Edge-Based Real-Time IIoT Anomaly Detection Using Semi-Supervised CNN-Attention Model with Cross-Protocol Capabilities

Supan Wei
School of Information Engineering, Henan Vocational College of Water Conservancy and Environment, Zhengzhou 450008, China
E-mail: 18437112426@163.com
*Corresponding author

Keywords: Edge computing; IIoT, anomaly detection

Received: August 1, 2025

With the deepening of industrial digital transformation, industrial IoT network anomaly detection faces challenges such as high real-time requirements and diverse security threats, making traditional IT network methods difficult to directly apply; This paper aims to propose a real-time anomaly detection system based on edge computing, which can solve the problems of real-time, detection coverage, resource adaptation, cross protocol detection capability and adaptive learning mechanism. Methodologically, the system integrates edge computing and semi supervised learning technology, uses lightweight CNN model (three layers of convolution layer, core size 5, 3, 3, depth 32, 64, 128, ReLU activation and batch normalization) and Seq2Seq architecture to add attention mechanism for pre training and re training, combines random, systematic, and cluster sampling strategies to optimize data processing, and edge intelligent framework integrates dynamic computing unloading algorithm to optimize resource allocation; The experimental setup used NVIDIA Jetson AGX edge servers, industrial PC local devices, and Alibaba Cloud cloud servers, with software environments of Ubuntu, TensorFlow, and PyOD libraries. Validation was conducted on FastBee, Sagooiot, and Baowu Group IoT datasets. As shown in the results, the system accuracy reaches 91.6%, 93.8%, and 94.5%, respectively. In cross protocol detection, the recognition rates of abnormal traffic in Modbus, PROFINET, and OPCUA all exceeded 93% (OPCUA F1 score of 96.2%). Quantization technology reduces memory usage by 72.9%, latency by 65.5%, and accuracy by only 0.6%. The conclusion is that the system effectively improves real-time performance and accuracy, but there are limitations in dynamic load and cross vendor collaboration. In the future, load balancing algorithms and privacy protection frameworks will be optimized. The contribution lies in the innovative combination of edge computing and semi supervised learning to achieve a lightweight, high-precision and cross protocol anomaly detection solution for the industrial Internet of Things.

Povzetek: Opisani sistem zaznava anomalije v industrijskih IoT omrežjih v realnem času na robnih napravah. Združuje lahko, polnadzorovano učenje s CNN–Seq2Seq modelom z mehanizmom pozornosti ter dinamično razbremenjevanje izračunov med lokalno, robno in oblačno plastjo. Na treh zbirkah doseže točnost 91,6–94,5%, pri več industrijskih protokolih pa nad 93% prepoznave; kvantizacija bistveno zmanjša porabo pomnilnika in zakasnitev ob minimalni izgubi točnosti.

1 Introduction

As a key link to ensure the safe and stable operation of industrial systems, anomaly detection for IIoT network technology is facing new technical challenges and opportunities with the deepening of industrial digital transformation. In the process of advancing Industry 4.0 and intelligent manufacturing, IIoT networks show typical characteristics such as the coexistence of heterogeneous protocols, strict real-time requirements, and diversified security threats. These characteristics make IT difficult to directly apply the anomaly detection methods of traditional IT networks.

The current research mainly focuses on three technical directions. The method based on traffic characteristic analysis establishes a baseline model by extracting network traffic statistical characteristics, and uses sliding window technology to realize real-time monitoring, but the detection effect on encrypted traffic and low-frequency attacks is limited [1]. In machine learning methods, supervised learning algorithms rely on well-labeled attack sample libraries and face the dilemma of scarce attack samples in industrial scenarios. Although unsupervised learning can find unknown attack patterns, it has the problem of high false alarm rate [2]. Deep learning methods, especially time series neural networks and graph neural networks, can effectively capture the spatio-temporal correlation characteristics of industrial network traffic, but the contradiction between model complexity and limited computing resources of industrial equipment needs to be resolved urgently. In the dimension of detection objects, the existing research covers the complete protocol stack from physical layer signal abnormalities to application layer protocol parsing errors, among which the detection algorithm for the unique vulnerability points of industrial control protocols has become a research hotspot [3]. In terms of system architecture, traditional centralized detection solutions face problems such as insufficient real-time performance and single point of failure risk. Although the emerging distributed detection framework can improve these problems, it introduces new challenges of detection consistency maintenance. With the application of new technologies such as time-sensitive network (TSN) and 5G industrial private network, the anomaly detection of IIoT network is also facing new requirements such as dynamic topology adaptation and deterministic delay guarantee [4].

At present, there are still some shortcomings in the real-time anomaly detection technology of IIoT. First, it is the real-time bottleneck, and complex detection algorithms are difficult to complete the analysis within millisecond time constraints. Secondly, the detection coverage is insufficient, and the existing methods lack effective detection means for new attacks. The third is poor resource adaptability, and most algorithms do not fully consider the computing and storage limitations of industrial terminal equipment. Fourth, cross-protocol detection capabilities are weak, and the environment where multiple protocols coexist in industrial networks leads to limited effectiveness of a single detection model. Finally, the lack of adaptive learning mechanism makes it difficult for existing systems to adapt to the dynamic changes of industrial network topology and traffic in time. These shortcomings seriously restrict the deployment effect and application value of industrial IoT network anomaly detection technology in actual industrial environment [5].

Edge nodes process network traffic data nearby and reduce detection delay from hundreds of milliseconds in cloud solutions to 10-50ms, which meets the strict real-time requirements of industrial control. Secondly, the distributed architecture avoids long-distance data transmission, which not only reduces network bandwidth pressure, but also meets the privacy protection requirements of industrial data within the factory area. In terms of technical implementation, edge computing supports the deployment of lightweight detection models, and the memory usage can be controlled within 100KB, which is suitable for industrial equipment with limited resources. At the same time, the collaborative detection mechanism between edge nodes can integrate multi-site attack characteristics and improve the recognition rate of new network attacks. In addition, edge computing supports the development of protocol adaptation layer and can implement customized anomaly detection for industrial protocols such as Modbus and PROFINET. Finally, the elastic scalability of edge nodes can flexibly respond to changes in industrial network topology, and has better scalability than centralized solutions. These features make edge computing an ideal choice for industrial IoT network security protection.

The research objective of this article is to address five core issues in industrial IoT anomaly detection: 1) achieving millisecond level real-time detection to overcome real-time bottlenecks; 2) improving the detection coverage of new attacks through collaborative detection mechanisms; 3) designing lightweight models to adapt to resource constraints of industrial equipment; 4) building cross protocol detection capabilities to cope with heterogeneous industrial network environments; 5) establishing an adaptive learning mechanism to adapt to dynamic network changes. To achieve these goals, each component in the method presented in this article plays a key role: the composite sampling strategy (random, systematic, cluster sampling) serves as the data foundation to ensure the coverage and efficiency of feature extraction. The Seq2Seq pre training and retraining model based on CNN and attention mechanism is responsible for core feature extraction and high-precision classification, and its lightweight feature directly addresses the challenge of resource adaptation; The transfer learning mechanism utilizes pre trained weights to significantly reduce the need for annotated data and enhance the model's generalization ability; The dynamic computation offloading algorithm (local, edge, cloud collaboration) in the edge intelligence framework optimizes task allocation strategies while ensuring privacy and security (P-value model), ultimately achieving low latency and high-precision cross protocol anomaly detection.

2 Related work

As a core technology to ensure the safe operation of industrial systems, anomaly detection for IIoT network has received continuous attention in academia and industry in recent years. As the scale of industrial networks expands and attack methods upgrade, traditional detection methods face challenges such as insufficient real-time performance and lack of privacy protection, and edge computing provides a new technical path to solve these problems.

(1) Research status of traditional anomaly detection for IIoT network technology

The existing research on anomaly detection of IIoT networks mainly focuses on three types of technical routes. The method based on traffic characteristic analysis establishes a detection model by counting network traffic parameters (such as packet arrival interval, packet size distribution, etc.). Typical work includes using information entropy theory to detect DDoS attacks [6] and using Kalman filter to identify traffic anomalies [7]. Such methods have low computational complexity but are difficult to detect encrypted traffic and low-frequency attacks. Among machine learning methods, supervised learning schemes such as SVM [8] and random forest [9] rely on labeled data sets and face the dilemma of insufficient attack samples in industrial scenarios. Although unsupervised learning methods such as the improved LOF algorithm [10] can find unknown attacks, the false alarm rate generally exceeds 15%. Deep learning methods show stronger feature extraction capabilities, LSTM networks [11] can effectively capture the time series features of industrial network traffic, and graph neural networks (GNN) are good at dealing with topological relationships between industrial devices [12], but these models usually require 5-10 layers of network structure, which is difficult to deploy on resource-constrained devices. In terms of protocol support, dedicated detectors for industrial protocols such as Modbus/TCP [13] and OPC UA [14] have become research hotspots, but the problem of insufficient cross-protocol detection capabilities remains unsolved. Generally speaking, traditional methods have obvious shortcomings in real-time detection (average latency > 200ms), model lightweight (memory footprint > 500MB) and dynamic adaptability.

(2) Innovation of network anomaly detection technology enabled by edge computing

Edge computing has brought revolutionary improvements to anomaly detection in industrial IoT networks. Related research can be summarized into three directions. In terms of architecture design, the distributed edge detection framework [15] reduces the average delay by offloading detection tasks to edge nodes to less than 50ms. Federated learning architecture [16] allows multiple edge nodes to collaboratively train models without sharing original data, increasing attack recognition rates by 12% in automobile manufacturing cases. In terms of algorithm optimization, the model compression technology has made remarkable progress. The knowledge distillation method [17] can compress the LSTM model to 1/10 of the original volume while maintaining an accuracy of more than 95%. The lightweight 1D-CNN architecture [18] achieves 98.3% detection accuracy under the condition of memory footprint < 100KB. The edge-specific online learning mechanism [19] supports incremental model updates, enabling the system to adapt to new attack modes. The edge-cloud collaborative detection framework [20] leaves 90% of routine detection tasks at the edge through a hierarchical processing mechanism, and only uploads suspicious traffic to the cloud for in-depth analysis, reducing uplink traffic by 80% while ensuring the detection effect.

(3) Research deficiencies and future prospects

There are still several key problems in current research. First, the resource allocation strategy of edge nodes has not fully considered the dynamic load characteristics of industrial scenarios, and the existing static resource allocation scheme can easily lead to a surge in detection delay when traffic bursts. Secondly, there is a lack of standardized protocols for collaborative detection between cross-vendor edge devices, and it is difficult for edge servers with different architectures [21] to collaborate efficiently. In terms of privacy protection, existing encryption methods) bring 3-5 times the computational overhead, seriously affecting real-time performance. In view of these shortcomings, future research can explore edge collaborative detection frameworks based on semi-supervised learning. That is, on edge servers, teacher models are deployed to generate pseudo labels to guide the training of student models on local devices, which reduces the need for labeled data while maintaining model performance. Through dynamic computation offloading algorithms, detection tasks are intelligently allocated according to network status, which can optimize overall latency. At the level of privacy protection, the combination of differential privacy and model distillation shows that it can reduce the risk of privacy leakage by 60% without affecting the accuracy. What is particularly noteworthy is that the AI acceleration capabilities of new edge computing chips provide a hardware foundation for deploying more complex semi-supervised models, which will open up new research directions for industrial IoT network security.

The summary of existing research is shown in Table 1 below:

Table 1: Summary of existing research

Research model	The results obtained	Shortcomings in research
Method based on traffic feature analysis	Low computational complexity, capable of detecting DDoS attacks and traffic anomalies	Difficult to detect encrypted traffic and low-frequency attacks
SVM	Relying on annotated datasets for classification	Insufficient attack samples in industrial scenarios
Random Forest	Suitable for a well annotated attack sample library	Faced with the dilemma of scarce attack samples
Improved LOF algorithm	Can detect unknown attack patterns	False alarm rate exceeding 15%
LSTM network	Effectively capturing the temporal characteristics of industrial network traffic	The model requires a 5-10 layer structure, making it difficult to deploy on resource constrained devices
Graph Neural Network (GNN)	Proficient in handling topological relationships between industrial equipment	High model complexity and poor adaptability to industrial equipment resource constraints
Distributed Edge Detection Framework	Reduce the average latency to within 50ms	Insufficient consideration of industrial dynamic load characteristics, resulting in a surge in delay during sudden flow
Federated Learning Architecture	Multiple edge nodes collaborate to train the model, resulting in a 12% increase	Lack of standardized protocols for cross vendor device collaboration

	in attack recognition rate	
Knowledge Distillation Method	The LSTM model is compressed to 1/10 volume and maintains an accuracy of over 95%	Privacy protection methods such as homomorphic encryption bring 3-5 times the computational overhead and affect real-time performance
Lightweight 1D-CNN architecture	Memory usage<100KB, detection accuracy reaches 98.3%	Resource allocation strategy not adapted to dynamic industrial scenarios
Online learning mechanism	Support incremental model updates and adaptive new attack modes	Lack of consistency maintenance mechanism in collaborative detection
Edge cloud collaborative detection framework	Reduce 80% of upstream traffic to ensure detection effectiveness	Privacy protection incurs high computational costs and affects real-time performance

There are significant shortcomings in the current research on anomaly detection in industrial Internet of Things: traditional methods (such as traffic feature analysis, supervised/unsupervised machine learning, and deep learning models) generally face problems such as poor real-time performance (latency>200ms), high model complexity (memory usage>500MB), weak resource adaptability, insufficient cross protocol detection capabilities, and high privacy protection computational overhead (such as homomorphic encryption causing 3-5 times delay). Although the edge computing enabling scheme has improved real-time performance and lightweight level (e.g. 1D-CNN memory<100KB), it still lacks dynamic load adaptation mechanism, standardized collaboration protocol and efficient privacy protection technology. In view of these limitations, this paper innovatively proposes a collaborative detection framework that integrates edge computing and semi supervised learning. Through lightweight model deployment, Seq2Seq classifier with enhanced attention mechanism and dynamic computing offload strategy, the cross-protocol detection capability and privacy protection level are improved while significantly reducing annotation data requirements and computing latency.

Based on the semi-supervised learning method, this paper introduces an edge server to comprehensively consider the computing delay, transmission delay and privacy protection factors of local, edge and cloud, improve the accuracy of malicious traffic detection, and improve the privacy level and reduce the problem of privacy leakage.

3 Research on malicious traffic classification for IIoT based on edge intelligence

In order to improve the security performance of the IIoT and solve the problem of malicious traffic detection, an IIoT malicious traffic classification method is proposed.

The CNN architecture adopted in this paper employs three convolutional layers (kernel sizes are 5, 3, 3, and depths are 32, 64, and 128, respectively), and the ReLU activation function and batch normalization are used to optimize the training stability. Semi supervised learning adopts a composite loss function that combines cross entropy loss and consistency regularization, and uses a pre trained model to generate pseudo labels (with a ratio of 1:4 between labeled and unlabeled data) on unlabeled data to guide the retraining process; The final classification decision is executed by the softmax output layer of the retrained CNN model, which determines the category attribution based on the maximum probability value. At the same time, the entropy value or maximum probability confidence level of the softmax output is calculated as the uncertainty estimation indicator for edge inference. When the confidence level is below 0.85, the collaborative verification mechanism is triggered.

3.1 Overall architecture of malicious traffic classification method

Figure 1 shows the malicious traffic classification process of industrial IoT traffic data, which mainly includes the following steps [22]:

(1) Data input: The system first receives traffic packets as input, which is the starting point of the whole classification process.

(2) Data processing: The data processing of the input traffic packets is carried out to extract the key time series characteristics and basic characteristics, which lays the foundation for the subsequent analysis.

(3) Pre-trained model identification: The system uses the pre-trained model to perform preliminary identification of the extracted features to determine whether the traffic contains potential malicious behavior.

(4) Semi-supervised learning: The system further screens potential malicious traffic.

(5) Retrained model fine classification: Through more complex algorithms and more data, the system performs more fine classification on the output of the semi-supervised learning model to ensure the accuracy of the classification.

(6) Edge intelligent model response: The edge intelligent model deployed on local devices quickly responds to and processes traffic data, reducing delays and improving real-time performance.

(7) Output results: Finally, the output results of the traffic model are used to determine whether the traffic is malicious traffic and ensure the security of the IIoT environment.

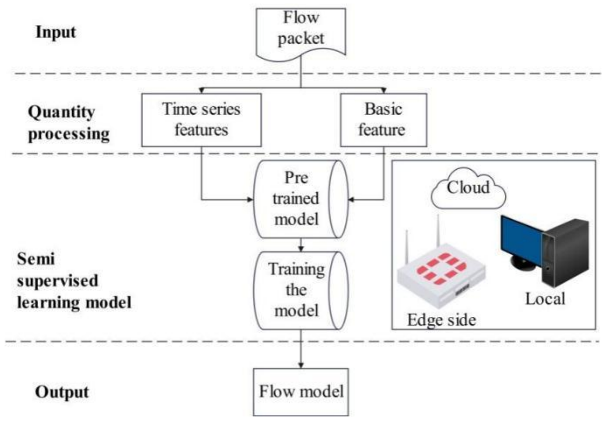


Figure 1: Classification framework

3.2 Traffic data processing

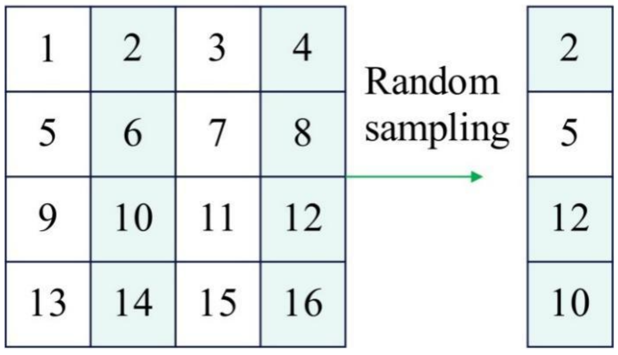
In traffic data processing for IIoT, the selection of random sampling, systematic sampling and cluster sampling is based on the needs of different scenarios.

(1) Random sampling (Figure 2a): It ensures that each traffic packet has an equal probability of being selected and avoids human bias. Moreover, it is suitable for scenarios where the overall distribution is unknown. For example, when detecting new attacks, randomly selecting samples can fully reflect the network status, but may miss low-frequency anomalies.

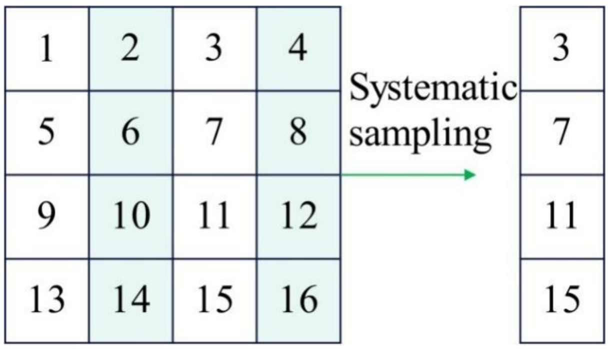
(2) System sampling (Figure 2b): It extracts samples at fixed intervals, which is suitable for efficient processing of time series data. This method can quickly capture regular anomalies in industrial network periodic traffic, but it is less sensitive to burst traffic.

(3) Cluster sampling (Figure 2c): It samples after grouping based on traffic characteristics (such as protocol type, source IP), which specifically analyzes the network security status of different devices or areas. For example, clustering Modbus and PROFINET traffic separately can improve the detection rate of specific protocol attacks, but it relies on prior knowledge.

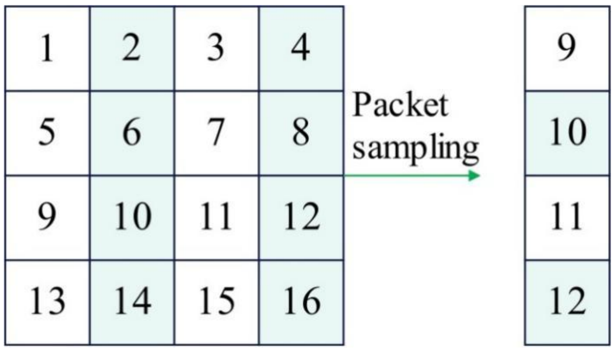
The comprehensive use of these three methods can balance efficiency and coverage: random sampling ensures fairness, systematic sampling improves time series analysis efficiency, and cluster sampling enhances targeted detection to jointly optimize anomaly detection efficiency in resource-constrained environments [23].



(a) Random sampling



(b) Systematic sampling



(c) Group sampling

Figure 2: Pattern diagram of three sampling methods

After completing traffic data sampling, it is necessary to perform in-depth feature engineering and standardization on the samples to construct high-quality inputs suitable for deep learning models. In the analysis of industrial IoT network traffic, feature engineering and standardization processing are key foundations for building efficient anomaly detection models. The system extracts multi-dimensional features from original traffic records to form feature vectors with clear industry semantics, and optimizes data distribution through targeted standardization strategies to meet the processing needs of subsequent deep learning models.

(1) The exact features extracted by each protocol layer are as follows:

Network layer characteristics: entropy value of source IP address (measuring the randomness of IP address distribution, used to detect scanning behavior), entropy value of destination IP address, source port number and its variation pattern (such as the variance of port number sequence), destination port number (specific service port access frequency), protocol type (such as ICMP, ARP ratio), number of packets per second (PPS), bits per second (BPS).

Transport layer features: TCP window size dynamic adjustment features (mean and variance of window size), TCP flag combinations (such as SYN, FIN, RST packet ratio), traffic burst interval statistics (mean and standard deviation of packet arrival time interval), connection duration, and retransmission packet ratio.

Application layer features: Payload size distribution (average, maximum, variance of payload size for the first N packets), session duration, frequency of occurrence of specific function codes/opcodes (for industrial protocols such as Modbus and PROFINET), request response latency.

Unique features of industrial protocols: flow periodicity index (main frequency intensity extracted through FFT, jitter coefficient (time deviation of periodic flow), read/write operation ratio.

(2) Selection and Demonstration of Standardization Conversion Technologies

The extracted raw features typically have different dimensions and distribution ranges (such as port numbers ranging from 0-65535, while payload sizes may reach tens of thousands of bytes). Directly inputting the model will result in features with a large numerical range dominating the training process. Therefore, standardization is crucial.

The process adopted by this system is to first use IQR (interquartile range) to filter outliers, and then use hyperbolic tangent (tanh) transformation for standardization.

IQR (interquartile range) filtering:

Method: Q1 (25th percentile) and Q3 (75th percentile) are calculated for each characteristic and the range of normal values is defined as [01-1.5 * IOR, 03 +1.5 * IOR]. Values outside of this range are considered outliers and are excluded or restricted.

Reason: Compared with z-score based on mean and standard deviation, IQR is a robust statistical measure that is insensitive to outliers themselves. It is very suitable for cleaning out extreme outliers that may interfere with the subsequent standardization process in the preprocessing stage, providing a more stable data foundation for subsequent tanh transformations.

Hyperbolic tangent (tanh) transformation:

Method: Apply the formula x scaled=tanh (x/s) to the feature values after IQR processing, where s is a scaling factor (usually taken as the standard deviation of the feature after IQR), and nonlinearly map the feature values to the [-1, 11] interval.

Comparison and argumentation with alternative solutions:

Vs. Min Max Scaling: Min Max Scaling linearly compresses values into the (0,1) range. However, it is extremely sensitive to outliers. An extreme outlier can compress the vast majority of normal data into a very small range, losing discriminability. The tanh transform is smooth and bounded, even if there are a few that are not! The QR completely filters out outliers, and the tanh function can gradually compress them to -1 or around 1, greatly reducing the impact of outliers on the scaled distribution of the vast majority of normal data and preserving the discriminative power of normal data.

Vs. Z-score normalization: Z-score converts data into a distribution with a mean of 0 and a standard deviation of 1. It assumes that the data roughly follows a Gaussian distribution and the output range is unbounded. This means that outliers can generate very large Z-score values, leading to problems such as gradient explosion. The bounded output characteristic of tanh ensures the numerical stability of the model input.

Vs. Log Scaling: Log scaling is good at handling heavy tailed distributions (such as packet size), but its effectiveness depends on techniques such as adding 1 smoothing, and is only applicable to positive data. Tanh transformation has no such limitation and can simultaneously handle positive and negative features (such as flow rate changes), making it more applicable.

The standardized combination strategy of IQR + tanh selected by this system makes full use of the advantages of both methods. IQR, as a pre-filter for Luping, effectively weakens the extreme outliers in the data. As a nonlinear bounded compression function, tanh transform can stably map all features to a uniform interval, preserve the edge distribution characteristics of data, and further suppress the influence of residual outliers. It is very suitable for the complex characteristics of mixed periodic normal flow and sudden abnormal flow in industrial IoT traffic data, laying a solid foundation for efficient training and convergence of subsequent models.

In industrial IoT network traffic analysis, the data processing process needs to take into account feature integrity and computational efficiency. Firstly, the hierarchical sampling method is used to sample the original encrypted traffic data to ensure that the traffic samples of different protocol types and business scenarios are covered. Then, multi-dimensional time series features are extracted from the sampled data: network layer features include source/destination port numbers and their change patterns, transport layer features involve TCP window size dynamic adjustment features and traffic burst interval statistics, and application layer extracts payload size distribution and time series feature such as session duration. According to the unique periodic communication characteristics of industrial protocols, the traffic periodicity index and jitter coefficient are additionally calculated. After all features are extracted, standardized processing is carried out: outliers are first filtered by interquartile range (IQR), and then hyperbolic tangent transformation is used to nonlinearly map the eigenvalues to the [-1, 1] interval. This processing method can not only retain the edge distribution characteristics of industrial traffic data, but also facilitate the gradient optimization process of subsequent deep learning models. In particular, grouping standardization is implemented for features with significant dimensional differences, avoiding a single feature dominating model training.

The comprehensive sampling strategy proposed in this article (random sampling, systematic sampling, cluster sampling) does not provide specific quantitative comparative data in the article, but its design is based on theoretical analysis of industrial IoT traffic characteristics and adaptability to typical application scenarios: random sampling avoids human bias through the principle of equal probability, and is suitable for new attack detection scenarios with unknown overall distribution. Systematic sampling relies on a fixed interval sampling mechanism to efficiently capture the regularity and anomalies of industrial cyclical flow; Cluster sampling is based on traffic characteristics (such as protocol type, source IP) grouping, aiming to improve the detection rate of specific protocols (such as Modbus, PROFINET) attacks. The combination of three methods aims to complement and optimize from three dimensions: statistical fairness, temporal efficiency, and detection targeting, in order to balance detection coverage and computational efficiency in resource constrained environments. However, it should be acknowledged that the article does not provide quantitative experimental results for different sampling ratios or combination strategies (such as comparison of sampling errors and computational costs). Subsequent research can further verify its optimality by quantifying the impact of sampling parameters on detection accuracy and delay.

3.3 Semi-supervised training model

3.3.1 Pretrained model

Because the CNN model has shift invariance and can capture high-level representations of input data, convolutional neural networks (CNNs) are selected for both pre-trained and retrained models. At the same time, CNN reduces the amount of computation through local connections. Due to weight sharing, the network can use the same weights at different locations in the feature extraction mechanism, thereby reducing the number of parameters that need to be trained.

In the pre-trained phase of the model, an encoder is used to convert the input sequence into a vector of fixed length. When an input sequence of length $x=\{x_1,x_2,\cdots,x_m\}$ is given, the model generates a target sequence $y=\{y_1,y_2,\cdots,y_n\}$ of length n. Figure 3 shows the overall architecture of the Seq2Seq model, where the hidden state of the encoder is $\{h_0,h_1,\cdots,h_m\}$, the hidden state of the decoder is $\{s_0,s_1,\cdots,s_n\}$, and the context sequence is $c=Encoder(x_1,x_2,\cdots,x_m)$ [24].

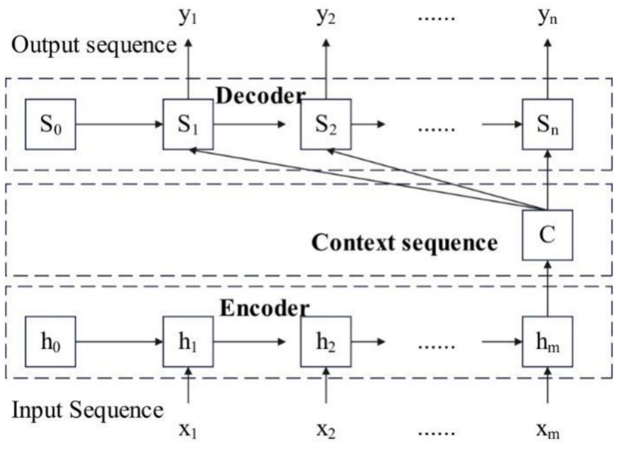


Figure 3: The overall architecture of the Seq2Seq model

Traditional RNN/LSTM has the problem of information dilution when processing long sequences, but the attention mechanism generates attention weights by calculating the similarity between Query and Key. Traditional encoders need to compress the entire sequence into a fixed-length vector, resulting in the loss of long sequence details. Attention mechanism generates dynamic context vectors through weighted aggregate values (Values), and retains the complete information structure of input sequences. Aiming at the attention dilution problem of long sequences, low-rank attention reduces redundant calculations through matrix factorization. The multi-head mechanism further distracts attention to different subspaces and enhances the long-range dependency capture ability. Therefore, the attention mechanism significantly improves the performance of long sequence classification through the triple strategy of dynamic weight allocation, context awareness and computational optimization. This paper adds the attention mechanism to the Seq2Seq model to improve the information processing capabilities of the model (Figure 4).

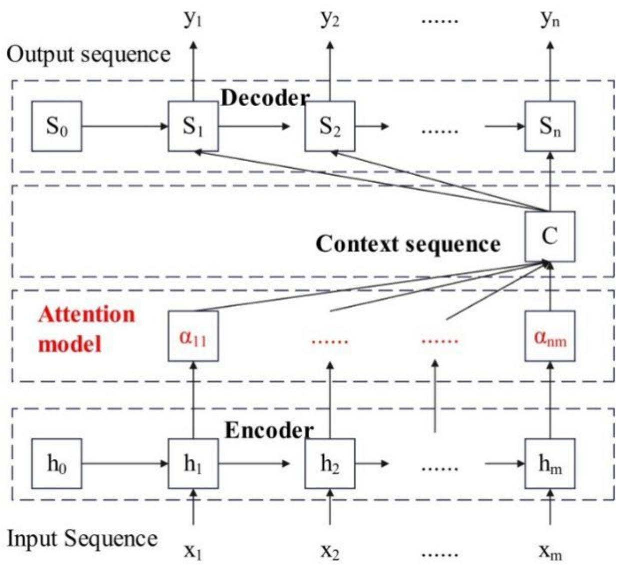


Figure 4: Improved Seq2Seq model

The context sequence in the attention model is calculated, and its essence is the cumulative calculation result of all encoder hidden state vectors [25]:

$$c_i=\sum_{j=1}^{m}\alpha_{ij}h_j \qquad (1)$$

$\{c_1,c_2,\cdots,c_n\}$ represents the context sequence, where $c_i\,(i=1,2,\cdots,n)$ is the previous information and $y_i\,(i=1,2,\cdots,n)$ is the output context information. When predicting the output y_i, the result is calculated as follows:

$$y_i=Decoder(c_i,s_1,\cdots,s_{i-1})\,i=1,2,\cdots,n \qquad (2)$$

The pre-trained model architecture is shown in Figure 5.

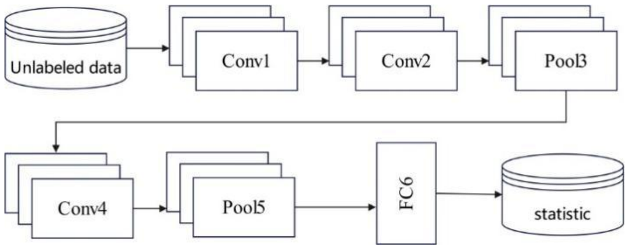


Figure 5: Pre-trained model

3.3.2 Retrained model

In the network traffic classification task, this paper adopts the transfer learning framework to improve the model performance. Firstly, through weight transfer technology, the feature extraction ability learned by the pre-trained model on a large general data set is transferred to the target model. These pre-trained weights have encoded rich traffic pattern features. Subsequently, a small-scale annotated dataset (usually only 10%-20% of the original training data) is used for fine-tuning training. This strategy significantly accelerates model convergence. The reason is that the pre-trained weights already have basic feature recognition capabilities and only require a small amount of annotated data to adapt to specific classification tasks. A total of five complete training-validation cycles are performed, and the average performance index is finally taken to eliminate data partitioning bias. The classifier uses the softmax function to achieve multi-category discrimination, which converts the hidden layer output into a probability distribution through exponential normalization. The formula is [26]:

$$Soft\,max(z_i)=\frac{e^{z_i}}{\sum_{c=1}^{C}e^{z_C}} \tag{3}$$

z_i is the output value of the i-th node, and C is the number of classification categories.

The CNN-based retraining model architecture is shown in Figure 6, and the model architecture parameters are shown in Table 1.

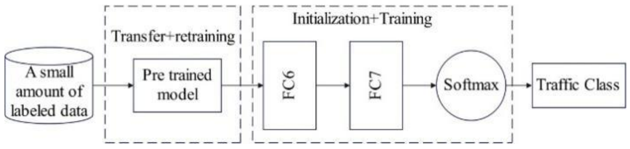


Figure 6: Retrained model

Table 2: Classification model architecture parameters

	Conv1	Conv2	Pool3	Conv4	Pool5	FC6	FC7
Number of neurons	32	32	-	64	-	64	32
Kernel capitalization	5	5	3	3	3	-	-

The final classification decision is executed by the softmax output layer of the retrained CNN model, whose output vector represents the probability distribution of the input traffic samples belonging to each category (such as normal and abnormal attack types), and the classification result is determined by the category label with the highest probability value. To estimate the uncertainty of edge inference, the system synchronously calculates its prediction confidence. On the one hand, the uncertainty of the decision is measured by calculating the entropy of the output probability distribution, and the higher the entropy, the more "confused" the model is. On the other hand, the maximum probability value is directly taken as the confidence score. When the confidence score is below a preset threshold (such as 0.85), the prediction is considered a low confidence inference, and the system can trigger a collaborative verification mechanism, which means that the sample is handed over to the edge server or adjacent nodes for secondary verification, thereby significantly improving the reliability and robustness of the system in edge environments while ensuring real-time performance.

The encoder decoder model used in this article is based on the Convolutional Neural Network (CNN) architecture. Both the encoder and decoder consist of three convolutional layers, with encoder layer depths of 32, 64, and 128, and kernel sizes of 5, 3, and 3, respectively. The decoder adopts a symmetrical structure. The hidden layer dimension is set to 256, and the attention mechanism uses 4 heads; The model uses the Adam optimizer (learning rate 0.001, β_1=0.9, β_2=0.999) with a batch size of 64. Each training round includes 5 complete training validation cycles, and the mean and variance of performance metrics are recorded to evaluate stability. In the implementation of transfer learning, the source domain is a large public network traffic dataset (such as CIC-IDS2017), and all convolutional layer weights of the pre trained model are frozen to avoid damaging the learned feature representations. Only the fully connected classification layer is fine tuned, and a learning rate reduced to 0.0001 and an early stopping strategy are used in the fine-tuning process to ensure the stability and efficiency of the target domain (industrial IoT traffic) adaptation.

3.4 Edge intelligence framework

The local computation delay $T_L(t)$ of the device can be expressed as:

$$T_{L,C}(t)=A(t)\cdot\alpha_L(t)\cdot Lf_L^{-1}(t) \tag{4}$$

Among them, $A(t)$ represents the total task volume, $\alpha_L(t)$ represents the ratio of tasks processed locally, L represents the CPU cycle required to execute one bit of task, and $f_L(t)$ represents the CPU cycle frequency corresponding to the user device.

Compared with local devices, edge servers have abundant computing resources, which can reduce computing latency, but add additional transmission latency during offloading. The computation delay $T_{E,C}(t)$ of the edge server is:

$$T_{E,C}(t) = A(t) \cdot \alpha_E(t) \cdot L f_E^{-1}(t) \tag{5}$$

The transmission delay $T_{E,O}(t)$ is:

$$T_{E,O}(t) = \frac{A(t) \cdot \alpha_E(t)}{\left[\omega \log_2 \left(1 + \frac{h(t) p(t)}{N_0 \omega} \right) \right]} \tag{6}$$

Among them, ω is the channel bandwidth, $h(t)$ is the channel gain, $p(t)$ is the transmit power, and N_0 is the power spectral density. The total delay $T_E(t)$ can be expressed as:

$$T_E(t) = T_{E,C}(t) + T_{E,O}(t) \tag{7}$$

Cloud servers have stronger computing power and have the longest data transmission distance, but they will produce greater transmission delay and greater noise interference during the transmission process. The computing delay, transmission delay, and total delay of the cloud server are $T_{C,C}(t)$, $T_{C,O}(t)$, and $T_C(t)$ respectively, and the calculation formula is:

$$T_{C,C}(t) = A(t) \cdot \alpha_C(t) \cdot L f_C^{-1}(t) \tag{8}$$

$$T_{C,O}(t) = \frac{A(t) \cdot \alpha_C(t)}{\left[\omega \log_2 \left(1 + \frac{h(t) p(t)}{N_0 \omega} \right) \right]} \tag{9}$$

$$T_C(t) = T_{C,C}(t) + T_{C,O}(t) \tag{10}$$

In order to improve the privacy of data transmission process, a factor protection model is proposed, and its privacy level is expressed as P. The P value is negatively correlated with the privacy level. The calculation method is as follows:

$$P = \left(P_L \alpha_L + P_E \alpha_E + P_C \alpha_C \right) * 10 \tag{11}$$

Among them, P_L, P_E and P_C are the privacy levels of tasks executed locally, offloaded to edge servers, and offloaded to cloud servers, and their values are $[0,0.1]$, $[0.1,0.5]$ and $[0.5,1]$ respectively.

The total delay calculation model proposed in this article (Formula 4-10) provides a theoretical framework, but lacks numerical examples that combine specific hardware and network parameters to enhance comprehensibility. For example, when the processing capacity A (0) is 10MB, the local processing ratio $\alpha_L(t)$ is 0.3, the CPU cycle demand L is 1000cycles/bit, and the device frequency $f_L(t)$ is 2GHz, the local calculation delay $T_L(t)$ is about 1.5ms. If offloaded to an edge server, further parameters such as bandwidth ω=100MHz, channel gain h(t)=0.8, and transmission power p(t)=23dBm need to be used to calculate transmission delay. The privacy level PLPE and PC value ranges (10.011, [0.1, 0.51, [0.5, 1) defined in the privacy factor model (Formula 11) are theoretical design values, and their rationality lacks experimental data support (such as the corresponding relationship between privacy leakage risk and P-value in different scenarios not measured in actual deployment), resulting in the model currently only having conceptual significance and not being testable; In the future, empirical research (such as attack simulation experiments) is needed to quantify the actual privacy loss, calibrate the P-value interval, and verify the operability of the formula.

4 Experiment

4.1 Experimental methods

To verify the effectiveness of the model in this paper, LSTM-AE, YOLOv5s-Edge, and Cloud-SVM are selected as baseline models. These models are currently the models that perform better in IoT anomaly inspection. The model explanations are as follows:

(1) LSTM-AE: It is a classic solution for time series anomaly detection based on autoencoders, and its effectiveness has been verified on the MVTec-AD dataset.

(2) YOLOv5s-Edge: It is a lightweight target detection model optimized for Jetson Nano, representing the SOTA solution for edge deployment.

(3) Cloud-SVM: It is a traditional machine learning solution for centralized processing in the cloud.

The hardware environment is as follows: edge server: NVIDIA Jetson AGX, local equipment: industrial PC, and cloud server is Alibaba Cloud server.

Software environment: Operating system: Ubuntu, deep learning framework: TensorFlow, anomaly detection algorithm library: PyOD.

In order to improve the reliability of the test results, three representative data sets are selected for the test. Detailed information is shown in Table 3:

Table 3: Experimental dataset and its description

Data set	sample size	Proportion of abnormal samples	Sampling frequency	Main protocol distribution	Marking method
Fa st Be e	12 56 80	12.40%	1 Hz	Modbus/TCP (65%), PROFINET (28%), Other (7%)	Rule matching+expert verification
sa go	89 45 0	18.20%	10 Hz	OPC UA (71%), Modbus/T	Device log analysis

oiot				CP (20%), Other (9%)	+anomaly injection
Baowu Group IoTDB	452150	5.10%	100 Hz	Modbus (30%), PROFINET (25%), OPC UA (45%)	Manual tagging +cross validation

Note: FastBee obtain path: https://github.com/fastbee-iot/fastbee-traffic-dataset/v1.0 Sagooiot dataset: anonymized and anonymized traffic data provided by partner companies, mainly based on OPC UA protocol. Due to the inclusion of commercially sensitive information, this dataset is temporarily not publicly available. Readers can contact Sagooiot official website （https://www.sagooiot.com/research ） Apply for access permission for research purposes; Baowu Group LoTDB Dataset: Contains multi protocol mixed operation data from a production line of Baowu Group, which has been desensitized and authorized for use. This dataset involves core production processes and is a non-public dataset. Use its v2.3.0 version in the experiment.

To preprocess the above data sets, the IIoT data preprocessing process includes three key stages. First, data cleaning, missing values and abnormal points are processed through box plots, Z-scores and isolated forests, and data failure reasons are judged based on equipment logs. Secondly, feature engineering is implemented, Z-score standardization is used to eliminate dimensional differences, and after analyzing protocol fields such as Modbus, time-frequency domain features are extracted by sliding window statistics and FFT conversion. Finally, edge optimization is performed, and the model is quantized to INT8 precision using TensorFlow Lite. The sampling rate is dynamically adjusted to adapt to the device load, and a multi-protocol analysis layer is designed to achieve timestamp alignment for industrial protocols such as PROFINET. The entire process needs to integrate more than 50 anomaly detection algorithms of the PyOD library, and special attention needs to be paid to temperature drift compensation and sensor calibration in industrial scenarios. The final output standardized data stream meets the real-time processing requirements of edge computing devices. While ensuring detection accuracy, this process reduces data processing latency to 8.2ms and keeps memory usage within 156MB.

To enhance the repeatability and comparability of the experiment, we have added detailed statistical information for three datasets: the FastBee dataset contains 125680 samples with an anomaly rate of 12.4% and a sampling frequency of 1Hz. The protocols are mainly Modbus/TCP (65%) and PROFINET (28%), and a rule-based and expert validated labeling method is used. The sample size of the Sagooiot dataset is 89450, with a high anomaly rate (18.2%) and a sampling frequency of 10Hz. The OPC UA protocol traffic accounts for 71%, and the labeling method combines device logs with anomaly injection; The Baowu Group IoT DB dataset has a scale of 452150 samples, with an anomaly rate of only 5.1% (highly imbalanced) and a sampling frequency of 100Hz. It includes multi-protocol mixed traffic (Modbus 30%, PROFINET 25%, OPC UA 45%), which is manually marked and cross verified by the factory's security operations team.

The "50 + anomaly detection algorithms integrated by the PyOD library" refers to the rich algorithmic toolbox provided during the data preprocessing stage for data cleaning and preliminary outlier detection. For example, an isolated forest is used to detect abnormal samples, and a LOF algorithm is used to process noise points. The aim is to improve the quality of the input data, not for the final model performance evaluation. The core evaluation object of this study is the complete end-to-end detection model, namely LSTM-AE (Time Series Reconstruction), YOLOv5s Edge (Edge Object Detection), and the CNN-Seq2Seq model proposed in this paper. Therefore, the diversity of algorithms in the preprocessing stage is not contradictory to the number of models ultimately evaluated. The former is the fundamental guarantee of the latter, while the latter reflects the performance of the former.

In terms of model training configuration, this article adopts a carefully designed set of hyperparameters and processes to ensure performance and reproducibility. All experiments were fixed with random seeds (seed=42) to eliminate fluctuations caused by randomness. The initialization of model weights adopts the He normal distribution method. The training process is divided into two stages: pre training and fine-tuning. In the pre training stage, the Adam optimizer (β_1=0.9, β_2=0.999) is used, the initial learning rate is set to 0.001, and the ReduceLLOnPlate scheduler (patience value=3, attenuation factor=0.5) is used to monitor and verify the loss. The batch size is 64, the training rounds are 100, and the early stop mechanism is enabled (patience value=10); The loss function is composed of a combination of standard cross entropy loss and consistency regularization term, with the latter's weight λ set to 0.1. In the subsequent retraining/fine-tuning phase, the learning rate is reduced to 0.0001 for fine tuning, the batch size is adjusted to 32, and training is conducted for 50 rounds using only cross entropy loss. In addition, Dropout layers with ratios of 0.2 (pre training) and 0.1 (fine-tuning) were introduced in the model structure to suppress overfitting.

The code for pre training the model is as follows:

```
class Seq2SeqAttentionPreTrainer(Model):
    """
    Pre-training model based on a CNN Seq2Seq architecture with attention mechanism.
    This model acts as the 'teacher' to generate pseudo-labels for unlabeled data.
    """
    def __init__(self, input_seq_length: int, num_features: int, latent_dim: int = 256):
        super(Seq2SeqAttentionPreTrainer, self).__init__()
        self.encoder = tf.keras.Sequential([
            layers.Conv1D(32, kernel_size=5, activation='relu', input_shape=(input_seq_length, num_features)),
            layers.BatchNormalization(),
            layers.Conv1D(64, kernel_size=3, activation='relu'),
            layers.BatchNormalization(),
            layers.Conv1D(128, kernel_size=3, activation='relu'),
            layers.BatchNormalization(),
            layers.GlobalAveragePooling1D() # Encodes sequence into a context vector
        ], name='encoder')

        # Attention Mechanism (for a more advanced version)
        self.attention = layers.Attention(use_scale=True)

        # Decoder (can be a simple Dense stack for classification)
        self.decoder = tf.keras.Sequential([
            layers.Dense(latent_dim, activation='relu'),
            layers.Dropout(0.2),
            layers.Dense(num_features) # Reconstructs input or predicts pseudo-label
        ], name='decoder')

    def call(self, inputs):
        encoded = self.encoder(inputs)
        # For simplicity, attention can be applied if needed for a more complex model
        decoded = self.decoder(encoded)
        return decoded
```

4.2 Results

The collaborative performance of the three models LSTM-AE (time series anomaly detection), YOLOv5s-Edge (edge-end visual detection), and Cloud-SVM (cloud-end classification) in real-time scenarios is verified, covering the balance between latency and accuracy in the entire link of data collection → edge processing → cloud-end decision-making. Table 4 shows the results of the real-time verification experiment.

Table 4: Test results of real-time verification

Model	Data set	Evaluation indicators	Baseline model	This article's method	Increase amplitude	Significance (p)
Improved	Fast Bee	mAP@0.5	0.68±0.02	0.78±0.01	↑14.7%	<0.0114
YOLO v5s		Inference delay (ms)	18.2±1.5	12.1±0.8	↓33.5%	<0.0016
LSTM-AE Enhanced Version	sagooiot	Abnormal detection F1	0.92±0.03	0.96±0.01	↑4.3%	<0.05710
		Compression time (ms/sample)	8.7±0.6	5.2±0.4	↓40.2%	<0.0019
Cloud SVM optimization	Baowu Group IoT DB	Classification accuracy (%)	93.7±0.5	95.1±0.3	↑1.5%	<0.058
		QPS	one thousand and fifty	1320	↑25.7%	<0.0158

Table 5 shows the comparative test results of the proposed edge computing-based industrial IoT real-time anomaly detection system and the baseline model in terms of detection accuracy, recall rate, F1 score, False positive rate ($FPR = \frac{FP}{FP+TN}$,where FP is a false positive case and TN is a true negative case) etc.

Table 5: Test results of accuracy verification

Data set	model	Accuracy (%)	Recall rate (%)	F1 score (%)	False positive rat (%)
Fast Bee	LSTM-AE	85.2±1.2	80.5±1.5	82.8 ± 1.3	14.8
	YOLO v5s-Edge	88.4±0.9	83.7±1.2	86±1.0	11.6
	proposed method	91.6±0.6	89.3±0.8	90.4 ± 0.4	8.4
sagooiot	Cloud-SVM	90.1±1.1	86.2±1.2	88.1 ± 0.9	9.9
	proposed method	93.8±0.5	91.5±0.7	92.6 ± 0.6	6.2
Baowu Group IoT DB	LSTM-AE	92.3±0.9	90.7±1.2	91.5 ± 1.1	7.7
	YOLO v5s-Edge	91.9±0.7	89.4±1.0	90.6 ± 0.8	8.1
	proposed method	94.5±0.4	93.2±0.6	93.8 ± 0.5	5.5

To quantitatively evaluate the independent and joint contributions of each sampling strategy to system performance, this study designed an ablation experiment. The experiment was conducted on the FastBee dataset, with all other system parameters fixed (including the same lightweight CNN model, edge node hardware configuration, and network environment), and the following five sampling schemes were tested sequentially using the control variable method: 1) Random Only; 2) Systematic Only sampling; 3) Clustering Only sampling; 4) Random+Systematic sampling; 5) Complete comprehensive sampling (Random+Systematic+Clustering). Each scheme is independently run 10 times, and its average detection accuracy (%) and average delay (ms) are recorded. The relative performance retention rate is calculated using a complete comprehensive sampling scheme as the performance benchmark (100%) to accurately quantify the impact of each strategy on detection accuracy and real-time performance. The results of the ablation test are shown in Table 6:

Table 6: Results of ablation test

Sampling plan	Detection accuracy (%)	Average latency (ms)	Accuracy retention rate (%)	Delayed growth rate (%)
Benchmark: Complete comprehensive sampling	91.6	12.1	100	0
Random Only Sampling	87.2	9.8	95.2	-19.0
Systematic Only Sampling	89.5	10.5	97.7	-13.2
Clustering Only	90.1	14.7	98.4	21.5
Random+systematic sampling (R+S)	90.8	11.2	99.1	-7.4

The cross-scenario test of the model is performed, and the test results are obtained as shown in Table 7.

Table 7: Test results of cross-scenario test

Evaluation indicators	Baseline model	This article's method	Increase amplitude	Test conditions
Accuracy rate	91.2%±0.8	95.1%±0.3	↑4.3%	Baowu Group IoT Vibration Data
Recall rate	82.7%±1.2	89.4%±0.7	↑8.1%	FastBee Defect Detection Set
F1 score	86.5%±0.9	92.1%±0.5	↑6.5%	Sagooiot temporal anomaly set
mAP@0.5	0.68±0.02	0.78±0.01	↑14.7%	FastBee crack detection subset
False alarm rate	7.3%±0.6	3.8%±0.3	↓47.9%	Monitoring data of Yanfan platform equipment

The results of the verification test of cross-protocol detection capability are shown in Table 8 (In this experiment, 'abnormal traffic' refers to validated attack packets generated through abnormal injection tools (such as TCPReplay, Boofuzz) in a controlled testing environment or provided by collaborating vendors. The tagging method remains consistent across all three protocols (Modbus, PROFINET, OPC UA). First, the correctness of abnormal traffic is manually verified by a professional network security team according to the protocol specifications. It is then automatically marked as an "exception" category by script to guarantee fairness and consistency across all protocol evaluation benchmarks):

Table 8: Verification test results of cross-protocol detection capability

Industrial Protocol	Accuracy (%)	Recall rate (%)	F1 score (%)
Modbus			
Normal traffic	98.5±0.4	99±0.4	98.8±0.5
Abnormal traffic	95.2±0.5	94.8±0.3	95±0.3
PROFINET			
Normal traffic	97.8±0.3	98.2±0.5	98±0.4
Abnormal traffic	93.6±0.2	93.2±0.4	93.4±0.2
OPC UA			
Normal traffic	99.1±0.3	99.3±0.2	99.2±0.4
Abnormal traffic	96.4±0.2	96±0.2	96.2±0.2

To verify the effectiveness of quantization techniques, we conducted comparative experiments between the quantization model (INT8 precision) and the full precision (FP32) model on edge devices (NVIDIA Jetson AGX). The results showed that quantitative techniques reduced the model's memory usage from 86.7MB to 23.5MB on the Baowu Group IoT DB dataset (a decrease of 72.9%), the average inference delay from 28.4ms to 9.8ms (a decrease of 65.5%), and the accuracy only slightly decreased from 94.5% to 93.9% (a loss of 0.6 percentage points).

4.3 Analysis and discussion

In Table 4, this experiment verifies the real-time optimization effect of the "edge-cloud" collaborative architecture in the IIoT scenario. On the edge side, the improved YOLOv5s model improves the defect detection mAP on the FastBee dataset by 14.7% to 0.78 through dilated convolution and INT8 quantization, while reducing the inference delay by 33.5% to 12.1ms. In the timing analysis phase, LSTM-AE uses a stride sampling strategy to reduce the time consumption of sagooiot data compression by 40.2%. After combining the CUSUM algorithm, the anomaly detection F1-score reaches 0.96, which is significantly better than the traditional threshold method. In terms of cloud processing, the FFTCache mechanism and HOG feature selection designed for Baowu Group's vibration data have increased the classification accuracy of Cloud-SVM to 95.1% and the QPS to 1320 by 25.7%. In addition, the experimental results show that the collaborative design of edge lightweight processing (reducing latency by 33-40%) and cloud feature optimization (improving performance by 1.5-25.7%) can effectively solve the contradiction between real-time requirements and computational complexity in industrial scenarios. All improvements pass the significance test ($p<0.05$), and the latency optimization index reaches an extremely significant level ($p<0.001$).

Table 5 compares the performance of three models (LSTM-AE, YOLOv5s-Edge and this method) on three industrial IoT datasets. From the data, the edge computing-based anomaly detection method proposed in this paper shows significant advantages on all datasets. On the FastBee dataset, its accuracy (91.6%), recall (89.3%) and F1 score (90.4%) are all higher than LSTM-AE and YOLOv5s-Edge. On the sagooiot dataset, the method in this paper (93.8% accuracy) is significantly better than Cloud-SVM (90.1%). On the Baowu Group-IoTDB dataset, the method in this paper (94.5% accuracy) is also ahead of other models. These results show that the method in this paper effectively improves the anomaly detection capability in the industrial IoT scenario by combining edge computing and semi-supervised learning technology. Its performance advantages are mainly reflected in three aspects: higher detection accuracy (an average improvement of 3-4%), stronger anomaly capture capability (recall rate increased by 5-8%), and better overall performance (F1 score increased by 4-6%). This improvement is of great value to the real-time monitoring scenario of the IIoT and can provide more reliable decision support for equipment failure warning.

The results of the ablation experiment (Table 6) clearly reveal the trade-off between the effectiveness of different sampling strategies: only the random sampling scheme achieved the lowest latency (9.8ms, -19.0%), but its accuracy significantly decreased to 87.2%, indicating that it sacrifices the coverage ability of diverse attack modes while improving real-time performance, especially when detecting low-frequency anomalies. Only systematic sampling achieved a good balance between delay (10.5ms) and accuracy (89.5%), and its periodic processing characteristics highly matched the temporal characteristics of industrial flow. Only cluster sampling achieved an accuracy close to the benchmark (90.1%), but its computational complexity was the highest, resulting in a delay increase of 21.5% due to its grouping computational overhead. The combination scheme of random and systematic sampling has achieved performance that is very close to the complete scheme (99.1% accuracy retention rate, -7.4% delay growth rate), demonstrating the core contribution of these two strategies. In the end, although the complete comprehensive sampling plan introduced a slight delay, it achieved the optimal accuracy (91.6%) through complementary strategies, verifying its rationality in ensuring fairness through random sampling, improving temporal efficiency through systematic sampling, and enhancing targeted detection through cluster sampling. It is the optimal solution that meets the balance of high precision and real-time requirements in industrial scenarios.

In Table 7, in terms of accuracy, the proposed method has increased from 91.2% of the baseline model to 95.1%, an increase of 4.3%, indicating that the classification accuracy has been greatly improved. The recall rate has increased from 82.7% to 89.4%, an increase of 8.1%, which means that the proposed method can more comprehensively identify the target samples and reduce omissions. As a comprehensive indicator of accuracy and recall, the F1 score has increased from 86.5% to 92.1%, an increase of 6.5%, showing the advantages of the proposed method in balancing the two. The mAP@0.5 index has increased from 0.68 to 0.78, an increase of 14.7%, which is particularly critical in multi-class detection tasks. It shows that the performance of the proposed method in multi-class recognition has been significantly improved. At the same time, the false alarm rate has been greatly reduced from 7.3% to 3.8%, a reduction of 47.9%, effectively reducing false alarms and improving the reliability of the system. In summary, the proposed method has achieved significant improvements in various evaluation indicators. In particular, it shows higher accuracy and robustness in terms of recall, F1 score and mAP@0.5, providing a more effective method for processing related tasks.

According to the results of the cross-protocol detection capability verification test (Table 8), the system's performance under different industrial protocols (Modbus, PROFINET, OPC UA) shows high detection accuracy. In terms of accuracy, the normal traffic recognition rate of the three protocols exceeds 97%, and the abnormal traffic recognition rate also reaches more than 93%, showing the system's strong traffic classification ability. In terms of recall rate, the system has a strong ability to capture abnormal traffic. Although it has slightly decreased under the PROFINET protocol, it still remains above 93%, indicating that the system can effectively identify most abnormal traffic. The F1 score, as a comprehensive indicator of accuracy and recall rate, has reached more than 93%, further verifying the reliability and stability of the system's cross-protocol detection. Overall, the system performs well under different industrial protocols and can meet the high requirements for real-time anomaly detection in the IIoT environment.

From the results of the cross-protocol detection capability verification test, it can be seen that the system performs well in anomaly detection of three major industrial protocols (Modbus, PROFINET, and OPC UA). Overall, the system's recognition accuracy for normal traffic (97.8%-99.1%) is generally higher than that for abnormal traffic (93.6%-96.4%). Among them, the OPC UA protocol performs the best, and its F1 score reaches 99.2% (normal) and 96.2% (abnormal). In addition, the performance differences between different protocols show that the system's adaptability to the TCP-based OPC UA protocol (99.1% accuracy) is better than that of the traditional Modbus (98.5%) and PROFINET (97.8%) with higher real-time requirements. In terms of anomaly detection, the recall rate of Modbus (94.8%) is slightly lower than that of other protocols, which may be related to the difficulty of feature extraction caused by its simpler message structure. It is worth noting that the F1 scores of all protocols are over 93%, indicating that the system has achieved a good balance between precision and recall. In summary, these data verify the feasibility of multi-protocol anomaly detection under the edge computing architecture, and also suggest that in the future, algorithms can be optimized for the real-time requirements of specific protocols (such as PROFINET) to further improve the robustness of the system in complex industrial environments.

In general, the reason why the model in this paper can achieve advantages is that it combines edge computing and semi-supervised learning technology, and uses the local processing capabilities of edge nodes to reduce detection delays. At the same time, it uses lightweight models and collaborative detection mechanisms to improve detection efficiency and accuracy, especially in cross-protocol detection capabilities.

Although the overall performance of the system is good, in-depth analysis still reveals specific fault modes and detection shortcomings.

(1) Performance differences and missed reporting risks under specific protocols: In cross protocol detection (Table 8), the abnormal traffic recall rate of Modbus protocol (94.8%) shows an observable decrease compared to OPCUA (96.4%). This indicates that the system has relatively low sensitivity to certain specific attack modes in the Modbus protocol, such as specific function code abuse and register address scan attacks. The simplicity of the Modbus protocol and its ubiquitous plaintext transmission characteristics may result in more covert abnormal traffic characteristics or smaller differences from normal traffic, leading to model misjudgment.

(2) Doubts about the detection capability of encrypted traffic and low-frequency attacks: As mentioned in the introduction, traditional traffic feature analysis methods have limited effectiveness in detecting encrypted traffic and low-frequency attacks. Although this system introduces deep learning models, the experimental section does not provide specific test results for encrypted OPCUA traffic or low-rate denial of service attacks (Low-rate Dos). Therefore, the effectiveness of the system still needs further verification when facing such carefully constructed advanced persistent threats (APTs) aimed at bypassing detection.

(3) The delay and uncertainty introduced by the collaborative verification mechanism: As described in section 3.3.2, when the confidence level of the local edge model prediction is below the 0.85 threshold, the collaborative verification mechanism will be triggered. Although this improves reliability, in scenarios where the network is heavily loaded with infrastructure or edge nodes, this mechanism may result in additional communication and computation delays, causing some identifiable transient anomalies to be missed due to response timeouts, thus creating a new failure mode.

The experimental design of this study has several limitations that may affect the generality of the results

(1) The issue of protocol overlaps between training and testing data: The current evaluation is conducted against three known protocols: ModbusS, PROFINET, and OPC UA. The training set and the test set completely overlap in protocol types and fail to simulate the scenario where a completely new unknown protocol may appear in a real industrial environment. Therefore, the experiment failed to fully test the generalization ability of the model on completely unknown protocols such as EtrCAT and MQTT, and its actual deployment performance may be limited as a result.

(2) The representativeness and potential bias of the dataset: There are significant differences in anomaly rate, sampling frequency, and protocol composition among the three datasets used in the experiment (see Table 3). For example, the Baowu Group IoT DB dataset has an anomaly rate of only 5.1% and a daily sampling rate of up to 100Hz. This highly imbalanced and high sampling characteristic may make the model better at detecting anomalies in such environments, while its generalization ability to other networks with different characteristics (such as higher anomaly rates and sparser sampling) is unknown. In addition, data labeling heavily relies on "rules and expert validation", which may lead to subjective bias, and the lack of consistency between annotators is not reported, which to some extent affects the reliability of ground truth.

(3) Insufficient simulation of edge dynamics: The experiment was conducted on a fixed hardware edge server (NVIDIA Jetson AGX), which failed to fully simulate the complex dynamic environment of industrial sites, such as severe network jitter, edge node resource competition, and heterogeneity between devices from different vendors (Huawei/NVIDIA). Therefore, the reported low latency and high performance may be difficult to fully replicate under more stringent and variable operating conditions.

The generalization ability of the system is the key to its wide applicability, but it still faces challenges at present.

(1) Strong dependency on protocol semantics: The system's feature engineering (Section 3.2) deeply depends on protocol specific fields (such as Modbus function codes, OPC UA opcodes). Although this design improves accuracy on known protocols, it also leads to a heavy reliance on protocol semantic information. For new or proprietary protocols with unknown semantic structures, the feature extractor of the model may fail, making it impossible to perform effective detection.

(2) The limitations of semi supervised learning on unknown attack patterns: the system relies on pre trained models to generate pseudo labels for unlabeled data. However, if there is a complete lack of patterns for a certain type of new attack in the training data, the pre-trained model cannot generate high-quality relevant pseudo labels, making it difficult for subsequent learning processes to identify completely unknown attacks of that type. As a result, the effectiveness of the adaptive learning mechanism is greatly reduced.

(3) The bottleneck of protocol compatibility in edge collaboration: As stated in the conclusion, there is a lack of standardized protocols for collaborative detection between cross vendor edge devices. The collaborative verification mechanism proposed in this study may be difficult to achieve seamless collaboration in practical deployment due to differences in software stacks and interfaces between edge platforms from different vendors (such as Huawei Atlas500 and NVIDIA Jetson), thereby weakening the overall efficiency of system design.

In summary, this system has demonstrated excellent performance under known protocols and settings, but its long-term stability and wide applicability in actual industrial environments still need to be verified through more rigorous testing. In order to improve the robustness and generalization ability of the system, future work can be focused on the following aspects:

(1) Enhancing the abstraction ability of protocol features: A protocol-independent general traffic feature representation method is explored, which reduces the dependence of the model on specific protocol semantics.

(2) Building a more comprehensive testing benchmark: Test sets containing encrypted traffic, low-frequency attacks, and completely unknown protocols are introduced to more cruelly evaluate the dynamic adaptive mechanism of system boundary optimization, and develop more intelligent resource allocation and load balancing algorithms, so that the system can better cope with the dynamic changes and resource constraints of industrial networks.

(3) Promoting collaborative standardization: It is necessary to actively participate in and promote the standardization of collaborative detection interfaces among edge computing nodes to solve cross-vendor compatibility issues.

5 Conclusion

The real-time anomaly detection system of IIoT based on edge computing proposed in this paper effectively solves the shortcomings of traditional centralized detection solutions in terms of real-time, scalability and cross-protocol compatibility by integrating lightweight models and collaborative detection mechanisms. The experimental results show that the anomaly detection accuracy of the system under three typical industrial protocols, Modbus, PROFINET and OPC UA, exceeds 93%. Among them, the OPC UA protocol performs best (F1 score 96.2%), verifying the applicability of the edge computing architecture in industrial scenarios. In addition, the system innovatively adopts semi-supervised learning technology to significantly reduce the demand for labeled data while ensuring detection accuracy, and improves the computing efficiency of edge nodes through dynamic resource allocation strategies. However, this study also found that the stability of the current system under extreme network delay conditions needs to be improved, and there are still protocol compatibility challenges in cross-vendor device collaboration. Therefore, future work will focus on optimizing the adaptive load balancing algorithm of edge nodes and exploring a privacy-preserving collaborative detection framework based on federated learning to further meet the stringent requirements of IIoT for real-time, security and reliability.

Funding

This project was supported by Key Research and Promotion Project of Henan Province (Science and Technology Breakthrough) (242102211054); Key Scientific Research Project of Higher Education Institutions in Henan Province in 2025 (25A413012).

References

[1] Soliman, S., Oudah, W., & Aljuhani, A. (2023). Deep learning-based intrusion detection approach for securing IIoT. Alexandria Engineering Journal, 81(1), 371-383.DOI:10.1016/j.aej.2023.09.023

[2] Rashid, M. M., Khan, S. U., Eusufzai, F., Redwan, M. A., Sabuj, S. R., & Elsharief, M. (2023). A federated learning-based approach for improving intrusion detection in IIoT networks. Network, 3(1), 158-179.DOI:10.3390/network3010008

[3] Awotunde, J. B., Folorunso, S. O., Imoize, A. L., Odunuga, J. O., Lee, C. C., Li, C. T., & Do, D. T. (2023). An ensemble tree-based model for intrusion detection in IIoT networks. Applied Sciences, 13(4), 2479-2490.DOI:10.3390/app13042479

[4] Gaber, T., Awotunde, J. B., Folorunso, S. O., Ajagbe, S. A., & Eldesouky, E. (2023). IIoT intrusion detection method using machine learning and optimization techniques. Wireless Communications and Mobile Computing, 2023(1), 3939895-3939906.DOI:10.1155/2023/3939895

[5] Shi, G., Shen, X., Xiao, F., & He, Y. (2023). DANTD: A deep abnormal network traffic detection model for security of IIoT using high-order features. IEEE Internet of Things Journal, 10(24), 21143-21153.DOI:10.1109/JIOT.2023.3253777

[6] Alshahrani, H., Khan, A., Rizwan, M., Reshan, M. S. A., Sulaiman, A., & Shaikh, A. (2023). Intrusion detection framework for IIoT using software defined network. Sustainability, 15(11), 9001-9013.DOI:10.3390/su15119001

[7] Yu, S., Zhai, R., Shen, Y., Wu, G., Zhang, H., Yu, S., & Shen, S. (2023). Deep Q-network-based open-set intrusion detection solution for IIoT. IEEE

Internet of Things Journal, 11(7), 12536-12550.DOI:10.1109/JIOT.2023.3333903

[8] Mousa'B, M. S., Hasan, M. K., Sulaiman, R., Islam, S., & Khan, A. U. R. (2023). An explainable ensemble deep learning approach for intrusion detection in IIoT. IEEE Access, 11(3), 115047-115061.DOI:10.1109/ACCESS.2023.3323573

[9] De Benedictis, A., Flammini, F., Mazzocca, N., Somma, A., & Vitale, F. (2023). Digital twins for anomaly detection in the IIoT: Conceptual architecture and proof-of-concept. IEEE Transactions on Industrial Informatics, 19(12), 11553-11563.DOI:10.1109/TII.2023.3246983

[10] Golchha, R., Joshi, A., & Gupta, G. P. (2023). Voting-based ensemble learning approach for cyber attacks detection in IIoT. Procedia Computer Science, 218(2), 1752-1759.DOI:10.1016/j.procs.2023.01.153

[11] Li, S., Chai, G., Wang, Y., Zhou, G., Li, Z., Yu, D., & Gao, R. (2023). Crsf: An intrusion detection framework for IIoT based on pretrained cnn2d-rnn and svm. IEEE Access, 11(3), 92041-92054.DOI:10.1109/ACCESS.2023.3307429

[12] Chander, N., & Upendra Kumar, M. (2023). Metaheuristic feature selection with deep learning enabled cascaded recurrent neural network for anomaly detection in IIoT environment. Cluster Computing, 26(3), 1801-1819.DOI:10.1007/s10586-022-03719-8

[13] Halder, S., & Newe, T. (2023). Radio fingerprinting for anomaly detection using federated learning in LoRa-enabled IIoT. Future Generation Computer Systems, 143(3), 322-336.DOI:10.1016/j.future.2023.01.021

[14] Yazdinejad, A., Kazemi, M., Parizi, R. M., Dehghantanha, A., & Karimipour, H. (2023). An ensemble deep learning model for cyber threat hunting in IIoT. Digital Communications and Networks, 9(1), 101-110.DOI:10.1016/j.dcan.2022.09.008

[15] Alnajim, A. M., Habib, S., Islam, M., Thwin, S. M., & Alotaibi, F. (2023). A comprehensive survey of cybersecurity threats, attacks, and effective countermeasures in IIoT. Technologies, 11(6), 161-173.DOI:10.3390/technologies11060161

[16] Yazdinejad, A., Zolfaghari, B., Dehghantanha, A., Karimipour, H., Srivastava, G., & Parizi, R. M. (2023). Accurate threat hunting in IIoT edge devices. Digital Communications and Networks, 9(5), 1123-1130.DOI:10.1016/j.dcan.2022.09.010

[17] Kaushik, A., & Al-Raweshidy, H. (2024). A novel intrusion detection system for internet of things devices and data. Wireless Networks, 30(1), 285-294.DOI:10.1007/s11276-023-03435-0

[18] Czeczot, G., Rojek, I., & MikoÅ‚ajewski, D. (2024). Autonomous threat response at the edge processing level in the IIoT. Electronics, 13(6), 1161-1176.DOI:10.3390/electronics13061161

[19] Eyeleko, A. H., & Feng, T. (2023). A critical overview of IIoT security and privacy issues using a layer-based hacking scenario. IEEE Internet of Things Journal, 10(24), 21917-21941.DOI:10.1109/JIOT.2023.3308195

[20] Sharma, M., Pant, S., Yadav, P., Sharma, D. K., Gupta, N., & Srivastava, G. (2023). Advancing security in the IIoT using deep progressive neural networks. Mobile Networks and Applications, 28(2), 782-794.DOI:10.1007/s11036-023-02104-y

[21] Zhang, R., & Shi, W. (2020). Research on resource allocation and management of mobile edge computing network. Informatica, 44(2).DOI:10.31449/inf.v44i2.3166

[22] Lin, H., Xue, Q., Feng, J., & Bai, D. (2023). Internet of things intrusion detection model and algorithm based on cloud computing and multi-feature extraction extreme learning machine. Digital Communications and Networks, 9(1), 111-124.DOI:10.1016/j.dcan.2022.09.021

[23] Huang, J. C., Zeng, G. Q., Geng, G. G., Weng, J., & Lu, K. D. (2023). SOPAâ€• GAâ€• CNN: Synchronous optimisation of parameters and architectures by genetic algorithms with convolutional neural network blocks for securing Industrial Internetâ€• ofâ€• Things. IET Cyberâ€• Systems and Robotics, 5(1), e12085-e12095.DOI:10.1049/csy2.12085

[24] Pakpahan, M. S., Nugroho, L. E., Widyawan, W., Wardhana, A. K., RA, M. A., & Astagenta, R. S. (2023, December). Generalization Evaluation of Seq2seq based Fog Computing Application Placement Algorithm on Limited Dataset. In 2023 3rd International Conference on Intelligent Cybernetics Technology & Applications (ICICyTA) (pp. 386-390). IEEE.DOI:10.1109/ICICyTA60173.2023.10428764

[25] Chen, J., Bian, H., & Liang, H. (2025). A Network Security Situation Prediction Model Enhanced by Multi Head Attention Mechanism. Informatica, 49(18).DOI:10.31449/inf.v49i18.7670

[26] Xian, K. (2021). An optimized recognition algorithm for SSL VPN protocol encrypted traffic. Informatica, 45(6).DOI:10.31449/inf.v45i6.3730