Hindawi
International Journal of Distributed Sensor Networks
Volume 2024, Article ID 5536615, 14 pages
https://doi.org/10.1155/2024/5536615

Research Article

Intrusion Detection Model for Wireless Sensor Networks Based on FedAvg and XGBoost Algorithm

Hongjiao Wu

Henan Vocational College of Tuina, Luoyang 471000, China

Correspondence should be addressed to Hongjiao Wu; whjrpf@163.com

Received 25 January 2023; Revised 3 March 2024; Accepted 8 March 2024; Published 25 March 2024

Academic Editor: Yanjiao Chen

For the characteristics of channel instability in wireless sensor networks, this paper proposes an intrusion detection algorithm based on FedAvg (federated averaging) and XGBoost (extreme gradient boosting) wireless sensor networks using fog computing architecture. First, the network edge is extended by introducing fog computing nodes to reduce the communication delay. It reduces the transmission bandwidth and privacy leakage risk while improving the accuracy of jointly learned global and local models. Then, the histogram-based approximation calculation method is improved to adapt to the unbalanced data characteristics of wireless sensor networks. Finally, by introducing TOP-K gradient selection, the number of model parameter uploads is minimized, and the efficiency of model parameter interaction is improved. The experimental results show that this algorithm has superior detection performance and low energy consumption. It is also compared with other algorithms to demonstrate the high detection rate and low computational complexity of this algorithm.

1. Introduction

The inherent characteristics of wireless environment make the information traffic of wireless sensor networks (WSNs) more vulnerable to attacks such as counterfeiting, stealing, forgery, and tampering. Timely detection of various network attacks hidden in the normal traffic is important for secure and reliable information transmission. Wireless sensor networks are special self-organizing networks that implement functions such as data sensing, processing, and control [1]. Wireless sensor network is a highly interdisciplinary research field, which has a very broad research prospect and can be widely used in battlefield reconnaissance, agricultural industrial measurement and control, electronic medical health and smart home, etc. [2, 3]. However, with the diversification of network attacks and the complexity of node deployment environments, it is of great theoretical and practical importance to study an intrusion detection method with low energy consumption and efficient detection of multiple attacks [4, 5]. With the development of communication network technology, Internet technology and its applications are becoming increasingly mature [6, 7]. However, the need for data security and privacy in the Internet poses unprecedented challenges for traditional Internet architectures [8]. Blockchain technology, artificial intelligence, as a promising new technology, has been applied to the Internet system [9, 10].

With the increased emphasis on WSNs, many national research organizations have joined in doing extensive and fruitful research on intrusion detection in WSNs [11, 12]. Intrusion detection methods, while making notable progress, exhibit certain limitations that impede their comprehensive effectiveness. One prominent issue lies in the asymmetrical treatment of known and unknown types of attacks. Existing systems often excel in detecting familiar attack patterns, showcasing a high detection rate in such scenarios. However, when confronted with novel or unknown attack types, their efficacy diminishes significantly. This limitation is rooted in the reliance on predefined signatures and patterns, rendering these systems less adaptive to emerging threats with unconventional characteristics.

Moreover, the practical implementation of intrusion detection schemes grapples with the challenges posed by the complex and dynamic nature of real-world application

scenarios. The intricacies of diverse network environments, varying traffic patterns, and evolving attack strategies contribute to a substantial false alarm rate [13, 14]. The high false alarm rate not only hampers the efficiency of intrusion detection but also poses a significant operational challenge, demanding a balance between sensitivity to potential threats and minimizing unnecessary disruptions to normal network activities.

Furthermore, the deployment of intrusion detection systems in resource-constrained environments, such as wireless sensor networks, poses unique challenges. These challenges include limitations in computational power, memory, and energy resources. Conventional intrusion detection methods may not be optimized to operate efficiently within these constraints, leading to compromises in detection accuracy and system responsiveness.

Considering the above constraints and challenges in wireless sensor networks, this paper proposes a new distributed joint intrusion detection algorithm based on FedAvg and XGBoost algorithms. The proposed FedAvg algorithm presents a promising avenue for enhancing intrusion detection in wireless sensor networks. By amalgamating local stochastic gradient descent on individual clients with a server performing model averaging, the FedAvg algorithm achieves a more stable classification performance, particularly on non-IID (nonidentically distributed) data. Significantly, it manages to curtail the number of communication rounds required, showcasing a substantial improvement (up to a factor of 10-100) compared to classical synchronous stochastic gradient descent algorithms. Moreover, the adoption of a distributed learning method grounded in fog computing introduces agility and responsiveness into the system. This approach enables faster responses, including swift detection alarms, and substantially improves the overall effectiveness of intrusion detection in scenarios typical of wireless sensor networks.

The main innovations of this paper are as follows:

(1) In this paper, we use a federated averaging (FedAvg) learning algorithm to aggregate the data of different users in different regions by fog nodes to break the original data resource silos. In this process, the nodes do not need to transmit the original data but only need to transmit fewer model parameters to collaboratively train the model. This not only ensures data privacy but also is suitable for the limited and unstable bandwidth of wireless networks

(2) This paper uses a histogram-based approximation algorithm to efficiently select the optimal features, thus reducing the computational pressure on the communication of fog nodes

(3) The gradients of the extreme gradient boosting (XGBoost) tree are updated in combination with the Top-K gradient selection method. In each training, the user calculates the model parameters and sorts them according to the magnitude of their differences from the server model parameters in absolute value. The K gradient values with the smallest difference are selected, and the server aggregates these K gradient values for spanning tree and model prediction. This minimizes the number of model parameter uploads and improves the efficiency of model parameter interaction

This paper consists of five main parts: the first part is the Introduction, the second part is the State of the Art, the third part is the Methodology, the fourth part is the Result Analysis and Discussion, and the fifth part is the Conclusion.

2. State of the Art

2.1. Research on Intrusion Detection Methods. Intrusion detection methods play a pivotal role in safeguarding network security, and extensive research has been conducted by scholars worldwide in this domain.

The literature [15] proposes a hierarchical blockchain-based federated learning framework for secure and privacy-preserving collaborative IoT intrusion detection. Various malicious activities can be detected while preserving data privacy, and smart contracts can also be developed. The literature [16] designs a blockchain-enhanced federated learning marketplace that makes data from computationally constrained devices available for training in social IoT. The amount of training data is maximized for a given budget of the federated learning task. Decentralize the FL marketplace with blockchain. It can significantly improve the overall utility of the requester and the average accuracy of the model. The literature [17] proposes a positive-negative partitioning mechanism that perturbs local model parameters before aggregation. This method can reduce the error in model testing accuracy with small privacy budget and number of users.

Literature [18] introduces a rough set approach to improve the artificial immunity-based intrusion detection model by organically combining anomaly detection and misuse detection and proposes an intrusion detection method. The method can achieve vaccine injection without terminating the intrusion detection behavior, effectively reducing the length of the detector and improving the detection speed. However, the immune-based intrusion detection method has the problem of low detection rate at the early stage of training. Literature [19] proposes a K-means algorithm to solve the problem of predetermined number of clusters and the algorithm falling into local optimum. The final number of detectors included in the algorithm is the number of clustering centers. In literature [20], the adaptive AP algorithm is combined with the clustering algorithm, and the adaptive AP clustering algorithm is proposed and applied to intrusion detection. The algorithm only clusters the baseline data and the samples that are far from the cluster center, and the rest of the sample data are directly correlated, which reduces the number of samples for clustering and the time for clustering. Data compression improves the efficiency of clustering but inevitably causes a decrease in accuracy. In literature [21], an improved algorithm is proposed to address the shortcomings of the traditional fuzzy C-mean clustering algorithm. The algorithm uses the Mercer kernel to define

the objective function of the optimized fuzzy C-mean algorithm to improve the merit-seeking ability of the fuzzy C-mean algorithm and uses the Lagrange multiplier method to calculate the clustering center and the affiliation matrix separately to improve the convergence speed of the algorithm. However, the algorithm does not address the effects of unbalanced clustering and noise points on the clustering results.

Literature [22] proposes an intrusion detection scheme based on semisupervised learning and information gain rate for the problem of low detection rate of unknown types of attacks by existing intrusion detection methods. In literature [23], a hybrid multilevel intrusion detection model is proposed to address the problem that the detection rates of Probe (probing), U2R (user to root), and R2L (remote to local) are relatively low. Experiments show that this method can improve the detection of network attacks such as Probe, U2R, and R2L. Literature [24] proposes a multiattribute trust method based on fuzzy logic, which uses multiple parameters as trust metrics and uses fuzzy computation theory to calculate the final trust value of each node. Literature [25] uses a noncooperative non-zero-sum game model and a Nash equilibrium approach in order to detect attacks and reduce communication overhead. Literature [26] uses the negative selection algorithm in immune theory to achieve the detection of attacking nodes. These algorithms balance the issues of detection rate and resource cost for wireless sensor network systems. However, these methods only consider the analysis under a specific model, so the applicability of the model is more limited.

In recent years, machine learning algorithms have been gradually applied to intrusion detection and have achieved better results [27]. Most of the traditional machine learning methods are aimed at terrestrial wired networks and mostly use centralized learning methods to aggregate traffic data for centralized processing and analysis. Literature [28] proposes a centralized intrusion detection method based on deep neural network and K-nearest neighbor algorithm. It is evaluated on a public data set and has a high detection accuracy. Literature [29] proposes a hierarchical intrusion detection method based on extreme learning machine, which clusters nodes according to the functions of wireless sensor networks, improves the intrusion detection accuracy, and reduces the detection time. However, the false alarm rate of this algorithm is high in the case of small percentage of attack traffic and unbalanced traffic categories in the network. In literature [30], a centralized intrusion detection algorithm based on recurrent neural network algorithm is proposed, which has better intrusion detection capability in software-defined network environment. In literature [31], a K-means clustering algorithm is used to train and classify a large amount of data for the purpose of detecting attacks. Literature [32] uses genetic algorithm—Levenberg-Marquardt back propagation method as a WSN intrusion detection mechanism. These machine learning methods are characterized by low false detection rate and moderate complexity, which can effectively extract useful features from the data and classify the data with high accuracy. However, both machine learning methods and deep learning require higher computing power of nodes. This undoubtedly increases the burden of infinite sensor networks and reduces the network life cycle, so the model still needs to be enhanced.

2.2. Wireless Sensor Network Architecture. As shown in Figure 1(a), a wireless sensor node is a miniature electronic device consisting of a processor, a memory unit, a transceiver module, one or more sensors, an analog-to-digital converter, and a power source (usually a battery). The node uses its sensors to measure fluctuations in current conditions in the adjacent environment. These measurements are converted from analog to electrical signals by an analog-to-digital converter unit, and the information is then processed by the node's processor. Through a transceiver, the node can transmit the data generated by its processor wirelessly to other nodes, sink node, or database [33], as shown in Figure 1(b).

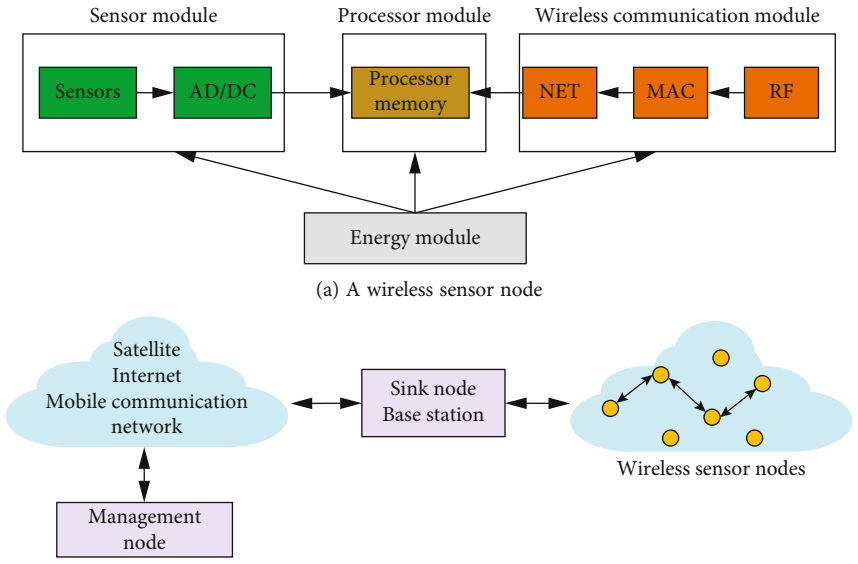
The base station uses the data transmitted to itself to both monitor the wireless sensor network to which it belongs and to transmit relevant information to human users or other networks (as shown in Figure 2).

2.3. Common Attacks in Wireless Sensor Networks. For the diversification of attacks in wireless sensor networks and information security models, in order to facilitate the analysis of security attacks, this section provides a specific classification of attacks.

(1) Based on the behavioral initiation of attackers, they can be classified as internal attacks and external attacks [34]. External attacks are mainly initiated by external attackers and the external physical environment. Its purpose is to physically damage the nodes in the network or impersonate the base station to prevent the communication transmission among the network members. Internal attacks are mainly initiated by malicious nodes in the network, which may be nodes captured by external attackers or nodes that choose to behave selfishly due to the need to reduce energy consumption

(2) Based on the protocol level of wireless sensor networks, intrusion attacks can be classified as physical layer attacks, data link layer attacks, network layer attacks, transport layer attacks, and application layer attacks [35–38]. The wireless sensor network intrusion attacks are categorized and organized as shown in Figure 3

2.4. Wireless Sensor Network Intrusion Detection Model. Intrusion detection is the process of detecting computer networks and systems to find events that violate security policies [39]. For the data that needs to be analyzed, we collectively call it an event. Intrusion detection is the analysis and processing of events [40]. A typical intrusion detection system should include at least three functional modules: (1) an information source that provides a stream of event records, (2) an analysis engine that finds signs of intrusion, and (3) a response component based on the analysis engine. The intrusion detection system consists of three main



(a) A wireless sensor node

(b) Wireless sensor network structure

Figure 1: Schematic diagram of wireless sensor network structure.

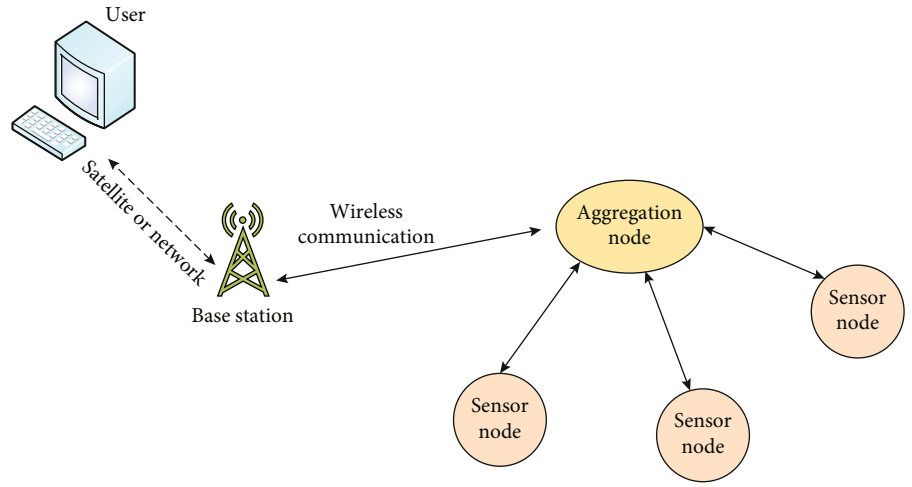


Figure 2: Typical architecture of a wireless sensor network.

components (event generators, event analyzer, and response module), and its generic model is shown in Figure 4.

3. Methodology

The model in this paper adopts a 3-tier network architecture for fog computing. The top layer is the cloud server, which has the strongest computing and storage capacity. The middle layer is the fog layer, where sensors, servers, or base stations with less computing power are used as fog nodes to share the computing and storage tasks for the cloud. The bottom layer is the terminal device.

In this paper, we use joint learning algorithm; fog nodes aggregate data from different users in different regions, breaking the original data resource silos. The nodes do not need to transmit the original data but only need to transmit a small number of model parameters to collaborate in training the model. This not only ensures data privacy but also fits into the limited and unstable bandwidth of wireless networks. Assume that there are a total of K nodes (or clients) and the z-th node stores a local dataset of $\mathscr{D}_z$, size $|\mathscr{D}_z|$. The dataset consists of input-output pairs $\{i_x, j_x\}_{x=1}^{t}$, where the input sample vector with d features is $i_x \in \mathrm{R}^d$, and the label of the input sample is $j_x \in \mathrm{R}$.

As shown in Figure 5, the training process of the joint intrusion detection algorithm consists of the following 3 steps:

Step 1. Task initialization.

The server specifies the global model and the hyperparameters of the training process, such as the learning rate, the maximum depth of the tree, and the minimum leaf node sample weights. The server broadcasts the initialized global model parameters M_{A}^{n}, the server-side data volume D, and the sampling rate ε_s to the selected participants.

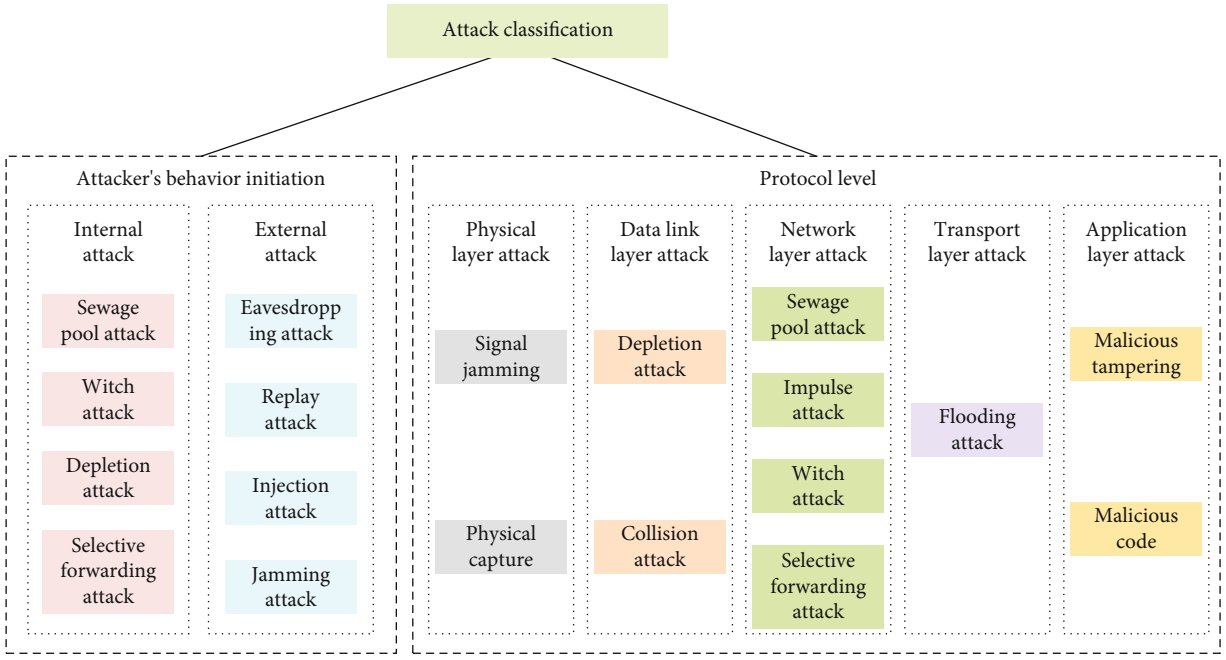


Figure 3: Classification of wireless sensor network intrusion attacks.

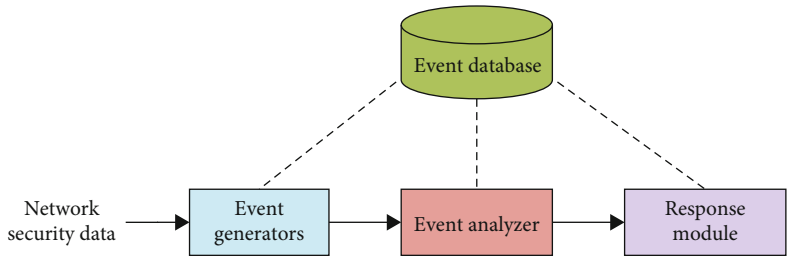


Figure 4: WSN intrusion detection model.

These hyperparameters, such as the learning rate and the maximum depth of the decision tree, are pivotal in determining the algorithm's effectiveness and experimental performance. The learning rate, influencing the step size during training, requires careful experimentation to find the right balance. Furthermore, the decision tree's maximum depth significantly impacts the model's complexity and generalization ability. Evaluating the algorithm's performance across varying tree depths illuminates the inherent trade-off between model complexity and generalization ability.

Step 2. Local model training and updating.

Based on the global model parameters M_A^n, Z fog nodes with high computational power are randomly selected according to the set ratio. Each fog node uses a specific sampling rate to segment the local data separately. The local model parameters are updated iteratively using the improved extreme gradient boosting (XGBoost) algorithm, and the final model parameters are noted as a_z, b_z.

Step 3. Global model aggregation and update.

All Z clients share and collaborate to train a global prediction model.

The server averages the client model parameters to obtain the new global model.

$$\begin{aligned} a_A^{n+1} &= a_A^n + \sum_{z=1}^{Z} \frac{|\mathscr{D}_z|}{T} a_z^{n+1}, \\ b_A^{n+1} &= b_A^n + \sum_{z=1}^{Z} \frac{|\mathscr{D}_z|}{T} b_z^{n+1}. \end{aligned} \tag{1}$$

where $T = \sum_{z=1}^{Z} |\mathscr{D}_z|$ is the total number of data samples from Z clients; a_A^n is the global aggregation parameter; b_A^n is the global update parameter; z denotes the client (i.e., fog node), $z = 1, 2, \cdots, Z$. The updated global model parameters are sent back to the data owner.

Since the limited data owned by any party can easily fall into a local optimum, the global model aggregation and updating process of the above Fed-XGB algorithm can optimize the local model parameters using the models learned by other participants. This can help the participants to get rid of local preferences and obtain more accurate models.

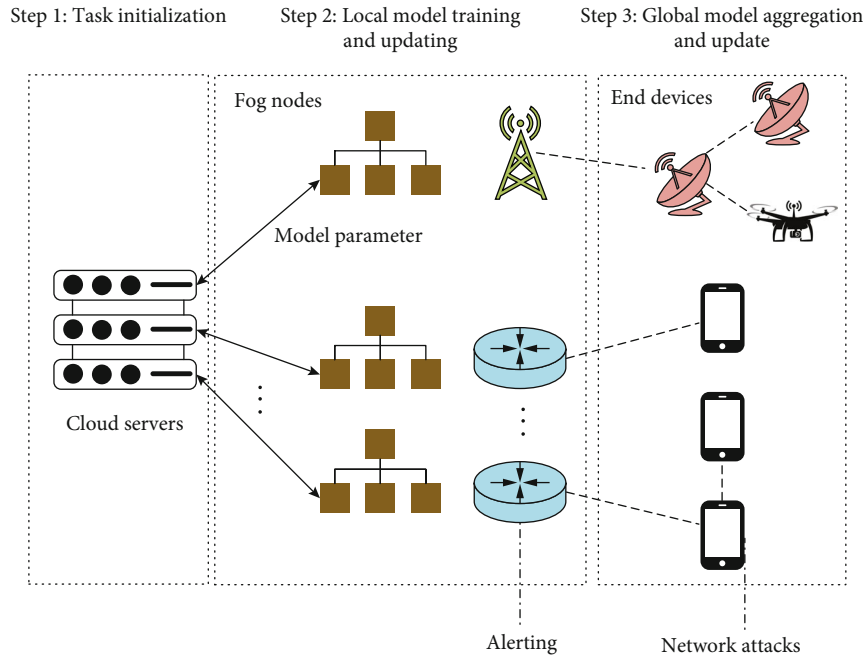


Figure 5: The intrusion detection framework based on FedAvg and XGBoost.

In this paper, in order to avoid transmitting irrelevant local updates, each client needs to know the trend of cooptimization in the global aggregation. In each learning iteration, clients should compare their local updates with the global updates to determine whether their updates are relevant. The optimization goal of the algorithm in this paper is to minimize the cumulative number of communications while ensuring the convergence of the learning algorithm, that is

$$\text{minimize} \sum_{n=1}^{N} C_n \text{s.t.} \lim_{\mathrm{N}\longrightarrow\infty} \frac{1}{N}\sum_{n=1}^{N} |f(i_n) - f(i^*)| = 0, \tag{2}$$

where C_n is the number of clients that upload local updates to the server in the n-th iteration; $f(i_n)$ is the model learned in the n-th iteration; $f(i^*)$ is the optimal model.

In this paper, we combine the Top-K gradient selection method and CMFL algorithm for gradient update selection. Specifically, in each training, the user calculates the model parameters a_x and b_x and selects the Z gradient values with the smallest difference according to the absolute value of the difference between them and the server model parameters. The server will aggregate these Z gradient values for spanning tree and model prediction.

In the context of the y-th global model update, where $M_y = \{b_1, b_2, \cdots, b_T\}$ represents the local model parameter update and $\bar{M}_y$ denotes the global model parameter, the correlation measure is computed as the percentage of identical symbolic parameters.

$$e\left(M, \bar{M}\right) = \frac{1}{T}\sum_{x=1}^{T} X\left(\text{sign}\left(M_y\right) = \text{sign}\left(\bar{M}_y\right)\right). \tag{3}$$

If M_y and $\bar{M}_y$ have the same sign, then $X(\text{sign}(M_y) = \text{sign}(\bar{M}_y)) = 1$; otherwise, it is 0. The updated relevance measure $e(M, \bar{M})$ is ranked, and then, the highest-ranked Z values are selected and uploaded for server-side model aggregation, which is used for tree generation and model prediction. In this way, the aggregation mechanism can speed up the model convergence and effectively prevent the uploading of parameters that are detrimental to the overall model, reduce C_n, and lower the communication overhead.

In this section, the XGBoost algorithm is improved by introducing a cost-sensitive function for unbalanced data samples on different fog nodes. The XGBoost algorithm is built upon the CART regression tree model [41]. Given a dataset $D = (i_x, j_x)$ with t samples and m features, the CART regression tree assigns the input sample features to each leaf node. Its prediction function is

$$\widehat{j}_x^{(n)} = \sum_{q=1}^{Q} f_q(i_x) = \widehat{j}_x^{(n-1)} + f_{\mathrm{q}}(i_x), \tag{4}$$

where $f(i) = m_y(i)$, $m_y(i)$ is the weight of leaf node y; $f_q(i)$ is the fifth regression tree. $f(i)$ is the update parameter of the model. The learning process of the XGBoost algorithm is to optimize the objective function by adding the f_q function to reduce the error between the prediction result and the actual result. The defined objective function is as follows.

$$\begin{aligned} \text{Obj}^{(n)} &= \sum_{x=1}^{t} l\left(j_x, \widehat{j}_x^{(n-1)} + f_q(i_x)\right) + \Omega\left(f_q\right) \\ &= \sum_{x=1}^{t} \left(a_x f_q(i_x) + \frac{1}{2} b_x f_q^2(i_x)\right) + \Omega\left(f_q\right), \end{aligned} \tag{5}$$

where $\Omega(f_q)$ is the regularization term, which is used to prevent overfitting of the model. $l(\cdot)$ is the loss function, which is used to measure the error between the predicted value and the true value.

$$\Omega\left(f_q\right) = \gamma N_n + \lambda \frac{1}{2}\sum_{y=1}^{N} m_y^2, \tag{6}$$

where γ is the weight factor; N_n is the number of leaf nodes occupied; λ is the regularization factor.

The traffic data categories collected by each fog node are unbalanced, among which the abnormal traffic of the attack class belongs to small samples. In order to improve the intrusion detection model to pay attention to small samples, the cost sensitivity function in each fog node is designed as follows.

$$v(c_x) = \frac{\mathrm{sum}(c_x)}{\mathrm{num}(c_x)}, \tag{7}$$

where $\mathrm{sum}(c_x)$ is the number of samples of all categories; $\mathrm{num}(c_x)$ is the number of samples of category c_x. The cost-sensitive function $v(c_x)$ in equation (7) is incorporated into the optimization process of the objective function $\mathrm{Obj}^{(n)}$ in equation (5). Specifically, the coefficients of the loss functions a_x and b_x coefficients in equation (5) are updated according to the following equation (8), where a_{c_x} and b_{c_x} are the first-order and second-order derivatives of the Taylor expansion of the loss function, respectively.

$$\begin{aligned} a_{c_x} &= v(c_x)\frac{\partial l\left(j_x, \widehat{j}_x^{(n-1)}\right)}{\partial \widehat{j}_x^{(n-1)}}, \\ b_{c_x} &= v(c_x)\frac{\partial^2 l\left(j_x, \widehat{j}_x^{(n-1)}\right)}{\partial^2 \widehat{j}_x^{(n-1)}}. \end{aligned} \tag{8}$$

The loss function is expanded by Taylor as follows.

$$\begin{aligned} \mathrm{Obj}_{c_x}^{(n)} &= \sum_{x=1}^{t} l\left(j_x, j_x^{(n-1)} + f_q(i_x)\right) + \Omega\left(f_q\right) \\ &= \sum_{x=1}^{t}\left(a_{c_x} f_q(i_x) + \frac{1}{2} b_{c_x} f_q^2(i_x)\right) + \Omega\left(f_q\right). \end{aligned} \tag{9}$$

The optimal fraction of leaf nodes is obtained by solving

$$m_y^* = -\frac{\sum_{x\in X_y} a_{c_x}}{\sum_{x\in X_y} b_{c_x} + \lambda}, \tag{10}$$

where X_y is the set of instances of leaf node y. The final objective function is derived as follows.

$$\mathrm{Obj}^{(n)} = -\frac{1}{2}\sum_{y=1}^{N}\frac{\left(\sum_{x\in X_y} a_x\right)^2}{\sum_{x\in X_y} b_x + \lambda} + \gamma N_n. \tag{11}$$

The tree structure will select the feature with the largest gain drop as the optimal splitting point, and the formula for gain drop is

$$\mathrm{Gain} = \frac{1}{2}\left[\frac{\left(\sum_{x\in X_L} a_{c_x}\right)^2}{\sum_{x\in X_L} b_{c_x} + \lambda} + \frac{\left(\sum_{x\in X_R} a_{c_x}\right)^2}{\sum_{x\in X_R} b_{c_x} + \lambda} - \frac{\left(\sum_{x\in X} a_{c_x}\right)^2}{\sum_{x\in X} b_{c_x} + \lambda}\right] - \gamma. \tag{12}$$

In the process, the most important thing is the a_{c_x} and b_{c_x} of the training samples. Due to the introduction of the cost-sensitive function, the optimal segmentation point selection according to gain will pay more attention to the small sample categories in the fog nodes, thus improving the detection accuracy of the small sample categories. This is more suitable for wireless sensor networks with uneven traffic data categories.

The optimal tree model for training XGBoost needs to calculate the gain fraction gain to find the splitting point. This means that for each data sample and feature, the corresponding gradient a_{c_x} and second-order derivative b_{c_x} need to be computed.

Most of the gradient boosting algorithms for joint learning allow the clients involved in training to transfer the gradient or feature value split candidates to the aggregator to determine the best splitting point for the overall model. A common approach to search for the best splitting point is to use the exact greedy algorithm. This algorithm enumerates the entire feature and value space to find the best splitting point. If t samples, d features, and w rounds are required, the complexity of this greedy algorithm can be as high as $O(t \times d \times w \times \mathrm{la}\, t)$, imposing a significant computational burden on the communication of fog nodes. To remedy this deficiency, this paper uses a histogram-based approximation algorithm to select the optimal features efficiently.

The histogram-based approximation algorithm first partitions all the cut points $u(u = 1, 2, \cdots, w)$ of the feature into buckets based on the quantile to obtain a set of candidate cut points $S_u = \{s_{u1}, s_{u2}, \cdots, s_{ul}\}$. Then, the values of the sample features are divided into buckets according to the set, and the gradient and second-order derivatives of the sample statistics in each bucket are accumulated to obtain A_{ul} and B_{ul}. Finally, the best splitting point is found on these accumulated statistics. The core idea of the quantile algorithm is to take the quantile of a feature according to its distribution and replace the true feature value by the quantile. In essence, it is a segmented discretization of continuous features to reduce the computational complexity. However, the direct application of this approximation algorithm to the joint learning framework may lead to the inability of the trained model to adapt to the bias of each data, especially in unbalanced data and non-IID data.

To solve this problem, instead of bucketing all participants' data in the same proportion, the improved algorithm in this paper considers the size of the local dataset on the participant's client in relation to the size of other participants to bucket proportionally (e.g., percentile). First, D_p is defined to represent the eigenvalue of the u-th dimensional.

$$D_u = \{(i_{1u}, b_1), (i_{2u}, b_2), \cdots, (i_{tu}, b_t)\}. \tag{13}$$

The sorting function is defined as follows.

$$r_u(k) = \frac{\sum_{(i,b)\in D_u, i<k} b}{\sum_{(i,b)\in D_u} b}. \tag{14}$$

This function represents the proportion of the sample distribution whose eigenvalue is less than k. The second derivative b can be interpreted as a measure of the sample's weight. Under the sorting function, find a group of points $S_u = \{s_{u1}, s_{u2}, \cdots, s_{ul}\}$ that satisfy

$$\left| r_n\left(s_{n,y}\right) - r_n\left(s_{n,y+1}\right) \right| < \varepsilon, \tag{15}$$

where ε is the sampling rate, which is the proportional size of each bucket. Eventually, a cut-off point of $1/\varepsilon$ will be obtained. In this paper, we set the parameter μ_n, $\mu_n = |C_n|/|D|$ to denote the size of the data volume $|C_n|$ of client n relative to the data volume D of the server. μ_n is used to adjust the sampling rate or bucket size of different clients, and the sampling rate of the client $\varepsilon_n = \varepsilon_s \mu_n$, where ε_s represents the server's sampling rate, enabling the approximation algorithm to dynamically adapt to imbalanced data. These parameters are passed and interacted in the joint learning framework to improve the overall model detection.

4. Result Analysis and Discussion

4.1. Experimental Environment. In the intrusion detection experiment, the coverage area of the perception layer network is 100 m × 100 m, with 100 perception nodes. A laptop computer is used as the gateway node, with 10 detection nodes and 90 normal nodes. The duration of each simulation is set to 200 s, and the average value of 20 simulations is used for the experimental results.

In this paper, NS-2 is used for simulation experiments to verify the effectiveness of the algorithm. The experimental platform is a desktop computer equipped with Core i9 9900k and GTX1060, using Anaconda 3.7 software and the federated learning framework Pysyft 0.3 for simulation experiments. In order to verify the effectiveness of the proposed method for intrusion detection in the perception layer network environment, the dataset WSN-DS is used in this paper. The WSN-DS dataset is specifically designed for intrusion detection in WSNs. The WSN-DS dataset contains 374,661 records, and each sample of the dataset contains 23 attributes. Sixty percent and 40% of the WSN-DS dataset were used as the training set and test set, respectively.

4.2. Evaluation Indicators. The detection results of the intrusion detection system include the following four types: (1) True positive (TP) indicates the proportion of abnormal behaviors correctly identified as abnormal behaviors. (2) False positive (FP) indicates the proportion of normal behaviors incorrectly identified as abnormal behaviors. (3) True negative (TN) indicates the proportion of normal behaviors correctly identified as normal behaviors. (4) False negative (FN) indicates the proportion of abnormal behaviors incorrectly identified as normal behaviors.

Based on the above four detection results, the accuracy rate, false negative rate, missing rate, and area under the subject's working characteristic curve are further evolved as the evaluation indexes adopted for the intrusion detection technique in this paper.

(1) Accuracy precision indicates the proportion of correctly identified abnormal lines and normal behaviors as occupied and is calculated as

$$P = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}. \tag{16}$$

(2) False positive rate (FPR) indicates the percentage of normal behavior identified as abnormal behavior

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}}. \tag{17}$$

(3) False negative rate (FNR) indicates the percentage of abnormal behaviors identified as normal behaviors

$$\mathrm{FNR} = \frac{\mathrm{FN}}{\mathrm{TP} + \mathrm{FN}}. \tag{18}$$

Overall, the proposed scheme in this paper is divided into 3 main parts: the first part is a specific introduction to the Fed-XGB algorithm; the second part is the improvement of the XGBoost algorithm; the third part is the improvement of the approximation algorithm. The detailed flow is shown in Figure 6.

4.3. Experimental Result Analysis

4.3.1. Parameter Tuning Experiments. Firstly, in order to gain a more comprehensive understanding of the impact of the learning rate on algorithm performance, we conducted a series of experiments systematically adjusting the learning rate and observing its effects on the convergence time and overall accuracy of the model. Through these experiments, our aim was to provide a clear understanding of why specific learning rates were chosen and how such choices influenced the performance of the algorithm.

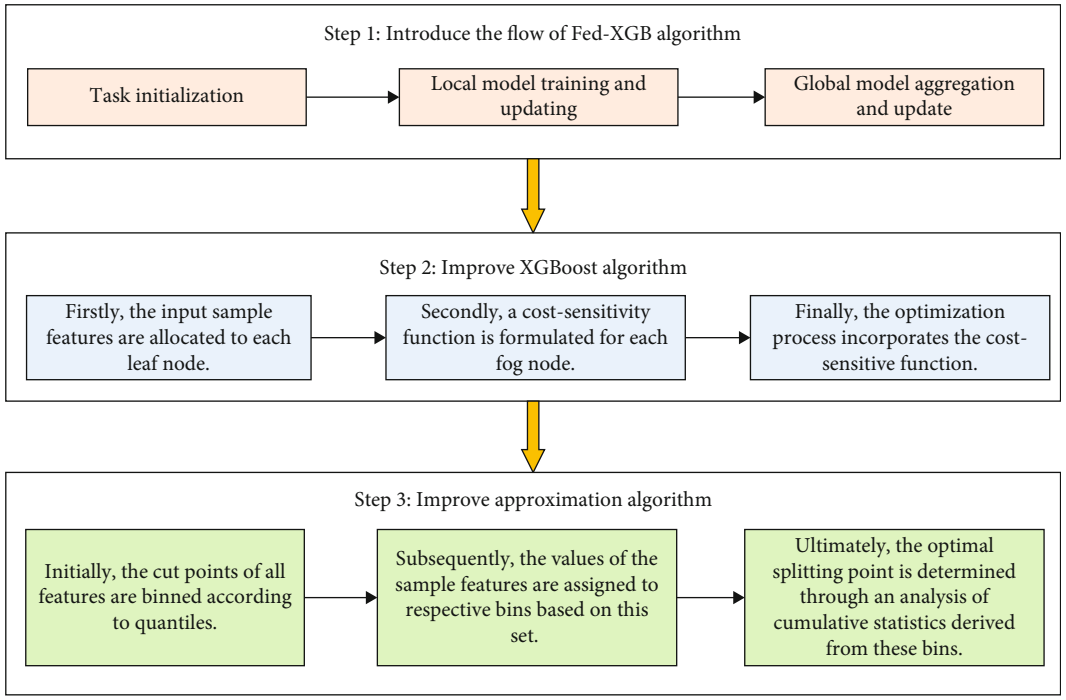


Figure 6: Fed-XGB algorithm design flow.

Table 1: Learning rate experiment results.

Learning rate	Convergence time	Accuracy
0.001	40	88.7%
0.01	25	95.2%
0.1	18	97.1%
0.5	15	93.3%
1	12	85.5%

The experimental results of Table 1 offer valuable insights into the impact of different learning rates on the convergence time and overall accuracy of the model. In the initial setting with a learning rate of 0.001, the model swiftly reached convergence in 40 epochs, but the accuracy was relatively lower at 88.7%. As we increased the learning rate to 0.01, the model exhibited faster convergence in 25 epochs and achieved a higher accuracy of 95.2%. A moderate learning rate of 0.1 led to even quicker convergence in 18 epochs, coupled with an improved accuracy of 97.1%. However, a higher learning rate of 0.5 resulted in a slightly longer convergence time of 15 epochs but significantly enhanced the model's accuracy to 93.3%. Notably, a learning rate of 1.0 demonstrated the fastest convergence in 12 epochs but at the cost of reduced accuracy, achieving 85.5%. These quantitative observations underscore the necessity for careful experimentation to strike the right balance between convergence time and model accuracy. In this context, a learning rate of 0.1 stands out as an optimal choice, demonstrating swift convergence and high accuracy. This emphasizes the critical importance of fine-tuning the learning rate parameter during the training process to achieve optimal algorithm performance.

Table 2: Impact of decision tree maximum depth on model performance.

Learning rate	Accuracy	FPR	FNR
3	91.20%	0.91%	0.92%
5	94.40%	0.83%	0.84%
7	97.90%	0.62%	0.71%
9	97.60%	0.74%	0.78%
11	92.50%	0.81%	0.82%

Decision tree's maximum depth, as a critical hyperparameter, directly influences the model's structure and performance. Through quantitatively evaluating the model's performance at different maximum depths, we aim to find an optimal depth that achieves superior fitting on the training data while ensuring robust generalization on unseen data. The specific experimental results are presented in Table 2.

Observing Table 2, the significant impact of the maximum depth on model performance is evident. Firstly, as the maximum depth gradually increases, the overall accuracy of the model shows a clear upward trend. Particularly noteworthy is the peak accuracy achieved when the maximum depth reaches 7, indicating that a relatively larger maximum depth has a positive effect on improving the model's fit to the training data. Next, focusing on the influence of the maximum depth on false positive rate (FPR) and false negative rate (FNR), it is observed that at a maximum depth of 7, FPR is relatively low, and FNR is maintained at a relatively low level. This suggests that the model performs well in classifying positive and negative categories, especially

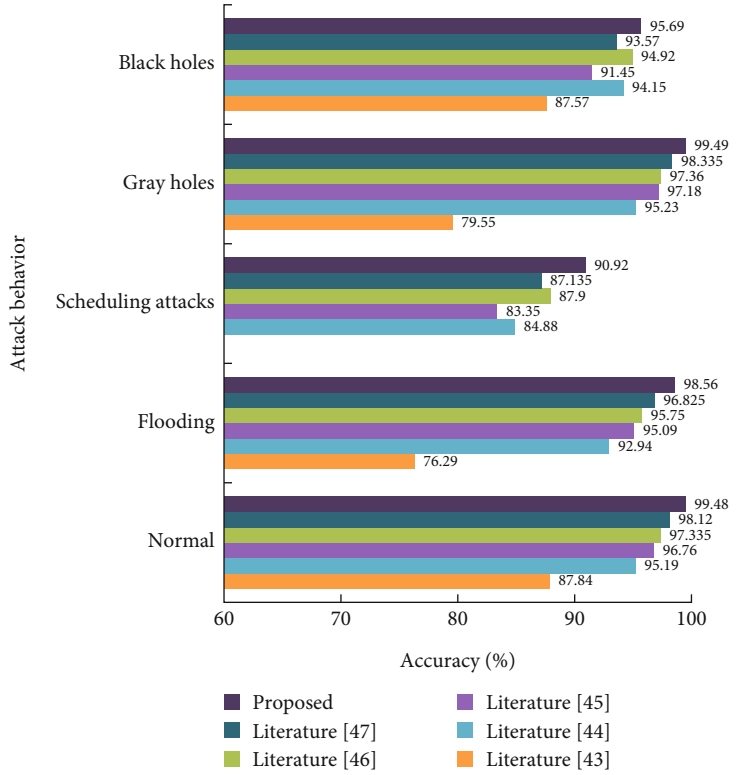


Figure 7: Comparison of accuracy.

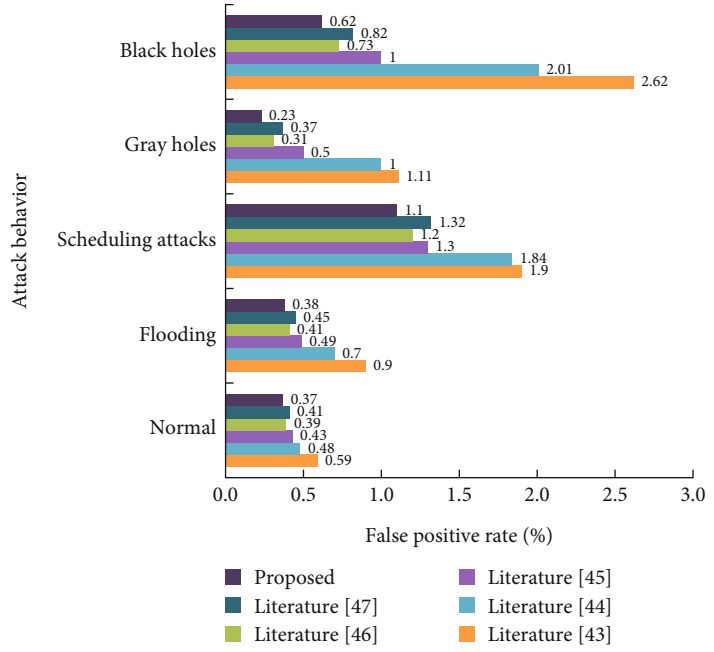


Figure 8: Comparison of FPR.

concerning misclassifications. Overall, at a maximum depth of 7, the model reaches a balance point. Through this experiment, we conclusively deduce that the decision tree's maximum depth significantly affects the model's complexity and generalization ability. When choosing the maximum depth, a careful balance between model accuracy and generalization

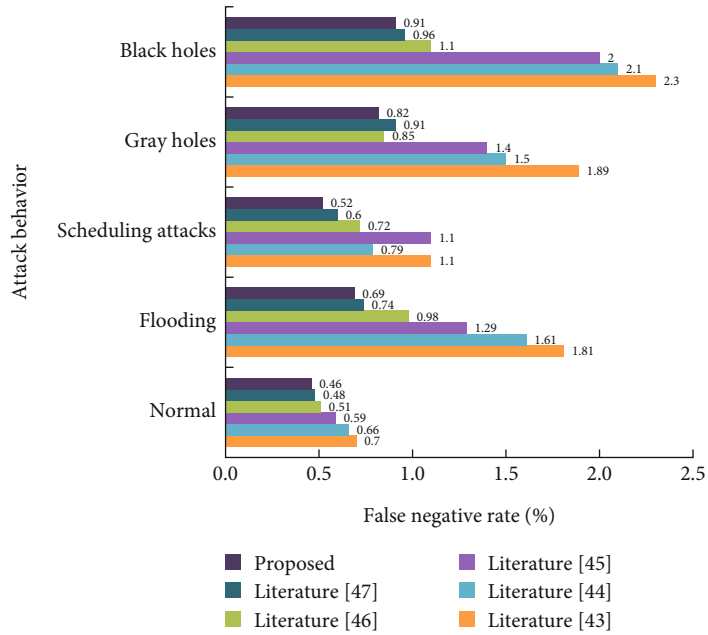


Figure 9: Comparison results of FNR.

capability on unseen data is necessary. The scenario with a maximum depth of 7 appears to strike an ideal balance, demonstrating superior performance in both aspects.

4.3.2. Algorithm Validation and Comparative Analysis. In order to verify the algorithm validation of the proposed method, the algorithms of literatures [42–46] are selected as the comparison algorithms in this section. Literature [42] is a novel intrusion detection system that combines the fuzzy C-mean clustering (FCM) algorithm with the support vector machine (SVM). It is able to detect anomalies in cloud environment with higher detection accuracy and lower false alarm rate. Literature [43] is an intrusion detection prediction model that uses the latest, publicly available intrusion detection datasets covering a wide range of attack types as training. Literature [44] provides an in-depth study and analysis of real intrusion detection cases by combining the current popular blockchain technology with federated learning. Literature [45] proposed a method based on long short-term memory (LSTM) to detect network attacks in Internet of Things (IoT) networks supported by software-defined networking (SDN) for intrusion detection systems. They introduced an LSTM-based architecture for effective multiclass classification of network attacks in IoT networks. Literature [46] presented a novel architecture for an unsupervised intrusion detection algorithm using a layered approach to enhance the security of integrated software-defined wireless sensor networks. In the proposed architecture, sensors analyze data from different regions clustered based on criteria such as entropy and cumulative point similarity. The analysis results are then sent to the software-defined wireless sensor network controller, which makes decisions after the final inspection of data as either normal or anomalous. These five comparative approaches cover the latest intrusion detection datasets as well as the latest research techniques and can represent the current state-of-the-art technology.

Table 3: Comparison of the overall detection accuracy of the 4 algorithms.

Algorithm	Overall detection accuracy %
Literature [42]	83.20%
Literature [43]	87.08%
Literature [44]	93.48%
Literature [45]	97.13%
Literature [46]	97.89%
Proposed	97.93%

The proposed algorithm and the control algorithm were trained on the training set, and then, the algorithm was validated on the test set. The 10-fold cross-validation method was used to take the average of the experimental results for comparison.

Figure 7 shows the accuracy of the proposed algorithm and the comparison algorithm. Compared with the comparison algorithm, the average accuracy of proposed algorithm in detecting five behaviors is 95.69%, 99.49%, 90.92%, 98.56%, and 99.48%, respectively, and the accuracy of this algorithm is optimal in detecting different behaviors. The algorithm of literature [42] is time-consuming and has the worst detection rate, followed by the algorithm of literature [43], which is better than that of literature [42]. The detection rates of literatures [43, 44] are similar, and this algorithm is slightly better. Literature [45] presents competitive results, particularly noteworthy in scheduling attacks and black hole detection, where it outperforms literature [42] and literature [44]. The proposed algorithm still maintains higher accuracy across different behaviors. Literature [46]

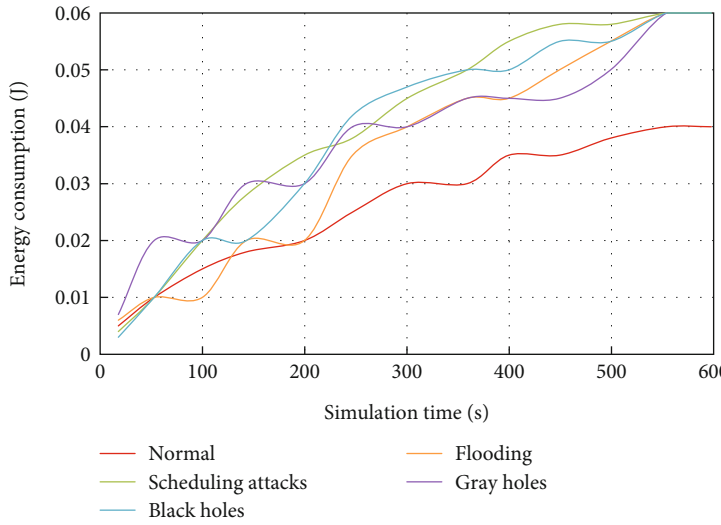


Figure 10: Experimental results of energy consumption of the proposed algorithm.

showcases commendable performance, with accuracy rates comparable to or exceeding literature [43] and literature [44]. However, the proposed algorithm consistently achieves the highest accuracy in all evaluated behaviors.

Figure 8 provides a comprehensive overview of the false alarm rate comparison between our proposed algorithm and the reference algorithms (literature [42–46]). In our evaluation, the proposed algorithm consistently outperforms the comparison algorithms across all five behaviors. Specifically, it attains the lowest false alarm rate for detecting normal behavior at 0.62%, showcasing its precision in distinguishing regular network activity. Furthermore, for the identification of gray holes, our algorithm achieves a remarkable false alarm rate of 0.23%, again surpassing the comparison algorithms. In the detection of scheduling attacks, the proposed algorithm maintains a low false alarm rate of 1.1%, indicating its effectiveness in discerning this particular threat. For flooding attacks and black holes, our algorithm achieves impressive false alarm rates of 0.38% and 0.37%, respectively, further underscoring its robust performance across a spectrum of network security challenges. These results collectively highlight the superiority of our proposed algorithm in minimizing false alarms compared to the selected reference algorithms.

In Figure 9, the false negative rate (FNR) comparison across literature [42], literature [43], literature [44], literature [45], literature [46], and the proposed algorithm provides valuable insights. The proposed algorithm consistently exhibits superior performance, achieving the lowest average FNR across all five behaviors. Specifically, it excels in detecting normal behavior with an FNR of 0.48%, showcasing its efficacy in minimizing instances of failing to detect genuine network activities. Literature [45] demonstrates competitive results, particularly in detecting gray holes and scheduling attacks with FNRs of 0.85% and 0.72%, respectively. Meanwhile, literature [46] stands out in handling flooding scenarios, achieving a low FNR of 0.74%. However, when considering the average FNR across diverse behaviors, the proposed algorithm outperforms all comparison algorithms, underscoring its robustness in ensuring a low rate of false negatives in intrusion detection scenarios.

Table 3 gives the results of the comparison between the algorithm of this paper and the comparison algorithm in terms of the overall detection accuracy.

Finally, the impact of using this paper's algorithm with or without it for different types of attacks on the network energy consumption is examined, as shown in Figure 10. This energy is generated by the cluster head node sending a control packet to the aggregation node at the beginning of each communication process. In addition, some energy is consumed when the network node receives a warning message from the intruder node. From Figure 10, the energy consumption of this algorithm is 0.04 J, which is equivalent to 0.02% of the energy consumption of the whole network without this algorithm. When all 10 intrusion nodes are detected by the network, the other nodes in the network consume about 0.06 J of energy to receive the warning message, which is 0.03% of the energy of the network. Compared to a network without an intrusion detection system, the increased energy consumption of the algorithm in this paper is almost negligible.

5. Conclusion

Intrusion detection algorithms for WSNs have been proposed one after another, but there are still problems such as no practical consideration of network energy consumption and low detection accuracy and efficiency of detection algorithms. To address these problems, this paper proposes an intrusion detection algorithm based on FedAvg and XGBoost algorithms using fog computing architecture. It combines the advantages of the FedAvg algorithm on non-IID data processing. The algorithm can better utilize the devices located at the edge of the wireless sensor network to provide services to the nearest users, thus reducing the

computational pressure on the cloud center. At the same time, it provides low-latency network services to improve real-time intrusion detection. The simulation experimental results prove that the intrusion detection method proposed in this paper has higher accuracy and more stable detection performance. Compared with other comparative models, this model has a higher detection rate, lower false alarm rate, and leakage rate. And this paper model of lower energy consumption can be applied to the actual WSNs scenario, enhancing the WSN resistance to internal network attacks while extending the life of the network. However, there are still some differences between the settings of some parameters in the experimental simulation process and the application in the real environment, and there are privacy issues. The next step will be to conduct deeper research on the confidentiality and compression of model parameters to improve the privacy and efficiency of joint learning.

Data Availability

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

Conflicts of Interest

The author declares that there are no conflicts of interest.

Acknowledgments

This work is supported by the Henan Vocational College of Tuina.

References

[1] H. Landaluce, L. Arjona, A. Perallos, F. Falcone, I. Angulo, and F. Muralter, "A review of IoT sensing applications and challenges using RFID and wireless sensor networks," *Sensors*, vol. 20, no. 9, p. 2495, 2020.

[2] S. Pragadeswaran, S. Madhumitha, and S. Gopinath, "Certain investigation on military applications of wireless sensor network," *International Journal of Advanced Research in Science, Communication and Technology*, vol. 3, no. 1, pp. 14–19, 2021.

[3] P. Sanjeevi, S. Prasanna, B. Siva Kumar, G. Gunasekaran, I. Alagiri, and R. Vijay Anand, "Precision agriculture and farming using Internet of Things based on wireless sensor network," *Transactions on Emerging Telecommunications Technologies*, vol. 31, no. 12, article e3978, 2020.

[4] B. Al Hayani and H. Ilhan, "Image transmission over decode and forward based cooperative wireless multimedia sensor networks for Rayleigh fading channels in medical internet of things (MIoT) for remote health-care and health communication monitoring," *Journal of Medical Imaging and Health Informatics*, vol. 10, no. 1, pp. 160–168, 2020.

[5] V. Bianchi, P. Ciampolini, and I. De Munari, "RSSI-based indoor localization and identification for ZigBee wireless sensor networks in smart homes," *IEEE Transactions on Instrumentation and Measurement*, vol. 68, no. 2, pp. 566–575, 2019.

[6] J. Q. Kadhim, I. A. Aljazaery, and H. T. S. Alrikabi, "Enhancement of online education in engineering college based on mobile wireless communication networks and IoT," *International Journal of Emerging Technologies in Learning*, vol. 18, no. 1, p. 176, 2023.

[7] P. Kumar, R. Kumar, G. P. Gupta, R. Tripathi, A. Jolfaei, and A. N. Islam, "A blockchain-orchestrated deep learning approach for secure data transmission in IoT-enabled healthcare system," *Journal of Parallel and Distributed Computing*, vol. 172, pp. 69–83, 2023.

[8] A. Alabdulatif, N. N. Thilakarathne, Z. K. Lawal, K. E. Fahim, and R. Y. Zakari, "Internet of nano-things (iont): A comprehensive review from architecture to security and privacy challenges," *Sensors*, vol. 23, no. 5, p. 2807, 2023.

[9] J. Li, M. S. Herdem, J. Nathwani, and J. Z. Wen, "Methods and applications for Artificial Intelligence, Big Data, Internet of Things, and Blockchain in smart energy management," *Energy and AI*, vol. 11, article 100208, 2023.

[10] A. A. Khan, A. A. Laghari, M. Rashid, H. Li, A. R. Javed, and T. R. Gadekallu, "Artificial intelligence and blockchain technology for secure smart grid and power distribution Automation: A State-of-the-Art Review," *Sustainable Energy Technologies and Assessments*, vol. 57, article 103282, 2023.

[11] S. M. S. Bukhari, M. H. Zafar, M. Abou Houran et al., "Secure and privacy-preserving intrusion detection in wireless sensor networks: Federated learning with SCNN-Bi-LSTM for enhanced reliability," *Ad Hoc Networks*, vol. 155, article 103407, 2024.

[12] H. M. Saleh, H. Marouane, and A. Fakhfakh, "Stochastic Gradient Descent Intrusions Detection for Wireless Sensor Network Attack Detection System Using Machine Learning," *IEEE Access*, vol. 12, pp. 3825–3836, 2024.

[13] G. S. Dhunna and I. Al-Anbagi, "A low power WSNs attack detection and isolation mechanism for critical smart grid applications," *IEEE Sensors Journal*, vol. 19, no. 13, pp. 5315–5324, 2019.

[14] W. Zhang, D. Han, K. C. Li, and F. I. Massetto, "Wireless sensor network intrusion detection system based on MK-ELM," *Soft Computing*, vol. 24, no. 16, pp. 12361–12374, 2020.

[15] M. Sarhan, W. W. Lo, S. Layeghy, and M. Portmann, "HBFL: a hierarchical blockchain-based federated learning framework for collaborative IoT intrusion detection," *Computers and Electrical Engineering*, vol. 103, article 108379, 2022.

[16] P. Wang, Y. Zhao, M. S. Obaidat et al., "Blockchain-enhanced federated learning market with social Internet of Things," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 12, pp. 3405–3421, 2022.

[17] Y. Ren, R. Liu, D. Wang et al., "A study of local differential privacy mechanisms based on federated learning," *Journal of Electronics and Information*, vol. 45, no. 3, pp. 784–792, 2023.

[18] S. Maza and M. Touahria, "Feature selection algorithms in intrusion detection system: a survey," *KSII Transactions on Internet and Information Systems (TIIS)*, vol. 12, no. 10, pp. 5079–5099, 2018.

[19] L. Tan, C. Li, J. Xia, and J. Cao, "Application of self-organizing feature map neural network based on K-means clustering in network intrusion detection," *Computers, Materials & Continua*, vol. 61, no. 1, pp. 275–288, 2019.

[20] S. Otoum, B. Kantarci, and H. T. Mouftah, "On the feasibility of deep learning in sensor network intrusion detection," *IEEE Networking Letters*, vol. 1, no. 2, pp. 68–71, 2019.

[21] Z. S. Khalafi, M. Dehghani, A. Khalili, A. Sami, N. Vafamand, and T. Dragičević, "Intrusion detection, measurement correction, and attack localization of PMU networks," *IEEE*

Transactions on Industrial Electronics, vol. 69, no. 5, pp. 4697–4706, 2022.

[22] S. R. Khonde and V. Ulagamuthalvi, "Ensemble-based semi-supervised learning approach for a distributed intrusion detection system," *Journal of Cyber Security Technology*, vol. 3, no. 3, pp. 163–188, 2019.

[23] A. A. Süzen, "Developing a multi-level intrusion detection system using hybrid-DBN," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 2, pp. 1913–1923, 2021.

[24] B. Pang, Z. Teng, H. Sun, C. Du, M. Li, and W. Zhu, "A malicious node detection strategy based on fuzzy trust model and the ABC algorithm in wireless sensor network," *IEEE Wireless Communications Letters*, vol. 10, no. 8, pp. 1613–1617, 2021.

[25] V. Kiran, S. Rani, and P. Singh, "Towards a light weight routing security in IoT using non-cooperative game models and Dempster–Shaffer theory," *Wireless Personal Communications*, vol. 110, no. 4, pp. 1729–1749, 2020.

[26] S. Hosseini and H. Seilani, "Anomaly process detection using negative selection algorithm and classification techniques," *Evolving Systems*, vol. 12, no. 3, pp. 769–778, 2021.

[27] E. Baraneetharan, "Role of machine learning algorithms intrusion detection in WSNs: a survey," *Journal of Information Technology*, vol. 2, no. 3, pp. 161–173, 2020.

[28] F. Louati and F. B. Ktata, "A deep learning-based multi-agent system for intrusion detection," *SN Applied Sciences*, vol. 2, no. 4, p. 675, 2020.

[29] K. Zhang, Z. Hu, Y. Zhan, X. Wang, and K. Guo, "A smart grid AMI intrusion detection strategy based on extreme learning machine," *Energies*, vol. 13, no. 18, p. 4907, 2020.

[30] Z. Tang, H. Hu, and C. Xu, "A federated learning method for network intrusion detection," *Concurrency and Computation: Practice and Experience*, vol. 34, no. 10, article e6812, 2022.

[31] F. Zhang and S. Wang, "Detecting group shilling attacks in online recommender systems based on bisecting k-means clustering," *IEEE Transactions on Computational Social Systems*, vol. 7, no. 5, pp. 1189–1199, 2020.

[32] R. Wang and W. Ji, "Computational intelligence for information security: a survey," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 4, no. 5, pp. 616–629, 2020.

[33] P. Arroyo, J. L. Herrero, J. I. Suárez, and J. Lozano, "Wireless sensor network combined with cloud computing for air quality monitoring," *Sensors*, vol. 19, no. 3, p. 691, 2019.

[34] W. Fang, W. Zhang, W. Chen, Y. Liu, and C. Tang, "TMSRS: trust management-based secure routing scheme in industrial wireless sensor network with fog computing," *Wireless Networks*, vol. 26, no. 5, pp. 3169–3182, 2020.

[35] A. Bashar and S. Smys, "Physical layer protection against sensor eavesdropper channels in wireless sensor networks," *IRO Journal on Sustainable Wireless Systems*, vol. 3, no. 2, pp. 59–67, 2021.

[36] H. Xie, Z. Yan, Z. Yao, and M. Atiquzzaman, "Data collection for security measurement in wireless sensor networks: a survey," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 2205–2224, 2019.

[37] R. S. Raghav, K. Thirugnansambandam, and D. K. Anguraj, "Beeware routing scheme for detecting network layer attacks in wireless sensor networks," *Wireless Personal Communications*, vol. 112, no. 4, pp. 2439–2459, 2020.

[38] S. Shams Shamsabad Farahani, "A review on reliable data transport protocols in wireless sensor networks," *Journal of Electrical and Computer Engineering Innovations (JECEI)*, vol. 10, no. 1, pp. 107–122, 2022.

[39] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 1, pp. 1–22, 2019.

[40] L. N. Tidjon, M. Frappier, and A. Mammar, "Intrusion detection systems: a cross-domain overview," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 4, pp. 3639–3681, 2019.

[41] R. Abedi, R. Costache, H. Shafizadeh-Moghadam, and Q. B. Pham, "Flash-flood susceptibility mapping based on XGBoost, random forest and boosted regression trees," *Geocarto International*, vol. 37, no. 19, pp. 5479–5496, 2022.

[42] A. N. Jaber and S. U. Rehman, "FCM–SVM based intrusion detection system for cloud computing environment," *Cluster Computing*, vol. 23, no. 4, pp. 3221–3231, 2020.

[43] J. L. Leevy and T. M. Khoshgoftaar, "A survey and analysis of intrusion detection models based on cse-cic-ids2018 big data," *Journal of Big Data*, vol. 7, no. 1, pp. 1–19, 2020.

[44] D. Preuveneers, V. Rimmer, I. Tsingenopoulos, J. Spooren, W. Joosen, and E. Ilie-Zudor, "Chained anomaly detection models for federated learning: an intrusion detection case study," *Applied Sciences*, vol. 8, no. 12, p. 2663, 2018.

[45] R. Chaganti, W. Suliman, V. Ravi, and A. Dua, "Deep learning approach for SDN-enabled intrusion detection system in IoT networks," *Information*, vol. 14, no. 1, p. 41, 2023.

[46] A. Arkan and M. Ahmadi, "An unsupervised and hierarchical intrusion detection system for software-defined wireless sensor networks," *The Journal of Supercomputing*, vol. 79, no. 11, pp. 11844–11870, 2023.