral movement detection based on evidence reasoning network for edge computing environment," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 7, pp. 4285–4294, 2019.

[18] C. Dietz, R. Labaca Castro, J. Steinberger et al., "IoT-botnet detection and isolation by access routers," in *Proceedings of the 9th International Conference on the Network of the Future (NOF)*, pp. 88–95, Poznan, Poland, November 2018.

[19] Z. Tian, X. Gao, S. Su, J. Qiu, X. Du, and M. Guizani, "Evaluating reputation management schemes of internet of vehicles based on evolutionary game theory," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 6, pp. 5971–5980, 2019.

[20] Z. Tian, S. Su, W. Shi, X. Du, M. Guizani, and X. Yu, "A data-driven method for future Internet route decision modeling," *Future Generation Computer Systems*, vol. 95, pp. 212–220, 2019.

[21] T. M. Nguyen, Q. M. Jonathan Wu, and H. Zhang, "Bounded generalized Gaussian mixture model," *Pattern Recognition*, vol. 47, no. 9, pp. 3132–3142, 2014.

[22] B. Yu, Z. Xia, and J. Wang, "Anomaly detection algorithm based on Gaussian process model," *Computer Engineering and Design*, vol. 37, no. 4, pp. 914–920, 2016.

[23] R. A. Maxion and K. M. C. Tan, "Anomaly detection in embedded systems," *IEEE Transactions on Computers*, vol. 51, no. 2, pp. 108–120, 2002.

[24] A. Cuzzocrea, E. Mumolo, and R. Cecolin, "Runtime anomaly detection in embedded systems by binary tracing and hidden Markov models," in *Proceedings of the IEEE 39th Annual Computer Software and Applications Conference*, pp. 15–22, Taichung, Taiwan, July 2015.

[25] S. Yan, L. I. Yong-Zhong, and L. Jun-Sheng, "Hybrid anomaly intrusion detection method using HMM and STIDE," *Computer Engineering*, vol. 34, no. 3, pp. 181-182, 2008.

[26] F. R. Castella, "Sliding window detection probabilities," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-12, no. 6, pp. 815–819, 1976.

[27] M. Zandrahimi, H. R. Zarandi, and M. H. Mottaghi, "Two effective methods to detect anomalies in embedded systems," *Microelectronics Journal*, vol. 43, no. 1, pp. 77–87, 2012.

[28] D. H. Summerville, K. M. Zach, and Y. Chen, "Ultra-light-weight deep packet anomaly detection for internet of things devices," in *Proceedings of the IEEE 34th International*

Performance Computing and Communications Conference (IPCCC), Nanjing, China, December 2015.

[29] Z. Ding and M. Fei, "An anomaly detection approach based on isolation forest algorithm for streaming data using sliding window," *IFAC Proceedings Volumes*, vol. 46, no. 20, pp. 12–17, 2013.

[30] C.-I. Chang, Y. Wang, and S.-Y. Chen, "Anomaly detection using causal sliding windows," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 8, no. 7, pp. 3260–3270, 2015.

[31] L. Guang, W. Jie, L. Jing, and L. Yue, "Application of sliding nest window control chart in data stream anomaly detection," *Symmetry*, vol. 10, no. 4, p. 113, 2018.

[32] M. A. Hayes and M. A. M. Capretz, "Contextual anomaly detection in big sensor data," in *Proceedings of the IEEE International Congress on Big Data*, pp. 64–71, Anchorage, AK, USA, June 2014.

[33] F. Ehsani-Besheli and H. R. Zarandi, "Context-aware anomaly detection in embedded systems," *Advances in Dependability Engineering of Complex Systems*, vol. 582, pp. 151–165, 2018.

[34] J. Mauro, M. Nieke, C. Seidl et al., "Anomaly detection and explanation in context-aware software product lines," in *Proceedings of the 21st International Systems and Software Product Line Conference—Volume B on ZZZ—SPLC'17*, Sevilla, Spain, September 2017.

[35] E. Pardo, D. Espes, and P. Le-Parc, "A framework for anomaly diagnosis in smart homes based on ontology," *Procedia Computer Science*, vol. 83, pp. 545–552, 2016.

[36] P. Duessel, C. Gehl, U. Flegel, S. Dietrich, and M. Meier, "Detecting zero-day attacks using context-aware anomaly detection at the application-layer," *International Journal of Information Security*, vol. 16, no. 5, pp. 475–490, 2017.

[37] D. Renaudie, M. A. Zuluaga, and R. Acuna-Agost, "Benchmarking anomaly detection algorithms in an industrial context: dealing with scarce labels and multiple positive types," in *Proceedings of the IEEE International Conference on Big Data (Big Data)*, pp. 1228–1237, Seattle, WA, USA, December 2018.