A mobile edge–cloud collaboration outlier detection framework in wireless sensor networks

Cong Gao1,2 | Guohao Song1 | Zhongmin Wang1,2 | Yanping Chen1,2

1 School of Computer Science and Technology, Xi’an University of Posts and Telecommunications, Xi’an, China
2 Shaanxi Key Laboratory of Network Data Analysis and Intelligent Processing, Xi’an University of Posts and Telecommunications, Xi’an, China

Correspondence
Cong Gao, School of Computer Science and Technology, Shaanxi Key Laboratory of Network Data Analysis and Intelligent Processing, Xi’an University of Posts and Telecommunications, Xi’an, Shaanxi 710121, China.
Email: cgao@xupt.edu.cn

Abstract
Wireless sensor networks (WSNs) are extensively deployed to collect various data. Due to harsh environments and limitation of computing and communication capabilities of sensor nodes, the quality and reliability of sensor data are compromised by outliers. With the advent of 5G, sensors tend to generate increasingly more complex data. When faced with big data, traditional outlier detection methods relied on sensor nodes and remote cloud are unable to accord satisfactory performance in terms of delay and energy consumption. To address this problem, we propose a mobile edge–cloud collaboration outlier detection framework. Outlier detection is performed by edge nodes between the remote cloud and the underlying WSNs, while the training and updating of detection model are conducted on the cloud. A fast angle-based outlier detection method is developed to obtain training data. The detection model is constructed based on support vector data description. An online learning-based iterative optimization scheme is devised to update the detection model. Besides, a fuzzy concept is incorporated into the detection model to alleviate the problem of loose decision boundary. Extensive experiments are conducted on real-world data set. Simulation results show that our model is superior to three popular methods in terms of delay and energy consumption. In addition, when the percentage of operational nodes is 60%, our proposal prolongs the network lifetime by 14.2% to 69.8% compared to the three methods.

1 | INTRODUCTION

With the rapid development of microelectronics and wireless technologies, wireless sensor networks (WSNs) have been extensively applied to a large variety of fields, such as agriculture [1], healthcare [2], industry [3–5], and smart home [6]. A WSN is a distributed network architecture formed by deploying a large number of sensor nodes in an area. It aims to collect desired data from the region of interest. However, as the core component of a WSN, sensor nodes are prone to generate outliers in the collected data [7]. The main reasons lie in the following three aspects [8].

1) Resource limitation. Sensor nodes possess stringent resource constraints, such as computing power, storage space, and communication bandwidth. Due to limited resources and capabilities, data generated by sensor nodes are apt to be inaccurate and unreliable. In addition, sensor nodes are battery powered. When battery is draining, the performance of a sensor node tends to deteriorate. Then, outliers come out more frequently.

2) Harsh environments. Due to unintended environments like dynamic climate change or harsh phenomenon in uncontrolled areas, sensors might report unstable data. Therefore, sensor nodes suffer from radio interference and potential damage from a harsh environment.

3) Malicious attacks. As sensor nodes are randomly deployed, they are prone to various kinds of malicious attacks. For instance, the collection and processing of data may be attacked by attackers. Moreover, attackers can also inject malicious codes to make sensor nodes produce erroneous data.

The above issues may lead to the occurrence of outliers. An outlier, also known as anomaly, is a pattern in data which does not conform to a well-defined notion of normal behaviour [9].
As the existence of an outlier may significantly compromise the accuracy and reliability of data, it is essential to efficiently identify outliers. Moreover, the occurrence of outliers may indicate events of interest, such as equipment failure, sudden environmental change, and security breach. Recently, outlier detection in WSNs has attracted much research attention. It is widely used for various real-world applications, such as fraud detection, intrusion detection, environmental monitoring, health and medical monitoring, localization, and target tracking.

For WSNs, outlier detection techniques often involve computation and communication among different sensor nodes. In general, outlier detection techniques are divided into centralized structure and distributed structure. In a centralized structure, each node transmits the received data to a central location (e.g., cloud platform). Besides delay and congestion induced by bottleneck effect, this type of structure also tends to cause highly inefficient resource utilization. Large amount of raw data needs to be transmitted from sensors to cloud platform. This entails substantial channel interference and energy consumption. However, in fact, only a small fraction of the transmitted data are anomalous [10]. Therefore, to improve the efficiency of resource utilization as well as responsiveness, a distributed outlier detection structure is preferred by researchers. However, outlier detection is challenging for sensors with limited resources due to the elusive nature of outliers and the variability of surrounding physical environment. Solutions based on various approaches have been proposed. In [11], outlier detection approaches are classified as six groups: statistical-based, clustering-based, classification-based, spectral decomposition-based, nearest neighbour-based, and other methods. However, most existing solutions are either computationally expensive or overwhelmed by communication overhead. Thus, the applicability to resource-constrained WSNs is limited.

In [12], the authors present a test methodology for the comparison of edge computing architecture and cloud computing architecture for anomaly detection systems. The experiments are conducted based on the implementations of deep learning algorithms. The major drawback of deep learning algorithms is that they usually require high computational power. The proposed methodology mainly concentrates on comparisons among deep learning algorithms. The applicability of this proposal is not wide. In [13], the authors propose a hybrid bacteria foraging algorithm to address the problem of edge task placement. The proposed algorithm aims to conduct task scheduling in edge network and it is able to explore a large search space and obtain a feasible solution in polynomial time. However, as the task placement is an NP-complete problem, the proposed model introduces some constraints. In [14], the authors describe an autonomous anomaly analysis framework for clustered cloud or edge resources. It aims to find out the cause of a user-aware anomaly in the underlying infrastructure by hidden Markov models. Experiments are just conducted in clustered cloud computing resources. Edge resources are not covered.

In addition, due to the advent of new information technologies (e.g., 5G) and the integration of sensors, more and more data with complex structure are generated. When conducting outlier detection for large amount of data with multi-features, existing approaches are confronted with excessive computation-intensive and energy-consuming for sensor nodes to accomplish are performed at the edge layer. In addition, an iterative optimization scheme is developed to update the outlier detection model. The main contributions are as follows.

1) A fast angle-based outlier detection algorithm (FastABOD) is introduced to obtain training data. Faced with the increasing dimensionality of high-dimensional data, the concept of distance in TODMs becomes less meaningful. Our FastABOD algorithm turns to angular variation among data points for outlier detection. Compared to the basic ABOD algorithm, our method achieves an improvement in the time complexity by an order of magnitude.

2) Traditional support vector data description (SVDD) method is only applicable to data set whose input space is circular. For an input space with a non-circular distribution, the actual performance of this method cannot be guaranteed. To address this problem, we introduce the concept of kernel function. The employed Gaussian kernel function possesses good robustness and is independent of sample size. When there are outliers in the training data, the traditional SVDD method is likely to accord loose decision boundaries. To address this problem, we introduce fuzzy theory to finding a minimum hypersphere with fuzzy constraints. The above building blocks constitute our f-SVDD package which is the detection model.

3) A mobile edge–cloud collaboration scheme is designed. The accuracy of outlier detection is improved by optimizing the detection model with on-line machine learning techniques. Edge nodes equipped with the f-SVDD package collect data and conduct outlier detection. The cloud trains the f-SVDD package and provides model optimization to the edge nodes.

The remainder of this paper is organized as follows. Section 2 reviews the existing outlier detection methods in WSNs. Section 3 presents our mobile edge–cloud collaboration outlier detection framework. Section 4 evaluates the proposed model with extensive experiments. Finally, conclusions and future work are presented in Section 5.

2 | RELATED WORK

Here, existing outlier detection methods in WSNs are summarized. The six categories of outlier detection approaches for WSNs given by [11] are detailed as follows.
2.1 | Statistical based

The core idea of statistical-based outlier detection is a prior probability distribution model with a confidence interval. Measurements that lie within the confidence interval are considered to be normal, while outliers are outside the confidence interval. Statistical-based outlier detection models can be classified as parameterized and parameterless based on the construction methods. This type of outlier detection is rational in mathematics. Provided a data distribution could be estimated in advance, statistical-based approaches are able to accord satisfactory outlier detection.

In [15], a distributed outlier detection model based on credibility feedback for WSNs is proposed. Final credibility of sensor nodes are derived from initial credibility by credibility feedback and Bayes theorem. Then, the set of outliers is adjusted. The presented message complexity reveals that the proposed model possesses high energy consumption. In [16], a statistical distributed outlier detection method based on Euclidean distance is proposed. The fitting of measurements in time series for forward expansion is conducted based on statistical distribution. Thus, a measurement which exhibits large deviation from the distribution center is considered to be an outlier. In [17], the authors proposed an on-line outlier detection approach based on a segmented sequence analysis (SSA) algorithm. A piecewise linear model of time series sensor data is constructed by SSA. The model is a two-layered distributed model where the first layer is a local detection process at an individual node, whereas the second layer is a centralized detection process at a cluster head.

Drawbacks: in a real-world scenario, the probability distribution of sensor data stream can hardly be obtained in advance. Thus, statistical-based outlier detection models are not easy to put into practice.

2.2 | Clustering based

Clustering-based outlier detection is popular in data mining. Different clusters are formed by grouping measurements with similar behaviours. Each cluster consists of similar measurements which are distinct from measurements in other clusters. For a measurement belongs to no cluster or a cluster whose members are significantly less than other clusters. This measurement is considered to be an outlier. Clustering-based outlier detection is able to work without a prior knowledge of data distribution. Thus, it is suitable for scenario where new data are generated constantly.

In [18], a lightweight $k$-means clustering method is proposed. It aims to improve the energy consumption and processing ability of sensor nodes. However, an appropriate value of $k$ is difficult to determine beforehand.

In [19], the authors proposed a distributed hyperspherical cluster-based algorithm for identifying outliers in measurements from WSNs. Measurements are clustered and the obtained clusters are further merged. The description of clusters sent to other nodes is a compact version. Thus, communication overhead is reduced. However, this method is just able to identify outliers at the level of an individual node. In [20], an outlier detection process called density-based spatial clustering of applications with noise (DBSCAN-OD) is proposed. This method combines the calculation of parameters and class identification in spatial-temporal databases. However, as the real-time outlier detection is conducted in a centralized base station, the issue of bottleneck is inevitable.

Drawbacks: clustering-based outlier detection requires predetermination of appropriate values for key parameters, which is not easy. In addition, the processing of multi-dimensional data set consumes large amount of computation resources.

2.3 | Classification based

Classification-based outlier detection basically consists of two phases: training and testing. In the training phase, a classifier is trained with labelled data. In the testing phase, unlabelled data are tested by the obtained classifier. A test case is identified as normal or abnormal. Existing outlier detection methods in WSNs are mainly based on two mathematical packages: support vector machine (SVM) and Bayesian network.

In [21], one-class support vector machine (OCSVM) is employed to perform outlier detection. Spatial-temporal correlation of sensor data are used to represent the relations among sensors. Thus, the correlation graph of a WSN can be established. Based on the readings given by sensors, the correlation graph is able to detect anomaly without any information of threshold. In [22], the authors proposed an unsupervised outlier detection scheme for unlabelled data in WSNs. One of the one-class SVM variants, centered hyper-ellipsoidal SVM (CESVM), is used as a classifier to detect outliers. In [23], an OCSVM-based method for outlier detection is proposed. It is able to reduce computation complexity of the training and the testing phase. In general, SVM-based detection possesses high accuracy and low false positive rate. However, if there are outliers in the training data, the performance of OCSVM becomes very poor. In [24], an outlier detection model with layered structure is proposed. The first layer introduces Bayes classifier to individual sensor node. The second layer determine whether there are outliers by aggregating results of different sensor nodes.

Drawbacks: classification-based outlier detection is confronted with high computation cost when dealing with multi-dimensional data. Moreover, the performance of classifier is heavily dependent on the quality of training data. Whereas, the acquisition of appropriate training data is difficult in certain circumstances.

2.4 | Spectral decomposition based

The basic idea of spectral decomposition-based outlier detection is as follows. Certain key features of data are obtained by spectral analysis methods. The extracted principal components are used for dimensionality reduction of multi-dimensional
data. If a measurement does not conform to the constitution of minimum principal components, it is considered as an outlier. Spectral decomposition-based outlier detection is suitable for high-dimensional data.

In [25], a distributed outlier detection model based on one-class principal component classifier (OCPCC) is proposed. Outlier detection is based on spatial correlation among sensor nodes. Each node in a cluster constructs a local normal reference model which is sent to the cluster head. The cluster head obtains a global normal reference model by integrating the received local models. Finally, the global normal reference model is sent back to local nodes for outlier detection. However, this model suffers from high false positive rate.

In [26], the authors proposed a novel local on-line outlier detection method based on principal component analysis (PCA). As methods based on the original PCA usually suffer from high computational overhead, the authors employ a recursive subspace tracking approach for PCA to address this problem.

Drawbacks: spectral decomposition-based outlier detection relies on the selection of appropriate model. This process requires a large amount of computation.

### 2.5 Nearest neighbour based

Nearest neighbour-based outlier detection uses distance to formulate the relations among data. For both univariate continuous data and multivariate continuous data, the most popular choice is Euclidean distance. If a measurement is far away from its neighbours, it is considered to be an outlier. Namely, the degree of anomaly is measured by distance. This idea performs well for low-dimensional data.

In [27], an outlier detection method based on local outlier factor (LOF) for machinery condition monitoring is proposed. Time domain features are extracted from segments of data. Then, PCA is applied to the obtained features. The degree of anomaly for a segment is evaluated by LOF calculated based on principal components. In [28], a $k$-nearest neighbour outlier detection method in WSNs is proposed. The computation complexity is reduced by remapping the outlier detection task from hypersphere detection region to hypergrid detection region. However, this method should be tested on more data with dynamic variation.

In [29], a LOF-based model is proposed. The average distance from a measurement to its nearest neighbours is denoted by $d_n$. The average of the nearest neighbours’ distances to their nearest neighbours is denoted by $d_n^{-1}$. The LOF uses the ratio of $d_n$ to detect outliers. In [30], a SVM based on $k$-nearest neighbour (KNN-SVM) algorithm for outlier detection in WSNs is proposed. The scale of training data is reduced by the KNN technique. Thus, both training time and optimizing time are shortened. In order to identify outliers, measurements are mapped to feature spaces by kernel function. However, the involvement of KNN and SVM introduces high computational overhead.

Drawbacks: for multi-modal data and high-dimensional data, nearest neighbour-based outlier detection demands considerable resources to cope with the increase of computation complexity.

### 2.6 Others

As a key concept of ensemble leaning, isolation is introduced to outlier detection. The intrinsic characteristics of outlier are taken into full account during the construction of outlier detector. On the whole, compared to normal data, outliers are few and different. In other words, outliers are more isolated than normal data. In [31], a distributed outlier detection model based on isolation forest (iForest) is proposed. Local isolation scores of a measurement is given by local detectors. The final isolation score of a measurement is calculated by a global detector which aggregates the results of local detectors.

Drawbacks: outlier detection based on iForest is not sensitive to outliers which are close to normal data.

With the blooming development of edge computing [32], more and more researchers in the field of outlier detection turn to this novel concept. In [10], a two-part outlier detection algorithm based on autoencoder neural networks is proposed. The two parts are sensors and IoT cloud. Outliers can be detected at an individual sensor in a distributed structure without communicating with any other sensors or the cloud. The computation-intensive learning tasks are handled by the cloud. In [33], a distributed sensor data outlier detection model based on edge computing is proposed. The continuity of time series and the correlation among multi-source data are used to conduct outlier detection.

Drawbacks: currently, the offloading degree of edge computing is still an open question. There are short of appropriate solutions. Thus, there is no guarantee of effective outlier detection in practice.

In general, the deployment of WSNs is considered mature in various application scenarios. As appropriate combination of edge device and wireless sensor node is able to improve the efficiency of network in terms of message overhead and energy consumption, the terminal layer of edge computing often employs WSNs to collect data. However, there are few literatures about the combination of edge computing and outlier detection in WSNs. Hereby, we propose an innovative mobile edge–cloud collaboration outlier detection framework in WSNs.

### 3 Mobile edge–cloud collaboration outlier detection framework

#### 3.1 Traditional outlier detection model

A WSN typically consists of a large number of sensor nodes scattered over a region of interest. The network is deployed to monitor specific physical phenomena. For different application
FIGURE 1 A typical topology of WSN

variants, various kinds of topologies can be formed based on different arrangements of sensor nodes. A typical network topology is shown in Figure 1. Sensor nodes in the network are grouped into seven clusters. Clusters are separated by dashed line. Each cluster has a cluster head which is denoted by a grey point. Other nodes are denoted by black points. There is only one sink node and it is denoted by a black circle. Communication links between a cluster head and the sink node are denoted by dotted line. The sink node transmits data collected by sensor nodes to a cloud platform.

For traditional methods, the idea of outlier detection is as follows. Sensor nodes within a cluster send collected data to cluster head. The cluster head conducts outlier detection based on the received data and sends the result to the sink node. In this way, cluster head makes a considerable amount of energy consumption during the process of outlier detection. In general, the main purpose of a WSN is collecting data and forwarding data packets from event regions to a sink. As sensor node is energy constrained, the corresponding computing capacity of a sensor node is distinctly insufficient when dealing with large amount of complex data. Consequently, the energy of cluster head is expected to be drained ahead of other sensor nodes. Typically, a certain amount of node failure may lead to the compromise of network usability.

3.2 Mobile edge–cloud collaboration outlier detection

Though there is no generally accepted definition of a detailed edge computing architecture, we employ a hierarchical architecture with three layers: (1) cloud layer, (2) edge layer, and (3) WSNs. The cloud layer possesses abundant storage space and powerful computation capability. The edge layer containing mobile edge nodes mainly focuses on outlier detection and uploading data to the cloud. It is also a connector between the cloud and the WSNs. The bottom layer contains various kinds of WSNs whose purpose is collecting data.

As different WSNs may target at different phenomena, mobile edge nodes for outlier detection are also different. For each WSN, mobile edge nodes establish the corresponding detection model (i.e. detector) based on the collected normal data. The detector is expected to detect outliers in a quick way during the process of data collection. In addition, certain data are uploaded to the cloud for the update of detection model described later in this section.

3.2.1 Fast ABOD

With the development of data science, WSNs are confronted with considerable amount of high-dimensional data. Traditional methods suffer from high time complexity which significantly compromises real-time requirements. Thus, the outlier detection of high-dimensional data in WSNs is an urgent issue [34]. We introduce a FastABOD to tackle this problem. With the increase of data dimensionality, the concept of distance is becoming more and more meaningless for high-dimensional data. Thus, distance is not suitable for formulating an outlier detection model. Then, researchers turned to angular variation among data points for the purpose of conducting outlier detection. This idea is able to alleviate the impact caused by curse of dimensionality. Figure 2 is presented to facilitate the description of angle-based outlier detection.

As shown in Figure 2, we introduce an apparent outlier which is denoted by $c$. Now, we consider the cluster formed by other points in Figure 2. For each point $a$, all cases of angle $\angle xoy$ are examined, where $x \neq o$ and $y \neq o$. Points located around the cluster center tend to form angles which are of great difference (e.g., point $a$), while points located around the border are likely to form angles which are of little difference (e.g., point $b$). For outliers (e.g., point $c$), differences among the formed angles are even slighter than that of points located around the border. Thus, angle variance can be chosen as an outlier factor to evaluate anomaly degree of each point. Detailed implementation is based on a weighted cosine variance, where weights are denoted by corresponding distances among points. In specific, for point $o \in D$, the angle-based outlier factor (ABOF) could be
formulated as

\[
ABOF(\omega) = \frac{\text{VAR}_{\omega \in D} \langle \vec{a}_x, \vec{y} \rangle}{\text{dist}(a, x)^2 \cdot \text{dist}(a, y)^2},
\]  

(1)

where \(\langle \cdot, \cdot \rangle\) denotes scalar multiplication, \(\text{dist}(\cdot, \cdot)\) is Euclidean distance. It is obvious that the distance between a point and the cluster is inversely proportional to the differences among the formed angles. While the differences among the formed angles are proportional to the value of ABOF. Points in set \(D\) are sorted in ascending order based on ABOF.

However, the above ABOD method has a major flaw. For each point, all pairs of other points in the cluster have to be examined. This contributes to a time complexity of \(O(n^2)\) which is unacceptable for large volume of data. Thus, we introduce FastABOD which is based on ABOF. This algorithm is able to reduce the time complexity of ABOD. Only the pair of points with the maximum variance weight is used. For pairs of points among \(k\) nearest neighbours, a randomly selected set of \(k\) points is approximately feasible. However, the weights of nearest neighbours are maximized. Thus, nearest neighbours are most likely to accord better approximation than other points. This is particularly true for low-dimensional data where distance is more meaningful. For FastABOD, an angle-based outlier factor \(\text{apprABOF}\) which is analogous to ABOF is formulated based on the weighted cosine variance

\[
\text{apprABOF}_k(\omega) = \frac{\text{VAR}_{\omega \in N_k(\omega)} \langle \vec{a}_x, \vec{y} \rangle}{\text{dist}(a, x)^2 \cdot \text{dist}(a, y)^2},
\]  

(2)

where \(N_k(\omega)\) denotes the \(k\) nearest neighbours of point \(\omega\). This approximation only involves \(k\) nearest neighbours. Thus, the time complexity is \(O(n^2 + n \cdot k^2)\). Compared to \(O(n^3)\), there is an improvement made by an order of magnitude.

As mentioned above, the basic ABOD method possesses a time complexity of \(O(n^3)\). This complexity is hardly acceptable for large amount of data in real-world application scenarios. As the FastABOD algorithm devised in our paper possesses a time complexity of \(O(n^2 + n \cdot k^2)\), there is a significant improvement. Consequently, the corresponding computational overhead during the labelling of training data is reduced.

During the process of data collection, mobile edge nodes conduct outlier detection with FastABOD. For an individual mobile edge node, we assume that there are \(n\) wireless sensor nodes within its collection range. Data collected by these sensor nodes are denoted as \(D_1, D_2, \ldots, D_n\). And \(y_m\) denotes a data sample whose dimensionality is \(m\). Based on the analysis presented in [35], the value of threshold \(p\) is set to 0.01. When \(\text{apprABOF}(\omega) < p\), \(\omega\) is considered as an outlier and it is labelled with -1. Whereas a normal data is labelled with 1. The detailed FastABOD algorithm is described in Algorithm 1.

In a word, Algorithm 1 labels data by the angle variation of data points. The labelled data are used as training data to construct the detection model subsequently.

### 3.2.2 Support vector data description

SVDD [36] is a one-class classification approach derived from SVM. In SVDD, data are mapped to high-dimensional feature space by kernel method. This algorithm does not require the prerequisite that data are normally distributed. Therefore, SVDD possesses great advantage in dealing with non-Gaussian and non-linear data. Moreover, it can be easily applied to function fitting. This greatly facilitates the analysis of other machine learning problems.

The basic idea of SVDD is as follows. Original data are projected into high-dimensional feature space by non-linear transformation. It is expected to obtain a closed and compact circle (or hypersphere) which is able to enclose all input data or as much data as possible. The boundary of the circle is the decision boundary. It is used to distinguish outliers and normal data. Due to the performance advantage of SVDD for binary classification, SVDD is suitable for outlier detection. The construction of detection model based on SVDD is given below.

Given data set \(X = \{x_1, x_2, \ldots, x_n\}\) which contains \(n\) data points, we denote the center and radius of the target hypersphere by \(a\) and \(R\), respectively. The optimization problem based on the SVDD can be formulated as

\[
\min_R \left( R^2 + C \cdot \sum_{i=1}^{n} \xi_i \right),
\]

s.t. \(\|x_i - a\|^2 \leq R^2 + \xi_i\),

(3)

where \(\xi_i\) is a slack variable, \(C\) is a penalty factor which is used to achieve a compromise between the size of the hypersphere and the number of enclosed data samples. The geometric model of SVDD is shown in Figure 3, where data samples are represented by black points.

The optimization problem described in (3) can be solved by the Lagrange multiplier method. Thus, the Lagrange equation is

\[
\text{Algorithm 1: FastABOD}(D, p)
\]

1: for \(i = 1\) to \(n\) do
2: for \(j = 1\) to \(n\) do
3: if \(\text{apprABOF}(\omega) < p\) then
4: \(y_j = -1\)
5: else
6: \(y_j = 1\)
7: end if
8: \(D_i \leftarrow D_i \setminus \{\omega\}\)
9: \(D_i \leftarrow D_i \cap \{\{\omega, y_j\}\}\)
10: end for
11: end for
12: return \(D\)
FIGURE 3 Geometric model of SVDD

constructed as

\[ L(R, a, \alpha_i, \xi_i) = R^2 + C \cdot \sum_{i=1}^{n} \xi_i - \sum_{i=1}^{n} \alpha_i \left[ R^2 + \xi_i - (\sigma^2 - 2\alpha_i \gamma_i + \alpha_i^2) \right] - \sum_{i=1}^{n} \gamma_i \xi_i, \]

where \( \alpha_i \geq 0, \gamma_i \geq 0. \) Let the partial derivatives of \( L \) with respect to \( R, a, \xi_i \) be zero, respectively. Then, we have

\[
\begin{aligned}
\sum_{i=1}^{n} \alpha_i & = 1 \\
\sum_{i=1}^{n} \alpha_i \gamma_i & = \sum_{i=1}^{n} \alpha_i \xi_i
\end{aligned}
\]

where \( 0 \leq \alpha_i \leq C. \)

Combining (4) and (5), we have

\[ L = \sum_{i=1}^{n} \alpha_i \left( \xi_i^2 - \gamma_i^2 \right) - \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j \left( \gamma_i \gamma_j - \xi_i \xi_j \right). \]

However, this method is only applicable to data set whose input space is circular. When the input space possesses a non-circular distribution, this method is unable to guarantee satisfactory performance. Thus, kernel function is introduced to improve applicability of the method.

For an appropriate mapping \( \phi \) which maps input data sample \( x_i \) to a high-dimensional feature space \( \phi(x_i) \). A hypersphere which encloses as many points as possible should be obtained in the high-dimensional feature space. The inner product \( \langle x_i, x_j \rangle \) in (6) can be replaced by kernel function \( K(x_i, x_j) \). We choose Gaussian kernel function which has a strong locality. As this kernel function possesses good robustness and is independent of sample size, it is widely used. Moreover, the variation of polynomial parameters has mild impact on the performance of kernel function. For sensor data in our application scenario, outliers and normal data are not linearly separable. Besides, sample size is considerable and feature space is large. Under the circumstances, Gaussian kernel function is able to work well. In specific, the Gaussian kernel function can be formulated as

\[ K(x_i, x_j) = \phi(x_i) \cdot \phi(x_j) = \exp \left( -\frac{(x_i - x_j)^2}{2\sigma^2} \right). \]

In this case, (5) could be transformed into a Lagrange dual problem:

\[
\begin{aligned}
\max_{\alpha} & \quad \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j K(x_i, x_j) - 1 \\
\text{s.t.} & \quad \sum_{i=1}^{n} \alpha_i = 1
\end{aligned}
\]

where \( 0 \leq \alpha_i \leq C, j = 1, 2, ..., n. \)

For the typical quadratic optimization problem in (8), the objective set \( \alpha = (\alpha_1, \alpha_2, ..., \alpha_n) \) can be divided into three categories: (1) \( \alpha_i = 0 \) denotes normal data points fall inside the hypersphere; (2) \( 0 < \alpha_i < C \) denotes normal data points fall on the boundary of the hypersphere; (3) \( \alpha_i > C \) denotes outlier points fall outside the hypersphere. Thus, the decision function of SVDD is

\[ f(x_i) = \text{sgn} \left( \|\xi_i\|^2 - \sigma^2 - R^2 \right). \]

For (9), \( f(x_i) \leq 0 \) denotes \( x_i \) is a normal data sample, while \( f(x_i) > 0 \) indicates \( x_i \) is an outlier.

In short, the outlier detection model is constructed based on the above SVDD algorithm and the training data obtained by Algorithm 1. Data labelled by decision function \( f(x_i) \) are transmitted to the cloud. However, as training data are insufficient in most cases, the obtained detection model is often unsuitable for the whole WSN. Thus, the obtained model needs to be optimized.

As described above, the decision boundary of SVDD is given by target objects, namely support vectors. A more flexible boundary can be obtained with the employment of kernel function, by which data are mapped to a high-dimensional feature space. Nevertheless, SVDD has a major disadvantage: when there are outliers in the training data, SVDD is likely to accord loose decision boundaries. As the essential idea of SVDD is to obtain a closed and compact circle (or hypersphere) which is able to enclose all input data or as much as possible, the boundary of the circle is the decision boundary. In general, the SVDD is just able to accord one description of the target set. In that way, when there are outliers in the training data, the characteristics of outliers are also considered. As outliers are treated as normal data for the training process, the obtained decision boundary for distinguishing normal data from outliers becomes imprecise. In this case, the number of outliers which are identified as normal data will increase, namely the false negative rate of outlier detection will increase. To address this problem, we introduce fuzzy theory to the above mentioned hypersphere. Then, the basic idea can be rephrased as the finding of a minimum hypersphere with fuzzy constraints. For a hypersphere
whose center and radius are $a$ and $R$, the data description can be obtained by minimizing error function

$$ F(R, a) = R^2 + C \sum \xi_i, $$

$$ \text{s.t.} \|x_i - a\|^2 \leq R^2 + \xi_i, \forall i $$

where $C$ denotes the tradeoff between the volume of the description and the error. Note that both parameter $C$ and slack variable $\xi_i$ are unable to tolerate noisy samples. They are tuned by system without knowledge of importance of samples. However, a satisfactory data description method should cover this aspect. In general, if $A$ is fuzzy less than $B$, we have

$$ A \lesssim B \rightarrow A \approx B + d(1 - \alpha). $$

Thus, the fuzzy inequality in (10) can be transformed to

$$ \|x_i - a\|^2 \leq (R^2 + \xi_i) + d_i(1 - \alpha), \forall i. $$

In (11), $d_i$ is the weight of each data, the effect of an outlier can be decreased by reducing the uncertainty value $\alpha$. In particular, when $\alpha = 1$, this fuzzy data description degrades into the SVDD mentioned above.

Similarly, the optimization problem given in (12) can be solved by the Lagrange multiplier method. Hence, the Lagrange equation is constructed as

$$ L(R, a, \alpha, \gamma, \xi) = R^2 + C \sum \xi_i - \sum \gamma_i \xi_i - \sum \lambda_i \left( (R^2 + \xi_i) + d_i(1 - \alpha) - (\|x_i\|^2 - 2ax_i + \|d_i\|^2) \right), $$

where $\lambda_i \geq 0, \gamma_i \geq 0$. Let the partial derivatives of $L$ with respect to $R, a, \xi$ be zero, respectively. Then, we have

$$ \begin{cases} \sum_{i=1}^n \alpha_i = 1 \\ a = \sum_{i=1}^n \lambda_i \xi_i / \sum_{i=1}^n \lambda_i \\ C - \lambda_i - \gamma_i = 0 \end{cases} $$

(14)

where $0 \leq \alpha_i \leq C$. For the typical quadratic optimization problem in (15), the objective set $\lambda = (\lambda_1, \lambda_2, ..., \lambda_n)$ could be divided into three categories: (1) $\lambda_i = 0$ denotes normal data points fall inside the hypersphere; (2) $0 < \lambda_i < C$ denotes normal data points fall on the boundary of the hypersphere; (3) $\lambda_i > C$ denotes outlier points fall outside the hypersphere. Thus, the decision function of $f$-SVDD is

$$ f(x_i) = \text{sgn} \left( \|\xi(x_i) - \lambda\|^2 - R^2 \right). $$

(16)

For (16), $f(x_i) \leq 0$ denotes $x_i$ is a normal data, while $f(x_i) > 0$ indicates $x_i$ is an outlier.

### 3.2.3 Model optimization

Most traditional machine learning algorithms possess a batch processing mode. This working mode assumes that all training data are predetermined, a classifier is obtained by minimizing the empirical error defined on training data. This kind of method works well with small amount of data. The batch learning mode impedes an outlier detection model from incremental learning. Training is conducted with the entire available data. And in most cases, this is done in an off-line manner. Thus, it is inevitable that considerable amount of time and computation resources are consumed, especially when the training data is of huge volume. Moreover, the update of a batch learning system requires a brand-new training corresponding to the entire available data set, though new data might be of a small fraction. When the training is completed, a new system replaces the old one. However, for large amount of data which introduce high computation complexity, certain real-time requirements cannot be met. On the contrary, on-line learning constantly feeds new data to the current system. Thus, learning effect is gradually accumulated. For small amount of incremental data, the related training and learning are swift and cost-effective. As continual real-time streaming data, sensor data can be effectively processed by on-line learning. In our model, mobile edge nodes conduct outlier detection during a random movement. In the meantime, the outlier detection model is iteratively optimized by on-line learning.

For a mobile edge node, both data collection and outlier detection are conducted within its collection range. During the process of outlier detection, both Algorithm 1 and the decision function $f(x)$ are used. By comparing the detection results accorded by Algorithm 1 and the decision function $f(x)$, the accuracy of $f(x)$ is calculated. New data whose detection results given by $f(x)$ contradict with that of Algorithm 1 are added into training set $D$. This iterative process is repeated until the accuracy of $f(x)$ meets or exceeds a required threshold. By that time, the decision function $f(x)$ is directly used to conduct outlier detection in subsequent collection ranges. In brief, Algorithm 1 is used to classify the input data and obtain labelled training set $D$. Then, the normal data in set $D$ is used to construct the decision function $f(x)$. The underlying principle is that $f$-SVDD is expected to accord a hypersphere which encloses as many
data points as possible. If there are outliers mixed in the normal data, the accuracy of $f(x)$ might be affected. To cope with the dynamics of sensor data, the iterative method is proposed to update $f(x)$.

3.2.4 Mobile edge–cloud collaboration

Based on the building blocks described above, we present the mobile edge–cloud collaboration. In brief, the actual task of outlier detection is conducted on the edge layer, while the training and updating of the detection model are carried out on the cloud.

Each edge node is equipped with the $f$-SVDD package and performs two tasks: (1) collecting data from sensor nodes in underlying WSNs and providing input data to the cloud as training data, (2) conducting outlier detection based on the collected data.

The cloud trains the $f$-SVDD package with data provided by all sensor nodes within the same collection range. For an updated detection model given by the cloud, the information of the center and the radius of the corresponding model are sent to each edge node.

4 NUMERICAL RESULTS

Here, we evaluate the effectiveness of our proposal. All experiments are conducted on a PC with Intel Core i5-4200M 2.50 GHz, which runs Microsoft Windows 7 Professional (64-bit) with 16GB RAM. Meanwhile, Matlab R2018b is used for preparation of data, implementation of algorithms, and the analysis of experimental results.

4.1 Data set and parameters

The real data set IBRL [37] is used to evaluate the proposed model. The Intel Berkeley Research Lab (IBRL) data set is based on the publicly available Intel Lab Data consists of measurements collected from 54 sensor nodes deployed at the IBRL. Mica2Dot sensors with weather boards are used to collect timestamped topology information, temperature, humidity, light, and voltage every 31 s. The specific deployment of sensor nodes for IBRL is shown in Figure 4. The simulation parameters are shown in Table 1.

4.2 Model establishment

The classification model and detection model are evaluated with 90 normal data samples and 10 outliers from the IBRL data set. As shown in Figure 5, the classification result of Algorithm 1 is depicted with two colours, where red points and blue points denotes outliers and normal data, respectively. As shown in Figure 5a, the above 90 normal data samples and 10 outliers are depicted. Roughly speaking, the 10 red points possess a patchy distribution pattern. On the contrary, the 90 blue points are densely distributed. For a clear view of these blue points, the area specified by the dotted line rectangle in Figure 5a is enlarged and redrawn in Figure 5b.

For the $f$-SVDD package with Gaussian kernel function, the obtained detection model based on training data is shown in Figure 6. In Figure 6b, the construction of hypersphere are depicted. The values of radius for each hypersphere are labelled on the boundaries. As shown in Figure 6a, purple curves indicate the boundary of the obtained detection model. Training data are denoted by red points. Data samples that lie outside the boundary are considered as outliers. The green points denote support vectors. Under normal circumstances, they are located at the boundary. These green points are used to determine the size of a hypersphere. As our model is trained with normal data, all green points in Figure 6a are the support vectors of normal data. In addition, there still exists certain error even for a well-trained model. This corresponds to the nature of machine learning, namely the result of training cannot be in full conformity with the training data.

As shown in Figure 7, 50 data samples which contain 10 outliers are examined by the obtained detection model. Red solid line denotes the radius of hypersphere. Blue dotted line denotes the distance between a data sample and a hypersphere. The detection result shows an accuracy of 92%. This result indicates that there is still room for improvement. A probable improvement of accuracy could be achieved by the iterative optimization scheme described in Section 3.2.3.
To further investigate our model, we employ three other methods to make a comparison. The first one is a TODM. In TODM, outlier detection is carried on local sensor nodes. The detection result is sent to sink node. The second one is an angle-based outlier detection model (AODM). In AODM, mobile edge nodes perform angle-based outlier detection during movement. Specifically, the above two methods are implemented based on parts of our methods. The third one is the mobile data cleaning model (MDCM) proposed in [38]. In MDCM, mobile edge nodes are also employed for data processing. In addition, our proposal can be described as mobile outlier detection with edge and cloud. For simplicity, we call it MODEC for short. Detection delay and energy consumption of the above four methods are evaluated in terms of data dimensionality, data volume, and ratio of outliers. Detailed experimental results are shown in Figures 8-10. The illustrated results are based on the average values obtained from extensive experiments. For Figures 8-10, each figure corresponds to 150 parallel experiments. We hold the opinion that the obtained average results are more accurate and of more generality than a single trial.
As a mobile edge node is moving along a certain trajectory, each mobile edge node has a data collection range. Data from sensor nodes in the underlying WSNs which are within the data collection range are likely to be collected by the mobile sensor node. The subsequent outlier detection is conducted using these collected data. In other words, only a part of the original data generated by a sensor node in an underlying WSN is collected by a mobile edge node. This part of data is processed for outlier detection. However, for a sensor node, the generation of data is continuous. Thus, certain amount of data are not collected by the edge layer. In this case, for the data generated by sensor nodes in WSNs, we employ the setting of 45% outliers to compensate the imbalance of the data collected by mobile edge nodes. This setting is chosen based on extensive experiments. In other words, a setting of high ratio of outliers (e.g., 45%) aims to facilitate the evaluation of the performance and adaptability of our outlier detection model. Theoretically speaking, the concept of outlier is not just simply ‘few’. The overall appearance of a certain group of outliers is also not simply ‘few and different’. For some outlier detection models (e.g., isolation-based models [39]), their detection principles are only able to handle the outliers which are ‘few and different’. While here, the proposed model is not based on the idea of ‘isolation’. In fact, it is expected that our proposal is able to handle the situation where there are ‘many’ outliers. Besides, when there are ‘few’ outliers, our proposal is also able to work. In other words, for our proposal, there is no conflict for the above two cases. Thus, a setting of high ratio of outliers (e.g., 45%) is employed and it works well.

As shown in Figure 8, data dimensionality $dd = 1, 2, 3, 4$. Data volume $dv = 300$, namely the number of data samples collected by each sensor node. The ratio of outliers $dar = 45\%$. For detection delay (Figure 8a), when data dimensionality is small (e.g., $dd = 1, 2$), the performance of TODM is the best. For data with low dimensionality, the computing capability of ordinary local sensor nodes is sufficient to meet the detection demand. On the contrary, when data dimensionality is high (e.g., $dd = 3, 4$), the performance of TODM is the worst. Namely, for high-dimensional data, ordinary local sensor nodes are unable to meet the corresponding detection demand. As mobile edge nodes are introduced to the other three sensor nodes, detection delay gets decreased compared to that of TODM. Furthermore, our method MODEC accords the best performance when data dimensionality is increased. Compared to AODM,
MODEC directly uses detection model based on SVDD to conduct outlier detection, the training time of this model is much less than that of AODM. For MDCM, the time complexity of FastABOD algorithm in MODEC is smaller than that of traditional ABOD algorithm. For energy consumption (Figure 8b), TODM accords the worst performance. As the burden of outlier detection is laid on ordinary local sensor nodes, considerable amount of energy is consumed. For MODEC, most computational tasks related to outlier detection are carried out on mobile edge nodes. Thus, energy consumption of sensor nodes in WSNs is significantly reduced.

As shown in Figure 9, data dimensionality $d_d = 1, 2, 3$. All 54 sensor nodes in the IBRL data set are used. The ratio of outliers $d_ar = 45\%$. With the increase of data volume, the detection delay and energy consumption of the four models increase steadily. The ranking of the four models in terms of delay by descending order is TODM, AODM, MDCM, and MODEC. The same ranking happens in terms of energy consumption. As TODM and AODM use onefold detection algorithm to conduct outlier detection, the detection models of both TODM and AODM are of large scale. MDCM and MODEC are superior to TODM and AODM, especially when the data volume becomes large. In a nutshell, this phenomenon is mainly due to the composite detection models of MDCM and MODEC. The more the sensor data collected are, the better the performance is. Besides, MODEC is superior to MDCM in terms of both delay and energy consumption. This happens because the model constructed based on SVDD algorithm in MODEC is able to accord better detection results than that of MDCM.

As shown in Figure 10, data dimensionality $d_d = 1, 2, 3$. Data volume $d_v = 300$, namely the number of data samples collected by each sensor node. The amount of outliers is manually changed within the range of $[35\%, 55\%]$. For detection delay (Figure 10a), with the increase of the ratio of outliers, both TODM and AODM exhibit a mild decrease. When the amount of outliers is increased, the set of training data becomes more balanced. Consequently, the scale of detection model constructed based on normal data is reduced. Thus, both MDCM and MODEC show a remarkable decrease. For energy consumption (Figure 10b), TODM, AODM, and MODEC are insensitive to variation of the ratio of outliers. As the above three methods upload data to the cloud, the increase of outliers has slight impact on the energy consumption. As MDCM is a data cleaning model, the detected outliers are directly removed. The remaining normal data are uploaded to the cloud. Thus, the amount of energy consumption for MDCM is the smallest.

Besides the above analysis of energy consumption illustrated in Figures 8–10. We also conducted extensive experiments to investigate the lifetime of the whole WSN. For the above four methods, the percentages of operational nodes are depicted in Figure 11.

The experiments that correspond to Figure 11 adopt the following parameter setting: data dimensionality $d_d = 3$, data volume $d_v = 300$, and the ratio of outliers $d_r \in \{35\%, 40\%, 45\%, 50\%, 55\%\}$. In specific, 100 parallel experiments are conducted, namely 20 times for each of the 5 $d_r$ values. As shown in Figure 11, the percentages of operational nodes for the four methods decrease over time. For $t \in [0, 1500]$, the trend of decrease is moderate. While $t \in [1500, 4000]$, the four curves decrease more sharply than that of $t \in [0, 1500]$ over time. Our proposal MODEC accords the best performance. The percentage of operational nodes falls below $60\%$ around $k_4(3940, 0.6)$. For the other three methods TODM, AODM, and MDCM, the $60\%$ points are $k_1(2320, 0.6)$,
$k_2(2715, 0.6)$, and $k_3(3450, 0.6)$, respectively. When the percentage of operational nodes is 60%, our proposal prolongs the network lifetime by 14.2% compared to MDCM. For TODM, the improvement is 69.8%.

5 CONCLUSION AND FUTURE WORK

Here, we proposed a mobile edge–cloud collaboration outlier detection framework in WSNs. The main features of our proposal include: the employment of mobile edge node above WSNs avoids complex operations of sensor nodes and improves the performance of outlier detection. Moreover, the decrease of energy consumption is beneficial for prolonging the lifetime of sensor nodes and the whole WSN. The FastABOD algorithm is developed for multi-dimensional data to obtain training data. The $f$-SVDD algorithm and Gaussian kernel function are used to construct the detection model. To improve the accuracy of outlier detection, the detection model is optimized with on-line machine learning techniques. Simulation experiments on real data set IBRL show that our proposal is able to improve the accuracy of outlier detection, reduce detection delay, and energy consumption. Compared to existing methods, our model possesses obvious advantages in terms of delay, energy consumption, and network lifetime. Thus, it is efficient in outlier detection of high-dimensional data in WSNs.

However, there is still room for improvement. To regulate the movement of edge nodes, a unified edge layer management scheme should be developed. In this scheme, the strategy of movement (e.g., random walk) for edge nodes should be self-adaptive to the dynamics of underlying WSNs. In addition, the edge layer still possesses great potential in computing power. In the future, the algorithm package of outlier detection should be rebuilt to further improve the accuracy of outlier detection and reduce the computation complexity.

ACKNOWLEDGEMENTS

This work is partly supported by the Science and Technology Project of Shaanxi Province, China (Grant No. 2019ZDLGY07-08), the International Science and Technology Cooperation Program of the Science and Technology Department of Shaanxi Province, China (Grant No. 2018KW-049), the Communication Soft Science Program of Ministry of Industry and Information Technology, China (Grant No. 2019-R-29), the Special Scientific Research Program of Education Department of Shaanxi Province, China (Grant No. 17JK0711), and the Special Funds for Construction of Key Disciplines in Universities of Shaanxi Province.

The authors would like to thank the anonymous reviewers whose comments and suggestions greatly helped to improve the quality and presentation of this paper.

ORCID

Cong Gao https://orcid.org/0000-0001-9829-8025
Guohao Song https://orcid.org/0000-0002-1899-6518
Zhongmin Wang https://orcid.org/0000-0003-3731-4049
Yanping Chen https://orcid.org/0000-0002-7812-4199
REFERENCES

1. Kiani, F.: Animal behavior management by energy-efficient wireless sensor networks. Comp. Electron. Agriculture 151, 478–484 (2018)
2. Kumar, P., et al.: A certificateless aggregate signature scheme for healthcare wireless sensor network. Sustain. Comput.: Inform. Syst. 18, 80–89 (2018)
3. AL-Mousawi, A.J., AL-Hassani, H.K.: A survey in wireless sensor network for explosives detection. Comp. Elect. Eng. 72, 682–701 (2017)
4. Lee, J., et al.: FlexiCast: Energy-efficient software integrity check to build secure industrial wireless active sensor networks. IEEE Trans. Ind. Inform. 12(1), 6–14 (2016)
5. Jaiswal, K., et al.: An energy-efficient optimal multi-path routing protocol to improve QOS in wireless sensor network for IOT applications. Wirel. Pers. Commun. 111, 2493–2515 (2019)
6. Filho, P.R.G., et al.: Residi: Towards a smarter smart home system for decision-making using wireless sensors and actuators. Comput. Netw. 135, 54–69 (2018)
7. Ayadi, A., et al.: Outlier detection approaches for wireless sensor networks: A survey. Comput. Netw. 129, 319–333 (2017)
8. Yang, Z., et al.: An online outlier detection technique for wireless sensor networks using unsupervised quarter-sphere support vector machine. In: International Conference on Intelligent Sensors, Sensor Networks and Information Processing, 2008 (ISSNIP 2008), pp. 151–156, IEEE, New York (2008)
9. Chandola, V., et al.: Outlier detection: A survey. ACM Comput. Surveys 14, 15 (2007)
10. Luo, T., Nagarajany, S.G.: Distributed anomaly detection using autoencoder neural networks in WSN for IOT. In: 2018 IEEE International Conference on Communications (ICC), pp. 1-6. IEEE, New York (2018)
11. Zhang, Y., et al.: Outlier detection techniques for wireless sensor networks: A survey. IEEE Commun. Surveys Tut. 12(2), 159–170 (2010)
12. Ferrari, P., et al.: Performance evaluation of full-cloud and edge-cloud architectures for industrial IOT anomaly detection based on deep learning. In: 2019 II Workshop on Metrology for Industry 4.0 and IOT (MetroInd4.0&IOT), pp. 420–425. IEEE, New York (2019)
13. Sohbanayak, S., et al.: Container-based task scheduling for edge computing in IOT-cloud environment using improved HBF optimisation algorithm. Int. J. Embedded Syst. 13(1), 85–100 (2020)
14. Samir, A., Pahl, C.: Anomaly detection and analysis for clustered Cloud Computing, GRIDs, and Virtualization, pp. 110–119. IARIA (2019)
15. Feng, H., et al.: Distributed outlier detection algorithm based on credibility feedback in wireless sensor networks. IET Commun. 11(8), 1291–1296 (2017)
16. Huo, W., et al.: AnomalyDetect: An online distance-based anomaly detection algorithm. In: International Conference on Web Servers, pp. 63–79. Springer, Berlin (2019)
17. Yao, Y., et al.: Online anomaly detection for sensor systems: A simple and efficient approach. Perform. Eval. 67(11), 1059–1075 (2010)
18. Andrade, A.T.C., et al.: k outlier detection using k-means clustering and lightweight methods for wireless sensor networks. In: IECON 2016-42nd Annual Conference of the IEEE Industrial Electronics Society, pp. 4683–4688. IEEE, New York (2016)
19. Rajasegarar, S., et al.: Hyperspherical cluster based distributed anomaly detection in wireless sensor networks. J. Parallel Distrib. Comput. 74(1), 1833–1847 (2016)
20. Abdal, A., et al.: Outlier detection for wireless sensor networks using density-based clustering approach. IET Wirel. Sensor Syst. 7(4), 83–90 (2017)
21. Gao, J., et al.: On threshold-free error detection for industrial wireless sensor networks. IEEE Trans. Ind. Inform. 14(5), 2199–2209 (2018)
22. Zamry, N.M., et al.: Unsupervised anomaly detection for unlabelled wireless sensor networks data. Int. J. Advances Soft Comput. Appl. 10(2), 172–191 (2018)
23. Feng, Z., et al.: A new approach of anomaly detection in wireless sensor networks using support vector data description. Int. J. Distrib. Sensor Netw. 13(1), 1550417716686161 (2017)
24. Titouna, C., et al.: Outlier detection approach using Bayes classifiers in wireless sensor networks. Wirel. Personal Commun. 85(3), 1009–1023 (2015)
25. Rassam, M.A., et al.: A distributed anomaly detection model for wireless sensor networks based on the one-class principal component classifier. Int. J. Sensor Netw. 27(3), 200–214 (2018)
26. Gil, P., et al.: Outliers detection methods in wireless sensor networks. Artif. Intell. Rev. 52, 2411–2436 (2019)
27. Xu, X., et al.: A LOF-based method for abnormal segment detection in machinery condition monitoring. In: 2018 Prognostics and System and Health Management Conference (PHM-Chongqing), pp. 125–128. IEEE, New York (2018)
28. Xie, M., et al.: Scalable hypergrid k-nn-based online anomaly detection in wireless sensor networks. IEEE Trans. Parallel Distrib. Syst. 24(8), 1661–1670 (2013)
29. Breunig, M.M., et al.: LOF: Identifying density-based local outliers. In: Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, pp. 93–104. ACM, New York (2000)
30. Xu, S., et al.: Support vector machines based on k nearest neighbor algorithm for outlier detection in WSNs. In: 2012 8th International Conference on Wireless Communication, Network, and Mobile Computing, pp. 1–4. IEEE, New York (2012)
31. Ding, Z.G., et al.: An isolation principle based distributed anomaly detection method in wireless sensor networks. Intern. J. Automat. Comput. 12(4), 402–412 (2015)
32. Shi, W., et al.: Edge computing: Vision and challenges. IEEE Internet Things J. 5(3), 637–646 (2016)
33. Qi, Z., et al.: Edge computing application: real-time anomaly detection algorithm for sensing data. J. Comput. Res. Develop. 55(3), 524 (2018)
34. Chen, Y., et al.: Fast density peak clustering for large scale data based on knn. Knowl. Based Syst. 187, 104824 (2020)
35. Pham, N., Pagh, R.: A near-linear time approximation algorithm for angle-based outlier detection in high-dimensional data. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 877–885. ACM, New York (2012)
36. Tax, D.M.J., Duin R.P.W.: Support vector domain description. Pattern Recognit. Lett. 20(11/13), 1191–1199 (1999)
37. Bodik, P., et al.: Intel lab data. Online dataset (2004). http://db.csail.mit.edu/labdata/labdata.html
38. Wang, T., et al.: Big data cleaning based on mobile edge computing in industrial sensor-cloud. IEEE Trans. Ind. Inform. 16(2), 1321–1329 (2019)
39. Wang, Z.M., et al.: An isolation-based distributed outlier detection framework using nearest neighbor ensembles for wireless sensor networks. IEEE Access 7, 96319–96333 (2019)