Journal of Computers Vol. 36 No. 2, April 2025, pp. 189-203
https://doi.org/10.63367/19911599202504360201 3

Analysis of Abnormal Intrusion Detection Methods and Protection Mechanisms for Cybersecurity in Industrial Internet of Things

Meng-Ying Yang[1,2], Xin-Hong Hu[1], Shao-Ting Liu[1],
Jun-Xia Zhang[1,2*], and Zhan-Feng Yang[3]

[1] Hebei Institute of Mechanical and Electrical Technology,
Xingtai City 054000, Hebei Province, China
{mengying20180107, xinhong5817, shaoting6677, junxia9987}@163.com

[2] Xingtai City Smart Manufacturing and Digital Twin Technology Innovation Center,
Xingtai City 054000, Hebei Province, China

[3] State Grid Hebei Electric Power Co., Ltd. Nangong Power Supply Branch,
Xingtai City 055750, Hebei Province, China
my992844593@163.com

Received 29 March 2025; Revised 8 April 2025; Accepted 16 April 2025

Abstract. With the increase of transmitted data and the growing demand for artificial intelligence, more and more industrial devices and data are connected to the network and are rapidly transforming towards intelligence. New application scenarios gradually raise the requirements for user-end service latency and distributed data processing and storage capabilities, while network risks also increase. To enhance the automatic identification and defense against network malicious attacks, this paper first analyzes the shortcomings of the graph attention network model and the advantages of the multi-head attention mechanism, and builds a multi-head-graph attention network model through integration. To prevent abnormal intrusions, the defense process against malicious traffic is modeled as a static Bayesian game model, with the two players being malicious traffic and traffic blocking strategies. The traffic blocking strategies include traffic dropping and traffic redirection, both of which require the programmability of the network. By using a custom network dynamic adjustment method, the centralized network control's ability to program the network and decouple the control plane and data plane, the network security is enhanced through the method of rapid traffic diversion. Finally, the recognition ability of the identification model is tested using existing datasets, achieving ideal results.

Keywords: Industrial Internet of Things, edge computing, graph attention network, game theory

1 Introduction

The Industrial Internet of Things (IIoT) is a system composed of networked intelligent objects, cyber-physical assets, related general information technologies, and optional cloud or edge computing platforms, which can achieve real-time, intelligent, autonomous access, collection, analysis, communication, and exchange of process, product, or service information in industrial environments, thereby optimizing overall production value. Its value includes improving product or service delivery, enhancing productivity, reducing labor costs, lowering energy consumption, and shortening production cycles [1].

With the increase in transmitted data and the growing demand for artificial intelligence, more and more industrial devices and data are being connected to the network and rapidly transforming towards intelligence. New application scenarios are gradually raising requirements for user-end service latency and distributed data processing and storage capabilities. The traditional centralized core network architecture is increasingly unable to meet its development needs. These new computing-intensive and latency-sensitive services pose new challenges to network performance. Emerging mobile edge computing technology is the right approach to address computing speed and storage requirements. The Internet of Things is gradually moving from the initial stage of isolated applications to a rapid growth stage that encompasses multiple features such as multi-dimensional perception,

* Corresponding Author

digital twins, and intelligent interconnection [2].

To meet the diverse demands of Internet of Things (IoT) applications, various communication network standards have been continuously proposed and updated, such as the third, fourth, and fifth generation mobile communication standards, narrowband IoT communication standards, short-range wireless communication technologies, and low-power wide-area networks. The popularization of communication standards and the construction of infrastructure have driven the continuous expansion of the IoT industry scale, connecting hundreds of millions of devices to the IoT. However, the perception of interaction, production, and environment by IoT devices often involves critical information such as personal privacy, business secrets, and national security. As edge computing systems move computing tasks to the network edge, edge nodes can obtain users' private data. Once maliciously attacked, all privacy may be exposed. Moreover, the distributed deployment and heterogeneous characteristics of edge nodes provide opportunities for malicious edge nodes. Additionally, compared to cloud centers, the servers involved in edge computing generally have simple processors and operating devices and are resource-constrained. Therefore, large and complex security protection schemes in cloud computing are usually not applicable to edge computing systems [3].

In 2017, a Middle Eastern petrochemical company was attacked by TRITON, which disrupted the safety system and enabled attackers to manipulate and damage industrial processes. Also in 2017, a botnet named IoTReaper exploited a large number of unprotected IoT devices for attacks. This botnet infected devices through vulnerabilities and weaknesses and used them to launch distributed denial-of-service (DDoS) attacks. Therefore, how to build new systems, technologies, and solutions that meet the needs of the IoT on the basis of traditional security mechanisms is the key to ensuring the security of the IoT. The emerging cross-domain technologies and blockchain technology have demonstrated numerous advantages in aspects such as privacy protection, attack identification, and defense decision-making [4]. Therefore, the work done in this paper on the security protection of industrial Internet of Things (IoT) networks by integrating emerging technologies is as follows:

Firstly, the importance of security in current industrial IoT is analyzed, and then the methods for monitoring malicious attacks are summarized and discussed from the perspective of monitoring network malicious attacks.

Secondly, in terms of malicious attack identification, a graph attention network is introduced. Considering that the conventional graph attention network only uses the conventional single-head attention mechanism when extracting features with the graph attention network, the model's expression ability and generalization ability are poor, and it lacks the ability to resist noise and outliers in the input. Therefore, this paper introduces how to integrate the multi-head attention mechanism, and models the sensor's dependency relationship as a graph structure by setting node embeddings for sensors and learning the graph structure.

Finally, the malicious traffic defense process is modeled as a static Bayesian game model, with the two players being malicious traffic and traffic blocking strategies. The traffic blocking strategies include traffic dropping and traffic redirection. The prerequisites for traffic dropping and traffic redirection are the programmability of the network, the use of custom network dynamic adjustment methods, the centralized network control's ability to program the network, and the decoupling of the control plane and data plane, to enhance network security through the method of rapid traffic diversion.

After the above process, implementing malicious traffic detection and blocking for the traffic data entering and leaving the industrial Internet of Things can enhance the security index of the industrial Internet of Things, which is of great significance for maintaining the security, stability and sustainable development of the industrial Internet of Things.

2 Related Work

This article conducts research from two aspects, namely network anomaly intrusion and the protection mechanism against intrusion. In terms of network intrusion anomaly detection, Zhendong Wang from Jiangxi University of Science and Technology compared the differences between the current Internet of Things (IoT) security and traditional system security in a review format. He classified the intrusion detection systems in detail from aspects such as detection technology, data sources, architecture, and working mode. Finally, starting from the dataset, he expounded on the current IoT intrusion detection systems based on machine learning [5].

Qixu Liu from the Chinese Academy of Sciences summarized and analyzed the achievements in IoT malicious code detection. By summarizing the existing research, he deeply discussed the problems existing in the current research on malicious code detection based on artificial intelligence. He proposed three possible development directions for future research in this field: combining large models to achieve IoT malicious code detection, im-

proving the security of detection models, and integrating zero-trust architecture [6].

Shengquan Liu from Xinjiang University proposed a fusion model DQN-LSTM based on deep reinforcement learning, which combines the spatial and temporal features of traffic data to conduct anomaly detection. Experimental research was conducted on the publicly available industrial control network natural gas plant dataset [7].

Shijie Jian from the Chinese Academy of Sciences proposed a novel deep neural network intrusion detection method based on the scenario of sparse abnormal sample data to address the issues of poor detection performance for massive traffic data, low model accuracy and F-measure values, and high processing costs for high-dimensional data. This method can effectively identify abnormal behaviors in imbalanced datasets, achieving detection accuracies of 99.06% and 99.16% and F-measures of 99.15% and 98.22% on two real-world typical datasets, respectively [8].

Yuzhang Jiang from Guizhou Normal University proposed a lightweight vehicle network intrusion detection model based on visualization and an improved MobileNet model by integrating vehicle Internet of Things with transfer learning. The attack traffic was visualized as color images, and then the images were expanded through bilinear interpolation to enhance the dataset and prevent model overfitting. The MobileNet was improved and the model was fine-tuned using transfer learning. The experimental results show that on the Raspberry Pi device with limited computing power, the test accuracy, precision, recall, and F1 value for the vehicle network traffic datasets Car-Hacking and OTIDS reached 100%, with average response times of 2.5ms and 2.9ms respectively, reducing the response time by at least 40% compared to classic deep learning models such as ResNet-18 [9].

In terms of the construction of abnormal intrusion prevention mechanisms, Chengfang Wu pointed out that industrial control networks have high requirements for data accuracy and timeliness, and traditional network intrusion security control strategies are difficult to be directly applied. Therefore, based on the 1DCNN-LSTM model, time series analysis and complete matching algorithms are adopted for optimization, thereby expanding the time series span of the unit data set and improving the training efficiency of the model system. Experimental results show that the optimized model has no significant difference in the total training time compared with the traditional model, but has a significant improvement in prediction accuracy, which is suitable for subsequent practical applications [10].

Kai Mou proposed a distributed network intrusion data autonomous defense method based on cluster information entropy. This method calculates the conditional probability formula, defines the attributes of the split data set based on the number of split points, and extracts the contribution degree of traffic features based on cluster information entropy to identify the multi-stage features of intrusion data. In addition, a new state transition rule is generated through an active fingerprint camouflage mechanism, achieving the design and verification of the autonomous defense method. Experimental results show that the average bandwidth utilization of this method is 14.276%, which is lower than that of traditional methods, demonstrating better performance [11].

Guangrui Liu proposed a general model update method that supports the filtering of polluted data for intelligent network intrusion detection systems. Firstly, the EdgeGAN algorithm was designed to accelerate the fitting process of the model's edge sample distribution by the generative adversarial network through fuzz testing. Secondly, the polluted sample subset was identified by calculating the mean square error (MSE) value between the new training samples and the original model, as well as the score of the updated model for the old edge samples. Subsequently, by allowing the model to learn from malicious edge samples, the influence of poisoning samples on the model performance was effectively suppressed, ensuring that the model could quickly recover after being poisoned. Finally, experimental verification based on five typical intelligent network intrusion detection systems demonstrated that the proposed update method was significantly effective in filtering pollution and repairing the model. Compared with the existing state-of-the-art methods, the new method increased the detection rate of poisoning samples by an average of 12.50% and improved the repair effect of poisoned models by an average of 6.38%. This method is applicable to protecting the update process of any common intelligent network intrusion detection system, significantly reducing the workload of manual sample screening, effectively lowering the cost of poisoning detection and model repair, and simultaneously enhancing the performance and robustness of the model [12].

In summary, this paper conducts research from two key aspects and summarizes the achievements outlined above. Section 3 introduces the intelligent detection method for malicious data, where the core model is comprised of three components: a data collection module, a detection module, and a response module. Section 4 addresses the issue of malicious intrusion detection by implementing traffic blocking strategies for both known and unknown types of malicious traffic, thereby enhancing the security of Industrial Internet of Things (IIoT) systems. Additionally, by introducing a custom network dynamic adjustment method and leveraging the programming capabilities of centralized network control alongside the decoupling mechanism between the control

plane and data plane, a rapid traffic diversion-based security enhancement solution was implemented, further improving network security performance.

3 Intelligent Detection Method for Malicious Data

The intrusion detection model mainly consists of a data collection module, a detection module and a response module. The data collection module collects relevant information data about the monitoring target according to the monitoring tasks issued by the user, aggregates and processes it to form a data source that conforms to the norms of the intrusion detection model, and then transmits the data source to the detection module. In the detection module, the intrusion detection model conducts anomaly detection on the received data source. After detecting abnormal data, the defense mechanism responds. The malicious data detection and framework are shown in Fig. 1.

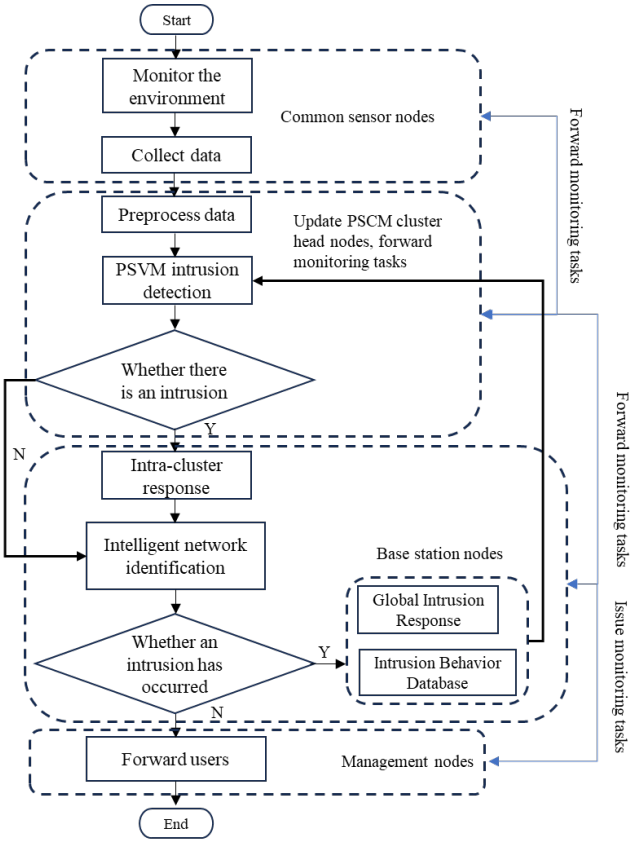

Fig. 1. Anomaly intrusion detection framework

3.1 The Construction of Malicious Data Detection Model

With the promotion of the attention mechanism in various fields, it has also been applied to the field of intelligent recognition. Veličković et al. proposed the graph attention network (GAT) based on the attention mechanism. When aggregating the features of neighboring nodes, this network achieves adaptive matching of weights for different neighbors through the attention mechanism, thereby generating the embedding representation of nodes in the next layer [13]. In addition, to stabilize the learning process of the attention mechanism, the multi-head attention mechanism is introduced to calculate the embedding representation of nodes and merged through concatenation or averaging.

The multi-head attention mechanism is mainly used to stabilize the learning process of self-attention. It calculates the weights of neighboring nodes to the central node from multiple features respectively, and then connects these features to obtain the output of the next layer. Through the multi-head attention mechanism, more hidden information can be learned, thereby improving the prediction accuracy of the model. Previous studies rarely used multi-head attention in long-term prediction, resulting in the model being unable to obtain more comprehensive feature information. To solve this problem, Zhao et al. proposed a multi-attention spatio-temporal graph network (MASTGN) for air pollution prediction. This network uses the data collected by atmospheric monitoring stations to construct a graph and applies the attention mechanism to three features: channels, space, and time. Among them, the channel attention network is used to determine the interrelationship between the atmosphere, the spatial attention network is used to capture spatial correlation to measure the mutual influence between different nodes in the atmospheric system, and for temporal correlation, the network combines the attention mechanism with one-dimensional time convolution for processing time series to identify the correlation between different time steps. Finally, the prediction results are obtained through a multi-layer perceptron with two hidden layers [14]. Zheng et al. proposed the Graph Multi-Attention Network (GMAN) to predict traffic conditions at different locations on the road network graph. This network adopts an encoder-decoder architecture and applies spatial and temporal attention mechanisms to the encoding and decoding processes. It extracts spatio-temporal correlations through a gated fusion mechanism and introduces transformation attention to alleviate error propagation, thereby enhancing long-term prediction performance [15]. However, the aforementioned method only constructs a single graph, neglecting the dependencies between different graphs and thus failing to fully explore the spatial correlations among them. To address this issue, Jin et al. constructed four different types of graphs based on the varying relationships between nodes and proposed a Spatio-Temporal Multi-Attention Multi-Graph Convolutional Network (STM2CN) to predict the data collected by sensors in smart cities. Specifically, STM2CN captures the dynamic spatio-temporal correlations of sensor data through spatio-temporal attention and uses a weight attention module to explore the dependencies between different graphs, thereby fusing the weights of each graph to obtain the context relationships between nodes [16].

This section discusses a method for detecting abnormal network malicious behaviors, the multi-head graph attention network monitoring method. This method models the dependencies among sensors as a graph structure by setting node embeddings for sensors and learning the graph structure, and updates the model parameters through joint optimization. Finally, it determines the abnormality through abnormal scores [17]. The multi-head graph attention network method model aims to model the sensors in the Internet of Things system as a network. In the network, sensors are regarded as nodes, and the learned relationships between sensors are regarded as edges. The model consists of five parts:

1) Use node embedding vectors to flexibly capture the unique characteristics of each sensor. In a large Internet of Things (IoT) system, the data collected by various types of sensors is a multivariate time series. These sensors and their relationships are modeled as a graph, with each sensor defined as a node in the graph. Randomly initialize the node embedding vectors for each sensor to represent its inherent attributes.

2) Learn the graph structure representing the dependencies between sensors.

3) Use a multi-head graph attention network to obtain network information dimension features. The main advantage over a single-head attention network is that it can simultaneously capture diverse and complex dependencies between nodes in the graph. Each attention head can learn different weight distributions, focusing on different semantic or structural features of the node neighbors (such as local/global patterns, heterogeneous relationships, etc.). Finally, the multi-head features are concatenated or aggregated to enhance the robustness and expressiveness of the representation. This mechanism can effectively alleviate the potential bias of single-head attention and usually yields more comprehensive and stable high-order network feature representations in tasks such as node classification and link prediction.

4) Jointly optimize the model parameters using a prediction-based model and a reconstruction-based model. The parameters of both models are updated simultaneously during training. The former is used to predict the value at the next time stamp, while the latter is used to capture the data distribution of the entire time series.

5) Determine whether the system is abnormal based on whether the anomaly score exceeds the preset threshold. The prediction-based model and the reconstruction-based model respectively generate the predicted values of the multivariate time series and the reconstruction probability obtained by the reconstruction-based model. The final anomaly score balances their respective advantages, thereby significantly improving the effectiveness of anomaly detection.

The model framework is shown in Fig. 2.

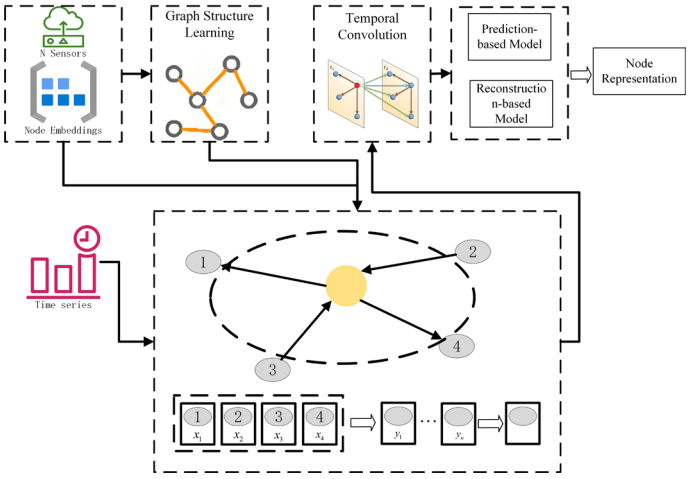

Fig. 2. Schematic diagram of the multi-head graph attention network structure

In the context of industrial Internet of Things (IoT), various types of sensors need to be configured, and each type of sensor will be equipped with a sufficient number based on usage requirements. These sensors form an interconnected network, and there are certain correlations among them. To capture the potential relationships among these sensors, this paper constructs a network structure diagram of the sensors and their relationships. The structure diagram includes various network nodes, and each sensor is placed at a node in the diagram. A random initial node embedding vector is assigned to each sensor to represent its inherent attributes. The node vectors of each sensor are expressed as follows:

$$n_i \in S^{\dim_i}, i \in [1, \cdots, m] \tag{1}$$

In the formula, n_i represents the node embedding vector of node i, $\dim_i$ is the dimension of node embedding vector n_i , m is the number of sensor nodes, and the similarity between node embedding vectors n_i reflects the data correlation of the data transmitted by sensors. The node network structure diagram composed of node embedding vectors is used as input for the learning of the sensor Internet of Things system and the recognition of the graph attention network.

In an Internet of Things system with a large number of sensors, there may be some prior information that can be used to determine the node dependency relationship. When such prior information exists, it can be described as a set of alternative relationships R_i for each sensor, each sensor is represented as i , that is, the set of neighboring sensors that may have a dependency relationship with the sensor:

$$R_i \subseteq [1, \cdots, m] \tag{2}$$

The default candidate neighbor node set of sensor node i includes all other sensor nodes in the system. Cosine similarity is used to calculate the similarity between the node embedding vector n_i of sensor node i and the node embedding vector n_i of its candidate sensor nodes $j \in R_i$:

$$sim_{j,i} = \frac{n_i n_j}{|n_i| \cdot |n_j|}, j \in R_i \tag{3}$$

Sort the calculated cosine similarities in descending order and select the top t nodes, that is, the top t nodes with the highest similarity to sensor node i , and connect each of them to node i with a directed edge. We can control the sparsity of the graph by specifying the size of t . When the sparsity is 1, it indicates that there is a directed edge between the candidate sensor and the sensor. If the sparsity is 0, it means that no directed edge exists [18].

3.2 Improvement of the Recognition Model

The multi-head attention network is used to capture the relationships among sensors, and a convolutional network is employed to calculate a threshold value ranging from 0 to 1 to control the importance of each attention head. The network structure is shown in Fig. 3.

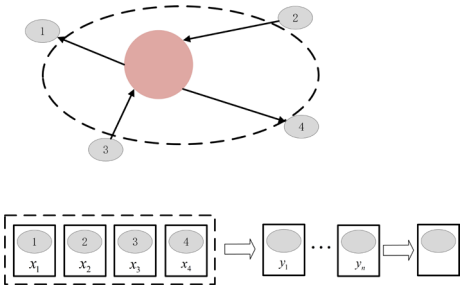

Fig. 3. Schematic diagram of the multi-head attention network structure

The signals input into neural networks are often single-phase modulated signals. To enable the neural network to better extract signal features, this paper proposes a mixed data augmentation algorithm (MDA) [19]. The original data is enhanced in both the time domain and the frequency domain. Time-domain enhancement is achieved by performing time-shift operations on the signal to change its position on the time axis, allowing the network to learn features under different time offsets. Self-disturbance data augmentation generates more diverse and representative training samples by applying various forms of perturbation or transformation to the input signal. Time-domain shift operations are implemented by translating the signal's time series.

3.3 Optimization of Recognition Models

This paper's recognition model combines a prediction-based model and a reconstruction-based model. The parameters of both models are updated simultaneously during the training process. The former is used to predict the value at the next timestamp, while the latter is used to capture the data distribution of the entire time series. The input of the reconstruction module and the prediction module is the output of the time convolutional network. The sum of the optimization objectives of the two models is defined as the loss function.

The reconstruction-based model captures the representation of the entire time series by learning to reconstruct the original input *input* based on the latent variable x. Here, we choose the variational autoencoder (VAE), which is unique in that it does not simply encode the input as a single definite point in the latent space but encodes it as a probability distribution in the latent space. Given an input *input*, it should be reconstructed by a conditional distribution. The optimization objective of the model is to find the best parameters that can best reconstruct the data distribution.

3.4 Threshold Setting for Malicious Data Detection

At each timestamp t , there are two inference results corresponding to the joint optimization objective. The prediction-based model and the reconstruction-based model respectively generate $Est(t)$ and p_i , where $Est(t)$ represents the predicted values of the multivariate time series, and p_i is the reconstruction probability obtained by the reconstruction-based model. The final anomaly score balances their respective advantages, thereby significantly improving the effect of anomaly detection. The anomaly score at each timestamp M is the sum of the anomaly scores of G time series. Specifically, the formula for the anomaly data threshold is:

$$\text{threshold} = \sum_{i=1}^{M} \frac{\left(\hat{Est}_i(t) - Est_i(t)\right)^2 + \lambda\left(1 - p_i\right)}{1+\lambda} \tag{4}$$

In the formula, $\left(\hat{Est}_i(t) - Est_i(t)\right)^2$ represents the squared error between the predicted value and the actual value, indicating the degree to which the sensor node deviates from the prediction. It is a hyperparameter selected for the test set to balance the error based on prediction and the probability based on reconstruction. $1 - p_i$ is the probability that sensor node i encounters an outlier according to the reconstruction model, and M is the total number of features. If the anomaly score at a timestamp exceeds the pre-set threshold, that timestamp is marked as "anomaly".

The pseudo-code for overall outlier detection is as follows:

```
# Import necessary libraries
import torch
import torch.nn as n
import torch.nn.functional as F
from torch_geometric.nn import GATConv
# Definition of Multi-head Graph Attention Network Model
class MultiHeadGAT(nn.Module):
    def __init__(self, num_features, num_classes, num_heads, hidden_dim, dropout):
        super(MultiHeadGAT, self).__init__()
        self.gat1 = GATConv(num_features, hidden_dim, heads=num_heads, dropout=-
dropout)
        self.gat2 = GATConv(hidden_dim * num_heads, num_classes, heads=1, drop-
out=dropout)
        self.dropout = dropout
    def forward(self, x, edge_index):
        x = F.dropout(x, p=self.dropout, training=self.training)
        x = F.elu(self.gat1(x, edge_index))
        x = F.dropout(x, p=self.dropout, training=self.training)
        x = self.gat2(x, edge_index)
        return F.log_softmax(x, dim=1)
# Define the anomaly detection function
def detect_anomaly(model, data, threshold):
    model.eval()
    with torch.no_grad():
        output = model(data.x, data.edge_index)
        anomaly_scores = torch.exp(output)  # Obtain abnormal scores
        max_scores = torch.max(anomaly_scores, dim=1).values  # Obtain the maximum
anomaly score for each node.
        anomalies = max_scores > threshold  # Determine whether it exceeds the
threshold.
    return anomalies
# Main function
def main():
    # Assuming that the data has been prepared.
    data = load_iiot_data()  # Load IIoT data
    num_features = data.num_features
    num_classes = 2  # Binary classification: normal or abnormal
    num_heads = 4  # The number of heads in the multi-head attention mechanism
    hidden_dim = 64  # Hidden layer dimension
    dropout = 0.5  # Dropoutprobability
    threshold = 0.8  # Abnormal scoring threshold
    # Initialize the model
    model = MultiHeadGAT(num_features, num_classes, num_heads, hidden_dim, drop-
out)
    # Train the model
```

```
    train_model(model, data)
    # Abnormal detection
    anomalies = detect_anomaly(model, data, threshold)
    # Output the anomaly detection results.
    print("Abnormal node:", anomalies.nonzero().squeeze().tolist())
# Run the main function.
if __name__ == "__main__":
    main()
```

4 Analysis of Network Intrusion Prevention Mechanism

Based on the malicious data identified in the previous section, a method of blocking malicious traffic of both known and unknown types is adopted to enhance the security coefficient of the industrial Internet of Things (IoT) system. Considering the incomplete information held by both the attacker and the defender in the industrial IoT environment and the simultaneous occurrence of their decision-making behaviors in logic, the process of defending against malicious traffic is modeled as a static Bayesian game model [20], with the two players being the malicious traffic and the traffic blocking strategies. The traffic blocking strategies include traffic dropping and traffic redirection. The prerequisites for traffic dropping and traffic redirection are the programmability of the network, the use of custom network dynamic adjustment methods, the centralized network control's ability to program the network, and the decoupling of the control plane and the data plane, to achieve enhanced network security through rapid traffic diversion.

A static Bayesian game is conducted between the two types of malicious traffic (known and unknown) and the two traffic blocking strategies (traffic dropping and traffic redirection), with the payoff values considering three factors: response cost, risk level, and the impact on current and future benefits. Through reasoning and proof, the equilibrium strategy is found [21]. The overall framework is shown in Fig. 4.

The overall structure is divided into two parts. The first part is the static Bayesian game model, which models the process of defending against malicious traffic in the industrial IoT as a static Bayesian game model, with the two players being the industrial IoT malicious traffic and the traffic blocking strategies. After establishing the game model, it is transformed, and finally, the Bayesian Nash equilibrium point is inferred and proved. The second part generates the corresponding OpenFlow rule entries for malicious traffic based on the dynamic network adjustment method to achieve the blocking purpose.

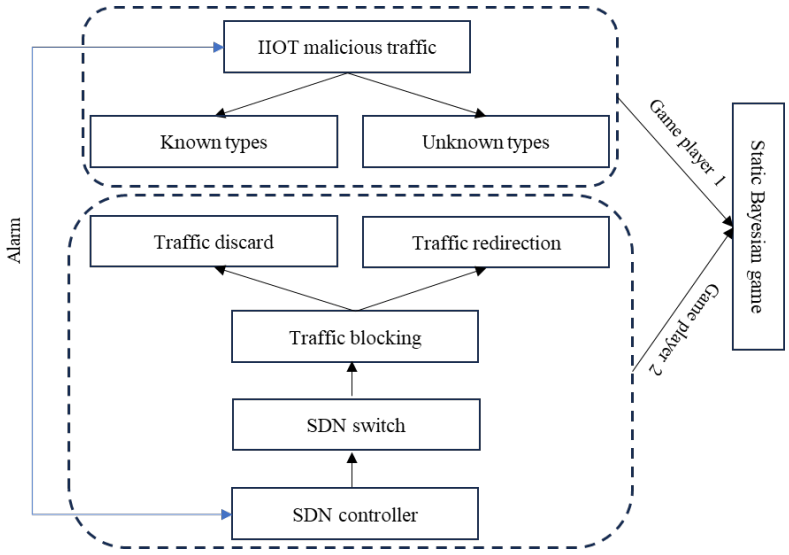

Fig. 4. Schematic diagram of the multi-head attention network structure

4.1 Model Establishment

The key factor of static Bayesian games is that each player knows their own payoff function, but cannot precisely understand the payoff functions of other players. During the game, to transform the uncertainty about payoffs into the uncertainty about types, $Type_i$ is used to represent the type of a player, $Space_i$ represents the type space of the player, and $Income_i$ represents the payoff of the player under the strategy combination [$Stra_1$, ..., $Stra_m$]. Each type corresponds to the possible situation of different payoff functions of the player, and its value is known only to the player themselves but not to other players, reflecting the feature of incomplete information in static Bayesian games. The expression of static Bayesian games is:

$$BG = \left\{ Stra_1, \cdots, Stra_m; Space_1, \cdots, Space_m; Income_1, \cdots, Income_m \right\} \tag{5}$$

The type of the player as the private information of the player determines the player's payoff function. By introducing the Harsanyi transformation to obtain the result of the dynamic game, the payoffs of each player are definite and known to each player, which is a complete information game. At this time, the original incomplete information game becomes a complete information game.

In this static Bayesian game, a strategy of each player is a complete plan on how they choose for each possible type of themselves, that is, for the static Bayesian game, it is a function of their various possible types. The function includes the actions selected by the player from their action space accordingly, and the function is denoted as $f_i(Type_i)$. If for any player i and each of their possible types $Type_i$, $f_i(Type_i)$, the chosen action $Stra_i$ can be satisfied, then the equilibrium strategy is expressed as:

$$\sum_{Stra_i}^{\max} \left\{ Income_i \left[f_1^* \left(Type_1 \right), \cdots, f_m^* \left(Type_m \right) \right] \right\} \tag{6}$$

Then the strategy function $f_i(Type_i)$ of the game is called a pure strategy Bayesian Nash equilibrium of the static Bayesian game BG. When a strategy combination of the players in a static Bayesian game is a Bayesian Nash equilibrium, no player wants to change their strategy. This is completely consistent with the connotation of Nash equilibrium.

4.2 Dynamic Network Customized Data Blocking Method

In the blocking strategy, the most crucial aspect is the packet matching process of the OpenFlow switch. The blocking principle diagram is shown in Fig. 5.

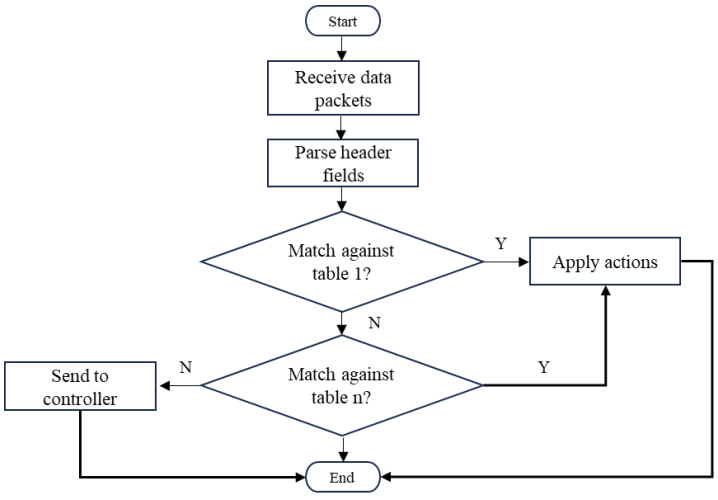

Fig. 5. The matching process of data packets by a switch

As can be seen from the figure, the flow table must contain a set of flow table entries for matching the fields or instructions of the packets entering or leaving the device. After receiving a data packet, the OpenFlow switch parses and matches it. It starts comparing from the first flow table entry and checks the flow entries according to their priority. If no matching option is found in the first flow table entry, it continues to match and check in sequence in the table pipeline. If a match is found in flow table N (flow table N being the last flow table entry), an action (forwarding, discarding, filtering, etc.) is executed. If no match is successful, the configuration on the Table-miss flow entry is executed.

```
// Define players: Malicious Traffic (Attacker) and Defense Strategy (Defender)
Players = {Attacker, Defender}
// Attacker types (e.g., low/high skill, specific attack patterns)
Type_Attacker = {Low_Skill, High_Skill}
// Defender's prior belief about attacker types (probability distribution)
Prior_Belief = {P(Low_Skill) = 0.6, P(High_Skill) = 0.4}
// Action spaces
Action_Attacker = {Probe, Exploit, Flood}
Action_Defender = {Monitor, Block_IP, Throttle}
// Utility functions based on actions and types
Function Compute_Utility(Attacker_Type, Action_A, Action_D):
    // Define cost/benefit matrices for each player
    If Attacker_Type == Low_Skill:
        U_Attacker = Lookup_Table_Low[Action_A][Action_D]
        U_Defender = -U_Attacker + Defense_Cost(Action_D)
    Else:
        U_Attacker = Lookup_Table_High[Action_A][Action_D]
        U_Defender = -U_Attacker + Defense_Cost(Action_D)
    Return (U_Attacker, U_Defender)
// Bayesian Nash Equilibrium (BNE) derivation
// Solve for optimal strategies where each type maximizes utility
BNE = Solve_Equilibrium(
    Players,
    Type_Attacker,
    Prior_Belief,
    Action_Attacker,
    Action_Defender,
    Compute_Utility
)
class SDNController:
    def __init__(self):
        self.flow_table = FlowTable()
        self.malicious_signatures = load_signatures("malicious_patterns.db")
    # Monitor traffic in real-time
    def monitor_traffic(self):
        while True:
            packet = capture_packet()  # Sniff network packet
            if self.is_malicious(packet):
                self.generate_openflow_rule(packet)
    # Detect malicious traffic using predefined signatures
    def is_malicious(self, packet):
        features = extract_features(packet)  # e.g., source IP, protocol, payload
        for sig in self.malicious_signatures:
            if match(features, sig):
                return True
        return False
    # Generate OpenFlow rule to block malicious flow
    def generate_openflow_rule(self, packet):
        rule = OpenFlowRule()
        # Define match fields
        rule.match.ipv4_src = packet.src_ip
        rule.match.ipv4_dst = packet.dst_ip
        rule.match.tcp_dst_port = packet.dst_port
```

```
            rule.match.protocol = packet.protocol
            # Set action to DROP and priority
            rule.action = "DROP"
            rule.priority = HIGH_PRIORITY  # Override lower-priority rules
            rule.timeout = 30  # Rule expiration time (seconds)
            # Deploy rule to switches
            self.flow_table.add_rule(rule)
            log("Rule deployed: " + rule.to_string())
        # Adaptive network reconfiguration
        def update_flow_table(self, new_rules):
            # Remove expired or outdated rules
            for rule in self.flow_table.expired_rules():
                self.flow_table.delete_rule(rule)
            # Add new rules with dynamic priority adjustment
            optimize_priorities(self.flow_table)
    # Main execution loop
    if __name__ == "__main__":
        controller = SDNController()
        controller.monitor_traffic()
```

5 Simulation Experiment Results and Analysis

The operating system used in this experiment is Windows 11, with an Intel(R) Core(TM) i7-10700, 8 cores / 16 threads, Turbo Boost up to 4.8 GHz, Turbo Boost Max Technology 3.0 processor and 16GB of memory. Python 3.9 is used throughout the entire solution. NumPy and Pandas are utilized for data processing, while TensorFlow and Keras frameworks are employed to build the model architecture. The Mininet simulation platform is adopted to establish and test the simulation tools for software-defined networks and network function virtualization. Iperf is used to generate three types of traffic in a random mode, namely normal traffic, known malicious traffic, and unknown malicious traffic. The generated traffic conforms to the characteristics of industrial Internet of Things traffic. Iman Almomani, Bassam Al-Kasasbeh, and Mousa AL-Akhras simulated the working environment of wireless sensor networks through the NS-2 simulator and used the most popular hierarchical routing protocol, the Low Energy Aware Cluster Hierarchy (LEACH) protocol. The resulting wireless sensor network dataset is called WSN-DS, which contains a total of 19 feature attributes, including 18 inherent attributes and 1 class label. The class label has five possible values, namely Normal and four abnormal behavior labels: Blackhole, Grayhole, Flooding, and Scheduling attacks. Black hole nodes discard all data packets from source nodes, blocking communication services with destination nodes. Black hole nodes will claim themselves as the most suitable relay nodes to the destination nodes, but during the data transmission process, they will discard all data packets from source nodes, causing transmission voids. Gray hole nodes also discard data packets from source nodes, but not all of them. They only discard certain types of data packets or randomly discard data packets and prevent forwarding data packets to the base station. Flooding nodes broadcast a large number of useless route request packets or send a large number of useless data packets, consuming the limited resources of WSN nodes and meaninglessly occupying bandwidth. Scheduling attacks occur in the initialization stage of the LEACH protocol, causing data conflicts between sensor nodes and ultimately resulting in data loss.The WSN-DS dataset has a total of 374,661 records, and the detailed distribution is shown in Table 1.

Table 1. Dataset category parameter

Data type	Quantity
Normal behavior	340066
Black hole attack	10049
Flood attack	3312
Grey hole attack	14596
Scheduling attack	6638

The recognition accuracy and recall rate of the data types in the dataset using the detection network proposed in this paper are shown in Fig. 6.

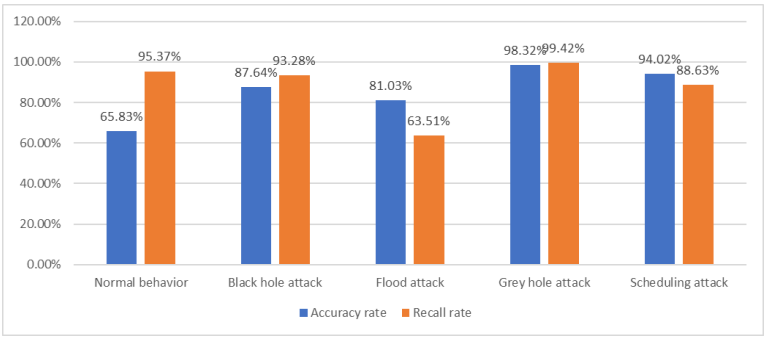

Fig. 6. Schematic diagram of multi-head graph attention network recognition results

Before adding multiple heads, the recognition accuracy and recall rate of the single-head graph attention network model for the five types of network transmission data were compared. The recognition accuracy and recall rate are shown in Fig. 7.

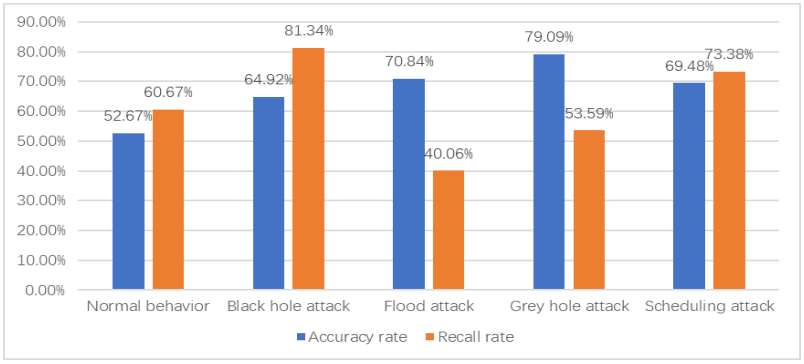

Fig. 7. Schematic diagram of Single-head graph attention network recognition results

It can be seen from the figure that the graph attention model with the multi-head attention mechanism has improved in both recognition accuracy and recall rate compared to the single-head graph attention model. Therefore, the network anomaly recognition method proposed in this paper can play its due role.

6 Conclusion

Anomaly detection, as an important tool for effectively identifying abnormal system behaviors in the industrial Internet of Things, is crucial for ensuring system security. It can detect potential problems in real-time or in advance, thereby avoiding situations that may cause significant economic losses and personal injuries. A multi-head graph attention network-based anomaly detection method is proposed. This method learns the graph structure representing the inter-influence relationship among sensors through a set of randomly initialized node embeddings. By inputting this graph structure and the data collected by sensors into a gated graph attention network, the spatial dependencies among sensors are captured. Next, the model parameters are updated by jointly optimizing the loss functions based on prediction and reconstruction. Finally, anomalies are determined through anomaly scores. In terms of defense, the process of defending against malicious traffic is modeled as a static Bayesian game model, with the two players being malicious traffic and traffic blocking strategies. Game theory is used to enhance the traffic blocking strategies, including traffic dropping and traffic redirection.

To further enhance the performance of anomaly detection methods in industrial Internet of Things (IoT), future research can be conducted in the following aspects:

First, the issue of data heterogeneity. In industrial IoT systems, data types are diverse, including traffic, time series, video, images, etc. Additionally, factors such as system architecture and external environments (such as weather conditions) also affect the system's operation. How to effectively train models in a heterogeneous data environment is an urgent and challenging research topic.

Second, the problem of imbalanced distribution of device anomaly data. Most public datasets mainly contain normal data samples, while the number of abnormal samples is extremely small. Many studies inject anomalies artificially or supplement abnormal data through accident reports and external sources. However, due to the fact that abnormal data only accounts for a small portion of the entire dataset, this imbalanced data distribution often leads to a decline in model performance. Therefore, how to handle data imbalance is an important direction for future research.

Third, the interpretability issue of graph neural networks in anomaly detection of industrial IoT systems. Similar to traditional deep learning models, the interpretability of graph neural networks has not been fully explored, which to some extent limits their wide application in industrial scenarios, especially in systems with strict requirements for model interpretability. Therefore, future research needs to deeply explore the interpretability of graph neural networks to improve their credibility and reliability in practical applications.

Fourth, although the intrusion detection model proposed in this paper shows good detection performance for the four types of intrusions in the current dataset, its detection scope still needs to be expanded to more types of wireless sensor network (WSN) intrusions to enhance the model's generalization ability. In addition, some algorithm parameters in the model are manually set. Although experiments have verified its good detection performance, there is still room for further optimization of these parameters to further improve the overall performance of the model.

Fifth, in the intrusion detection scheme proposed in this paper, the response strategy, as one of the key components, has a significant impact on the overall detection effect. In the next step, more comprehensive response strategies need to be developed based on the detection results to enhance the system's comprehensive defense capability. At the same time, although the intrusion detection scheme shows good performance in experiments, it has not been deployed and verified in real network environments, lacking credibility and feasibility assessment in practical applications. Therefore, future research needs to further verify and improve the scheme in real network environments to ensure its effectiveness in actual scenarios.

References

[1] R.-H. Yao, D. Li, J.-D. Zhu, Analysis of the Current Status and Development Trend of the Internet of Things, Electronics Quality (7)(2023) 109-114.

[2] J.-W. Zhang, X. Chen, S.-Y. Wang, Y.-J. Jing, J.-F. Song, Review of Application of Spatiotemporal Graph Neural Networks in Internet of Things, Computer Engineering and Applications 61(5)(2025) 43-54.

[3] X.-T. Yang, Y.-C. Liu, J.-M. Chang, Secure transmission method for IoT perceived data based on API service gateway technology, Wireless Internet Science and Technology 20(24)(2023) 119-121.

[4] M.-G. Liu, H.-M. Xiao, S. Shang, S.-Q. Ding, Research on the Security Analysis and Methods for the Internet of Things of Equipments, Instrumentation Technology (1)(2023) 46-49+68.

[5] Z.-D. Wang, L. Zhang, D.-H. Li, Survey of Intrusion Detection Systems for Internet of Things Based on Machine Learning, Computer Engineering and Applications 57(4)(2021) 18-27.

[6] Q.-X. Liu, J.-X. Liu, Z. Jin, X.-Y. Liu, J.-X. Xiao, Y.-H. Chen, H.-W. Zhu, Y.-K. Tan, Survey of Artificial Intelligence Based IoT Malware Detection, Journal of Computer Research and Development 60(10)(2023) 2234-2254.

[7] S.-Q. Liu, B. Liu, Research on intrusion detection in industrial networks based on deep reinforcement learning, Journal of Northeast Normal University (Natural Science Edition) 56(1)(2024) 80-86.

[8] S.-J. Jian, Y. Liu, B. Jiang, Z.-G. Lu, Y.-L. Liu, B.-X. Liu, Network Intrusion Detection Using Cluster Oversampling and Auto-Encoder, Journal of Cyber Security 8(6)(2023) 121-134.

[9] Y.-C. Jiang, Y. Xu, K.-Z. Li, Q.-K. Qin, S.-C. Zhang, Lightweight In-Vehicle Network Intrusion Detection Method Based on Deep Learning, Computer Engineering and Applications 59(22)(2023) 284-292.

[10] C.-F. Wu, X. Lyu, W.-D. Sun, X.-Y. Rong, Intrusion security protection method of industrial control network based on time series perfect matching algorithm, China High and New Technology (13)(2024) 35-37.

[11] K. Mu, K.-Y. Zhao, Self defense method of distributed network intrusion data based on cluster information entropy, Journal of Baoji University of Arts and Sciences(Natural Science Edition) 43(4)(2023) 15-18.

[12] G.-R. Liu, W.-Z. Zhang, X.-J. Li, Data Contamination Defense Method for Intelligent Network Intrusion DetectionSystems Based on Edge Examples, Journal of Computer Research and Development 59(10)(2022) 2348-2361.

[13] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, Y. Bengio, Graph Attention Networks, in: Proc. International Conference on Learning Representations, 2018.
[14] P.-J. Zhao, K. Zettsu, K. Mastgn, Multi-attention spatio-temporal graph networks for air pollution prediction, in: Proc. Proceedings of the 2020 IEEE International Conference on Big Data, 2020.
[15] C.-P. Zheng, X.-L. Fan, C. Wang, J.-Z. Qi, GMAN: A Graph Multi-Attention Network for Traffic Prediction, in: Proc. Proceedings of the AAAI Conference on Artificial Intelligence, 2020.
[16] Z.-L. Jin, J. Xu, R.-Q. Huang, W. Shao, X. Xiao, STM2CN: A Multi-graph Attention-based Framework for Sensor Data Prediction in Smart Cities, in: Proc. of the 2022 International Joint Conference on Neural Networks, 2022.
[17] X.-X. Liang, M.-M. Xia, Y.-Y. He, T. Liang, Traffic Flow Prediction Based on Spatio-Temporal Multi-head Graph Attention Network, Acta Electronica Sinica 52(2)(2024) 500-509.
[18] X.-Y. Fan, T.-T. Fu, N.-I. Listopad, P. Liu, S. Garg, M.-M. Hassan, Utilizing correlation in space and time: Anomaly detection for Industrial Internet of Things (IIoT) via spatiotemporal gated graph attention network, Alexandria Engineering Journal 106(2024) 560-570.
[19] J.-L. Zhang, S.-F. Zhang, Research on time domain enhancement algorithm based on adaptive Kalman filter, Information Technology (8)(2016) 9-13.
[20] H. Zhang, S.-G. Shen, X.-J. Wu, Q.-Y. Cao, WSN malware infection model based on cellular automaton and static Bayesian game, Telecommunications Science 35(6)(2019) 60-69.
[21] R.-N. Ma, E.-N. Zhang, G. Wang, Y.-F. Ma, J. Wong, Network Defense Decision-making Method Based on Improved Evolutionary Game Model, Journal of Electronics & Information Technology 45(6)(2023) 1970-1980.