Research on Intrusion Detection Models with Multi-Head Attention Mechanism Based on Graph Neural Network

Baiping Sun, Hong Dai*, Jiwang Sun, Xi Wei

***Abstract*—The swift expansion and diversification of Internet of Things (IoT) devices have heightened the risk of IoT-related attacks. This research offers an intrusion detection model, the Edge Multi-Head GAT-Graph Sample and Aggregate (EMG-GraphSAGE) model, to tackle the issues presented by the extensive and diverse data environment in IoT scenarios. The model utilizes a Multi-Head Graph Attention Mechanism (MHGAM) derived from GraphSAGE to improve message-passing efficiency among nodes and edges. Furthermore, the training procedure incorporates the Fast Gradient Sign Method (FGSM). FGSM is utilized in training to produce adversarial samples, facilitating the assessment of model robustness. Unlike conventional graph neural network, the proposed model dynamically allocates weights to node and edge attributes using a multi-head graph attention method. This approach efficiently consolidates node characteristics, adaptively modifies edge weights, merges node and edge features, and classifies them to elucidate intricate heterogeneous network relationships. The model's efficacy for intrusion detection is confirmed by trials conducted on the NF-BoT-IoT dataset. The experimental findings demonstrate that the suggested model significantly enhances essential metrics, including ACC, Precision, Recall, and F1 score, compared to conventional IoT intrusion detection methods. This method provides an efficient resolution for intrusion detection in IoT settings.**

***Index Terms*—Internet of Things, Intrusion Detection, GraphSAGE, Graph Neural Network, Multi-Head Graph Attention Mechanism**

I. Introduction

THE digital era and intelligence have been significantly propelled by the swift progress of the Internet of Things technology, which has hastened the digital transformation across several industries. The Internet of Things facilitates instantaneous connectivity and data exchange among devices through various smart gadgets and sensors [1]. It is extensively utilized in sectors such as industrial control, smart homes, and smart cities. The extensive implementation of IoT presents considerable security issues. The networking attributes and resource limitations of IoT devices render them particularly susceptible to hackers. IoT intrusion detection has become an essential strategy for safeguarding network by promptly detecting potential threats and reducing dangers to devices [2].

The intricate nature of IoT data structures and their unique non-Euclidean traits make conventional intrusion detection techniques, which depend on feature matching and statistical analysis, inadequate against ever-advanced attack strategies [3]. In this context, machine learning and deep learning-based IoT intrusion detection techniques have attracted considerable interest from researchers. GNN have become a central focus in IoT intrusion detection research due to their capacity to interpret complicated relational and non-Euclidean geographical data correctly. GNN utilizes a message-passing method to elucidate links between nodes, such as devices in graph-structured data, rendering it particularly effective for the relational modelling of IoT data [4-5].

Network intrusion detection often examines flow-based network data, including NetFlow. Flows are characterized by communication endpoints, including IP addresses, L4 port numbers, and L4 protocols, and are supplemented with flow fields that specify packet counts, byte counts, and flow duration. Flow data can be inherently depicted as a graph, with flow endpoints corresponding to graph nodes and network traffic illustrated as graph edges. Topological information and edge feature data are crucial for classifying network traffic and identifying attack flows [6-7].

GraphSAGE, a prominent graph neural network model, efficiently gathers neighbourhood information in IoT by sampling and aggregating adjacent nodes, including devices or connections. Nevertheless, the conventional GraphSAGE approach assigns uniform weights to aggregated neighbouring nodes, neglecting to highlight the significance of pivotal devices and connections in IoT [8]. In summary, the majority of contemporary IoT threat detection research prioritizes model performance, overlooking the influence of feature weight magnitude on classification efficacy. Moreover, model testing frequently demonstrates restricted dataset applicability and individual deep-learning techniques encounter difficulties in extracting adequate features [9]. The paper offers an IoT attack detection method utilizing the

Manuscript received November, 25, 2024; revised February 23, 2025. The research work was supported by the Fundamental Research Funds for the Liaoning Universities (No. LJ212410146058) and the Graduate student science and technology innovation project of University of Science and Technology Liaoning(No. LKDYC202423).

Baiping Sun is a postgraduate student of University of Science and Technology Liaoning, Anshan, Liaoning, CO 114051, China. (email: sunbaiping2023@163.com).

Hong Dai* is a professor in the School of Computer Science and Software Engineering, University of Science and Technology Liaoning, Anshan, Liaoning, CO 114051, China. (corresponding author to provide phone: +086-186-4226-8599; fax: 0412-5929818; e-mail: dear_red9@163.com).

Jiwang Sun is a postgraduate student of University of Science and Technology Liaoning, Anshan, Liaoning, CO 114051, China. (email: 1470573808@qq.com).

Xi Wei is a postgraduate student of University of Science and Technology Liaoning, Anshan, Liaoning, CO 114051, China. (email: 760861694@qq.com).

EMG-GraphSAGE model for feature extraction and training to overcome these challenges. The conventional GraphSAGE network, utilizing uniform aggregation among nodes, is augmented with a weighted aggregation approach grounded in a multi-head graph attention mechanism. Experimental results demonstrate that the proposed GraphSAGE model with a multi-head graph attention mechanism significantly outperforms traditional methods on the IoT dataset NF-BoT-IoT. The model's superior performance in anomaly detection is evident in the visualized confusion matrix results. The proposed model offers an effective solution for IoT intrusion detection and provides critical technical support for safeguarding IoT devices and applications.

II. Related Work

Network data can be depicted as graph structures, facilitating the utilization of GNN for intrusion detection. A GNN is a model engineered to analyse graph-structured data. GraphSAGE [10], a version of Graph Neural Network, overcomes the constraint of conventional Graph Neural Network in producing vector representations for previously unobserved nodes. It is extensively utilized for classification jobs and is improved by two principal techniques: local sampling and aggregation. Neighbourhood sampling facilitates dispersed training on extensive graph data, whereas neighbourhood aggregation consolidates information from adjacent nodes using several methodologies.

Xiao et al. [11] introduced a graph embedding technique for detecting intrusions in network flow. They transformed network flows into first-order and second-order graphs. First-order graphs encapsulate host-specific characteristics through IP addresses and port numbers, but second-order graphs derive global properties by employing source and destination IP addresses and ports. The embedding, together with raw characteristics, is utilized to train a random forest classifier for assault detection. Graph convolutional network (GCN) extends to the graph scenario with the convolution operation that is used on traditional data. The key idea of GCN is to learn a function that maps the graph information into a new representation. In this function, a node aggregates not only its features but also its neighbor's features in order to generate a new representation of the data. Cheng et al. [12] proposed Alert-GCN, a solution that aims at correlating alerts that belong to the same attack using GCN. Alert-GCN tackles this task as a node classification problem. It starts by building an alert graph using the alert information from neighbors, which is then fed into GCN to perform node classification.

Zhou et al. [13] proposed a GNN-based methodology for the automatic learning and identification of botnet tactics. This data-centric approach utilizes GNN to identify botnets, depending exclusively on graph structures for identification. Sarhan et al. [14] standardized the UNSW-NB15, BoT-IoT, and ToN-IoT dataset by transforming them into a cohesive NetFlow-based format with eight shared properties. The datasets NF-UNSW-NB15, NF-BoT-IoT, and NF-ToN-IoT are publicly available. Employing the ExtraTree integrated classifier, they attained F1 values of 0.85, 0.97, and 1.00 for binary classification and 0.98, 0.77, and 0.60 for multi-class classification.

Lo et al. [15] introduced the E-GraphSAGE model, which utilizes sampling and aggregation techniques for intrusion detection. This model produces enhanced graph representations by integrating edge and node data with an innovative edge vector representation approach, facilitating direct network flow detection without supplementary classifiers.

This study introduces EMG-GraphSAGE, an improved GraphSAGE model that integrates a multi-head graph attention mechanism. This methodology enhances essential feature detection in IoT by computing attention weights for adjacent nodes and edges. Furthermore, it utilizes FGSM-based adversarial training to improve intrusion detection efficacy.

III. Related Algorithm

A. Graph Neural Network

In intrusion detection, Graph Neural Network provide a promising avenue in deep learning, as they adeptly utilize the graph structures intrinsic to extensive IoT data, including social media network, knowledge graphs, and intricate texts. Graph architectures represent data by encapsulating information regarding objects, their interrelations, and the overarching framework. In contrast to other data formats, graph-structured data encompasses node and edge attributes, which accurately characterize interactions among devices in intricate IoT systems. Graph Neural Network excel in learning feature embedding while simultaneously capturing the spatial links inherent in graph structure. By transmitting messages between nodes and edges, GNN utilizes graph structures to efficiently learn and generalize data, resulting in low-dimensional embedding vectors [16].

Research on Graph Neural Network strongly relates to graph embedding, which encodes vertices as low-dimensional vectors while maintaining network topology and node properties for later machine-learning tasks. A primary function of GNN is to produce node embedding, which represents nodes as low-dimensional vectors while maintaining critical linkages and graph positions. Node embedding functions as crucial inputs for subsequent tasks, including node classification and clustering [17-18]. Graph Neural Network have garnered considerable attention due to their robust performance and the interpretability of node embedding in visualization tasks.

Graph Neural Network are a swiftly expanding domain within machine learning. Their power resides in using network structures intrinsic to extensive datasets from real-world areas such as social media, biology, and telecommunications [19]. Graph formats represent structural information by representing entities and their interrelations. In graph structures, entities are represented as nodes, while connections are illustrated as edges. In computer network, individual hosts (IP addresses) are depicted as graph nodes, whilst network flows are represented as graph edges [20].

B. The GraphSAGE Algorithm

The GraphSAGE algorithm, created by Hamilton et al., is a prominent graph neural network. GraphSAGE diminishes spatial and temporal complexity by consistently and randomly choosing a predetermined selection of nodes. This

technique, however, neglects graph structure, including node degree distribution and batch sizes. GraphSAGE operates on a graph *G(V, E)* where *V* represents the set of nodes and *E* is the set of edges. The node features of node *V* are represented as vectors X_V and the set of node feature vectors is $\{X_V, \ \forall v \in V\}$. A critical hyperparameter of GraphSAGE is the number of graph convolutional layers *k*, which determines the number of hops for aggregating node information in each iteration [21].

GraphSAGE emulates the convolution operation used in convolutional neural network by aggregating data from a node's local vicinity to derive its embedding [22]. The GraphSAGE approach presupposes a pre-trained model with static weight matrices and parameters for the aggregator function. The technique iteratively consolidates information from each node's neighbors and higher-order neighbors. During each iteration, the neighbourhood of the node is sampled, and the data from the sampled nodes is consolidated into a singular vector. At the *k*-th layer, the aggregated information for node *v*, based on its sampled neighbourhood *N(v)*, $h^k_{N(v)}$ is represented by Equation 1.

$$h^k_{N(v)} = AGG_k(\{h^{k-1}_u, \forall u \in N(v)\}) \tag{1}$$

Here, h^{k-1}_u denotes the embedding of node *u* in the previous layer. These embedding of all nodes *u* in the neighbourhood of *v* can be aggregated into the embedding of node *v* in the *k*-th layer.

The model framework, illustrated in Fig. 1, samples and aggregates the topology and node features from the *k*-hop neighborhood of each graph node. The GraphSAGE model supports various aggregation methods, such as averaging, pooling, or neural network-based approaches.

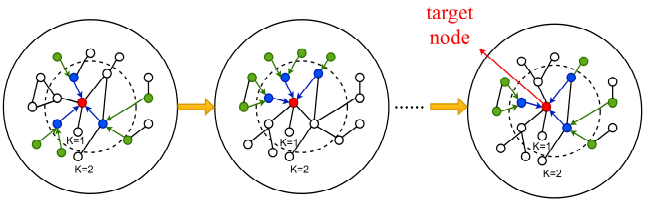

Fig. 1. GraphSAGE modelling framework

The aggregated vector representation of the sampled neighbourhood $h^k_{N(v)}$ is connected to the node vector representation of the previous layer h^{k-1}_v, and by applying the trainable parameter W^k of the model, the weight matrix can be trained, and after passing the result through the nonlinear activation function σ, the node *v*-vector representation of the *k*-th layer is computed, as shown in Equation 2.

$$h^k_v = \sigma\left(W^k \cdot CONCAT\left(h^{k-1}_v, h^k_{N(v)}\right)\right) \tag{2}$$

The final representation (embedding) of a node *v* is denoted as Z_v, which is essentially the node's embedding in the final *K* layer, as shown in Equation 3. For node classification purposes, Z_v can be passed through an S-neuron or Softmax layer.

$$Z_v = h^K_v, \quad \forall v \in V \tag{3}$$

C. Multi-Head Graph Attention Mechanism

The multi-head graph attention mechanism, a fundamental element of the Graph Attention Network (GAT), augments the model's expressiveness and resilience by acquiring feature weights of adjacent nodes from various viewpoints through numerous attention heads. In GAT, numerous attention heads autonomously calculate weights and subsequently either concatenate or average the results. This method allows the model to assimilate several facets of adjacent node characteristics, hence improving its generalization capacity [23]. Neighbour feature aggregation is dynamically modified by calculating correlation weights between nodes and their neighbouring nodes. The graph attention mechanism is adept at handling unstructured graph data, efficiently capturing intricate interactions among nodes in IoT systems. This study formulates an efficient IoT intrusion detection model by merging the previously stated techniques, providing a robust technical solution to mitigate IoT security threats.

The multi-head graph attention technique enhances the model's representational capability via several concurrent attention heads. Each head autonomously calculates relationships between nodes and their neighbors using attention weights, with outcomes generally amalgamated by concatenation or averaging. The multi-head approach allows the model to evaluate the significance of adjacent nodes from different viewpoints, so successfully representing the diversity and complexity of the graph [24].

The multi-head graph attention mechanism functions by independently performing all graph attention processes for each attention head: linear transformation, calculation of attention coefficients, and weighted aggregation of surrounding node information. Each head independently acquires the attention weights. The output features from all attention heads are ultimately amalgamated, typically by concatenation, to enhance the dimensionality of the resultant feature representation.

Input features include the node feature matrix *X* (feature vectors of each node) and the adjacency matrix *A* of the graph, or adjacency relations. The matrix *X* is linearly transformed to derive *Query*, *Key*, and *Value* representations, as shown in Equation 4.

$$Query = W_Q X, Key = W_K X, Value = W_V X \tag{4}$$

Here, W_Q, W_K, W_V represents the learned weight matrix.

The *AttentionScore* is computed using the *SoftmaxScore*. It quantifies the correlation between a given character and other characters at the same position, reflecting the "attention" paid to different positions. The *AttentionScore* is then scaled and normalized to obtain the *SoftmaxScore*, where d_k represents the dimension, as shown in Equation 5 and Equation 6.

$$AttentionScore = Query * Key \tag{5}$$

$$SoftmaxScore = softmax(\frac{AttentionScore}{\sqrt{d_k}}) \quad (6)$$

Finally, each Value vector is multiplied by *SoftmaxScore* to get a weighted sum, which is the first input *AttentionValue*, as shown in Equation 7.

$$AttentionValue = Value * SoftmaxScore \quad (7)$$

Numerous attention heads can acquire various facets of graph structure in separate subspaces, hence augmenting the model's capacity to discern intricate linkages. The outputs from multiple heads are concatenated, which mitigates overfitting or information loss that may arise from a singular attention head. Each attention head can capture unique forms of neighbouring information or concentrate on various sections of the network, thereby enhancing the graph representation. The multi-head attention mechanism is illustrated in Fig. 2.

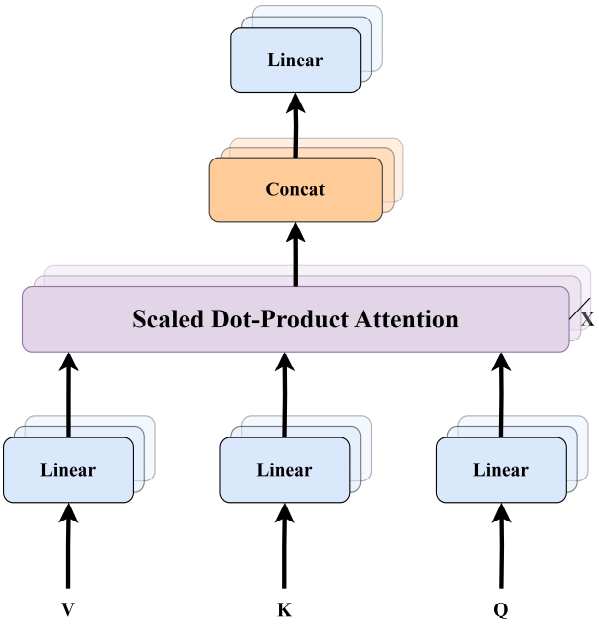

Fig. 2. Multi-attention mechanism

D. FGSM Match Training

The Fast Gradient Sign Method is a technique for producing adversarial samples [25]. It alters the input data by introducing minor perturbations derived from the gradient information of the neural network, resulting in erroneous model predictions. This approach is frequently employed in adversarial assault research to evaluate model robustness and to develop more resilient models. FGSM necessitates merely one backpropagation to calculate the gradient, succeeded by straightforward procedures, rendering it computationally more efficient than most other adversarial techniques. This efficiency renders it appropriate for practical large-scale testing. FGSM is considerably faster than approaches such as Projected Gradient Descent (PGD) because it calculates the gradient only once.

In the realm of IoT, security becomes a paramount concern. IoT devices generally possess constrained computational resources and decentralized data, rendering conventional security measures like encryption and access control less applicable in these contexts. FGSM attacks enable researchers to evaluate the resilience of machine learning models, including intrusion detection and anomaly detection systems utilized in IoT devices. These models must exhibit resilience against malicious assaults that manipulate the input data, potentially disrupting the model's functionality.

The core idea of the FGSM attack is to find a direction by calculating the gradient of the model's loss function concerning the input data such that the model's prediction changes when the input data is modified along this direction. Specifically, FGSM generates adversarial samples by computing the gradient of the loss function concerning the input and adjusting the input data along the direction of the sign of the gradient. Given an input sample x and corresponding label y, and the model's loss function $J(\theta, x, y)$, where θ is a parameter of the model, the FGSM's attack formula 8 is shown.

$$x' = x + \epsilon \cdot sign(\nabla_x J(\theta, x, y)) \quad (8)$$

Here, x' is the generated adversarial sample. x is the original input sample. ϵ is the size of the perturbation, such as the strength of the attack, which is usually a tiny positive number controlling the magnitude of the perturbation. $\nabla_x J(\theta, x, y)$ is the gradient of the loss function concerning the input. A sign denotes the sign operation on the gradient, which produces the sign of each element.

IV. IoT Intrusion Detection Model

A. Data Preprocessing

Data preparation is crucial for transforming raw NetFlow data into a graph structure for training and testing purposes. Initially, essential flow data is collected from the NetFlow dataset. The source and destination IP addresses, along with their respective port numbers, are amalgamated into a singular string to form a cohesive feature for further analysis and training. The original port columns are then eliminated, and attack labels are converted into a numerical format suitable for model processing. Feature normalization is implemented during the data cleansing step to guarantee that all features are uniformly scaled.

Furthermore, edge attributes and labels are incorporated to enhance the graph structure, allowing the model to discern more intricate links. The data is subsequently transformed into a graph by extracting pertinent fields and establishing graph edges. Target encoding of categorization features guarantees data consistency and model compliance. In graph creation, node, and edge features are initialized to furnish the model with foundational information for efficient learning. Training and test graphs are created to supply the dataset required for model training and assessment.

Finally, the dataset is split into 70% for training and 30% for testing, enabling validation on an independent dataset to assess the model's generalization ability. The data is converted into a GNN-compatible graph representation. This provides the model with a clean, standardized, and structured dataset for practical training and prediction.

B. Model Training

This study presents a multi-head graph attention mechanism derived from GraphSAGE to tackle the diversity and differing significance of device connections in the Internet of Things. In each information aggregation, attention weights are computed for each surrounding node to emphasize significant nodes and essential linkages. The multi-head attention mechanism establishes distinct linear transformation layers for each head to gather information regarding various relationships flexibly. Each brain individually calculates the attention score by interacting with source nodes and edge features, utilizing the RELU activation function. Each attention head aut the pseudo-code is provided in TABLE I.

TABLE I
EMG-GRAPHSAGE MODEL PSEUDO-CODE

Algorithm 1: EMG-GraphSAGE edge embedding

input: Graph $\mathcal{G}(\mathcal{V},\mathcal{E})$;
input edge features $\{\alpha_{uv}, \forall uv \in E\}$;
input node features $x_v = \{1,\ldots,1\}$;
depth K; the number of attention heads M;
weight matrices $W^k, \forall k \in \{1,\ldots,K\}$;
non-linearity σ;
differentiable aggregator functions AGG_k;
output: Edge embedding z_{uv}, $\forall uv \in \mathcal{E}$;

$h_v^0 \leftarrow x_v, \forall v \in \mathcal{V}$
for $k = 1$ to K **do**
  for $v \in V$ **do**
    $AttentionScore = Q * K$
    $\alpha_{uv}\_M = Attention(u, v, E, N, w^k\_M)$
    $\alpha_{uv}\_M = softmax(\alpha_{uv}\_M)$
    $\alpha_{uv}\_M = AGG_k(\alpha_{uv}\_M$ * (N of u and v, E)).
    $h_{uv}\_M = \sigma(w^k\_M * \alpha_{uv}\_M)$
    $h_w = concat(h_{uv}\_M\_1, ..., h_{uv}\_M\_n)$
    $z_{uv} = AGG_k(h_w)$
  end
end

Following data preparation, training and test graphs are generated, encompassing node and edge features along with their respective labels. The parameters of the graph neural network model, comprising weights and biases, are subsequently initialized and adjusted throughout training to reduce the model's prediction error.

Upon initialization, the model commences the training phase, during which it assimilates data patterns over numerous iterations. The model analyses the input graph data via its neural network layers to produce predictions. The model's predictions are juxtaposed with actual labels, and the value of the loss function is computed. The loss function quantifies prediction accuracy, and the objective is to minimize its value. The model's parameters are adjusted according to the gradient of the loss function by performing backpropagation across the network layers.

FGSM is employed to produce adversarial samples, followed by the evaluation of the model. Subsequently, the settings are refined using an optimization technique to minimize prediction error. Upon achieving satisfactory model performance, a final assessment is performed on the test set during training. Should the test results reach the anticipated accuracy, the model is preserved and can be utilized for real traffic classification jobs. The objective during the training process.

In the conventional training phase, the model assimilates the feature distribution of the network data. Following training, FGSM adversarial training is implemented to augment the model's resilience by producing adversarial samples, hence enhancing stability against malicious traffic. Following adversarial training, the EMG-GraphSAGE model undergoes testing to evaluate its performance and determine if its accuracy satisfies the anticipated criteria. Upon meeting the accuracy criteria, the trained EMG-GraphSAGE model is preserved, and the data is classified into normal and abnormal traffic for further multi-classification tasks. Suppose the accuracy fails to satisfy specific. The IoT intrusion detection framework is shown in Fig. 3.

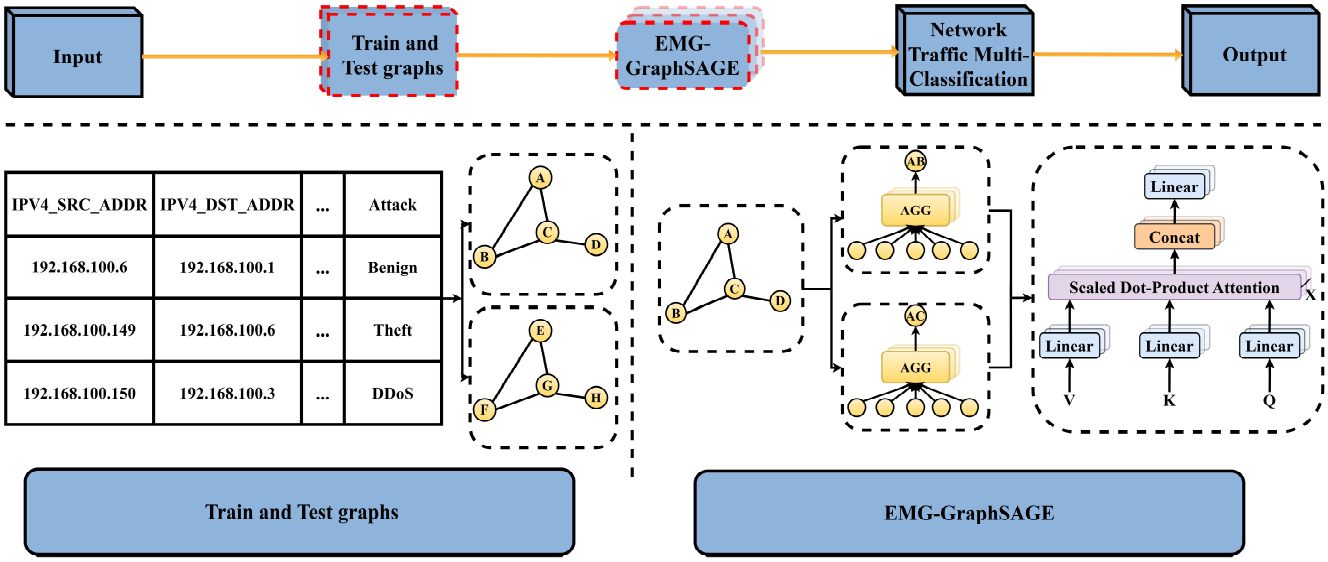

Fig. 3. IoT intrusion detection framework

V. Experimental Results And Analysis

A. Dataset

The NF-BoT-IoT dataset, created by Koroniotis et al. in 2019, is dedicated to IoT. The authors used the Node-RED tool to simulate IoT services, such as water stations, and generate the corresponding IoT traffic. Feature extraction was then performed using the Argus tool.

The dataset includes six attack types and 47 features, each labelled with the corresponding class. It contains 477 (0.01%) benign flows and 3,668,045 (99.99%) attack flows, totalling 3,668,522 flows. The lack of a standardized format and feature set across NIDS datasets complicates the comparison of ML-based network traffic classifiers and their ability to generalize to different network scenarios.

Sarhan et al. tackled this issue by providing NetFlow versions of the three aforementioned NIDS datasets. The authors converted the raw packet capture (PCAP) files of the original NIDS dataset into NetFlow format using the NPROBE tool. They selected 12 fields for extraction, resulting in a new variant: NF-BoT-IoT. The NF-BoT-IoT dataset contains 600,100 flows, of which 586,241 (97.69%) are attack flows and 13,859 (2.31%) are benign. Detailed data is presented in TABLE II.

TABLE II
INTRODUCTION TO THE NF-BOT-IOT DATASET

Classification	Sample(records)	Sample Proportion(%)
Benign	13859	2.31
Reconnaissance	470655	78.42
DDoS	56844	9.48
DoS	56833	9.47
Theft	1909	0.32
Total	**600100**	**100**

B. Environment Parameter Settings

The platform used for the experiments in this paper is Intel(R) Xeon(R) Silver 4214 CPU @ 2.20GHz、NVIDIA GeForce RTX 4090 Laptop GPU、24.0 GB GDDR6. The software environment is: Windows 11 operating system, Python3.10, Cuda11.8.

C. Model Parameter Settings

The EMG-GraphSAGE model requires appropriate parameter tuning to achieve controllable fitting. In this paper, multiple rounds of parameter tuning were performed to optimize model performance and achieve optimal experimental results. The model parameters are as follows: Hidden layer size = 64, Dropout rate = 0.2, Number of training rounds = 5000, Number of Attention Heads = 3, loss function = BCE, Activation Function = RELU, and Learning Rate = 0.001.

D. Evaluation Metrics

To assess the efficacy of the proposed model, we chose four metrics: Accuracy (ACC), Precision, Recall, and F1 score. ACC quantifies the ratio of accurate predictions to the total number of samples. Precision denotes the ratio of correctly predicted positive samples to the total expected positives, indicating the model's Accuracy. The Recall is the ratio of accurate positive samples identified as positive, indicating the model's coverage. This denotes the ratio of actual positive cases, indicating the model's accuracy. Recall quantifies the ratio of accurately predicted positive cases to the total actual positive cases, indicating the model's coverage. F1 is a statistic that integrates Precision and Recall to evaluate the efficacy of binary or multi-classification models. These metrics are derived from four counts: True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN). The equations for determining ACC, Precision, Recall, and F1 are as follows.

$$ACC = \frac{TP + TN}{TP + FP + TN + FN} \times 100\% \tag{9}$$

$$Precision = \frac{TP}{TP + FP} \times 100\% \tag{10}$$

$$Recall = \frac{TP}{TP + FN} \times 100\% \tag{11}$$

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \times 100\% \tag{12}$$

E. Analysis of Experimental Results

This study conducts a comparison experiment to evaluate the performance of the EMG-GraphSAGE model and assess the contribution of the multi-head graph attention mechanism to its overall performance. The experimental results are analyzed through comparison with other models and the role of the polygraphic attention mechanism was highlighted. Fig.4 displays the confusion matrix of the experimental results.

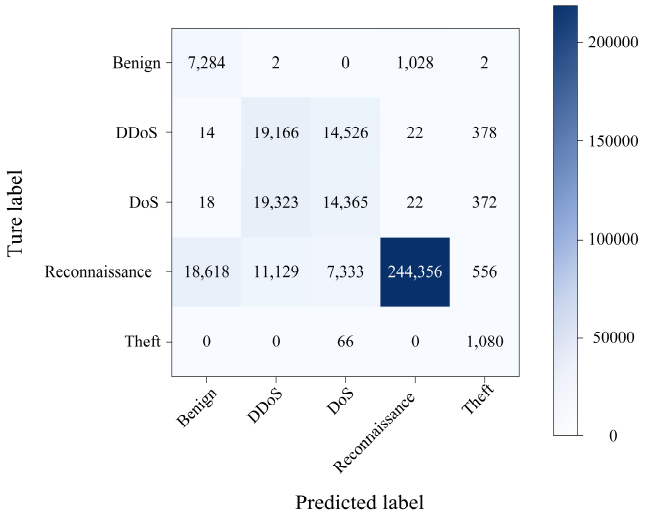

Fig. 4. Confusion matrix for NF-BoT-IoT dataset

This experiment evaluates the performance of three models: EMG-GraphSAGE, E-GraphSAGE, and Extra Tree Classifier. The results, measured by metrics such as Accuracy, Precision, Recall, and F1 score, highlight the advantages of the EMG-GraphSAGE model in IoT intrusion detection, particularly in attack detection. The EMG-GraphSAGE model surpasses all other models in the assessment criteria, with an Accuracy of 79.50%, Precision of 86.98%, Recall of 79.48%, and F1 score of 82.16% and the comparison results are presented in TABLE III.

TABLE III
COMPARATIVE TEST TABLE

Model	ACC(%)	Precision(%)	Recall(%)	F1(%)
EMG-GraphSAGE	79.50	**86.98**	79.48	**82.16**
E-GraphSAGE [15]	78.16	85.76	78.16	81.05
Extra-Tree Classifier[14]	74.32	79.10	73.58	77.36
MLP[26]	**80.87**	70.66	**80.87**	74.91

By introducing multi-head attention, the model can learn node relationships from various perspectives and subspaces rather than relying on a single attention mechanism. This enables the model to capture more comprehensive patterns and detect attack behaviors within the network. This mechanism enhances the model's robustness and optimizes the aggregation of node information, improving its ability to handle the diversity and complexity of network traffic. The multi-head attention mechanism allows the model to learn distinct features and relationships from each attention head, enabling each head to focus on different local regions or attack patterns in the network traffic. This multi-angle information fusion significantly enhances the model's ability to detect complex attacks and subtle anomalies, improving its representational learning capacity in intrusion detection tasks.

Compared to other models, the EMG-GraphSAGE optimized with multi-head attention outperforms both traditional machine learning models and other graph neural network models. The EMG-GraphSAGE with multi-head attention performs more effectively than the original GraphSAGE model in capturing complex node dependencies and handling various attack types. The multi-head mechanism allows the model to focus on multiple key network features simultaneously, particularly enabling the identification of low-frequency attack patterns hidden in complex data, thus strengthening network security.

EMG-GraphSAGE demonstrates significant advantages over traditional graph neural networks due to the addition of multi-head attention. In comparative experiments, the model not only improves Accuracy, Precision, Recall, and F1 score but also detects complex intrusion types more efficiently and accurately. This indicates that the multi-head attention mechanism enhances the model's ability to detect a variety of potential attacks in complex network traffic.

With the aid of multi-head attention, EMG-GraphSAGE better captures and integrates information between nodes, uncovering different attack features through parallel computation across multiple heads. As a result, the model processes node information more intricately and gains a broader perspective, improving the performance of the network intrusion detection system in real-world applications.

In conclusion, the EMG-GraphSAGE model, with the multi-head attention mechanism, significantly improves its performance in network traffic analysis and intrusion detection, outperforming traditional methods. The model can learn complex relationships between nodes from multiple perspectives, enhancing its ability to identify various attack types, particularly in complex network environments. This innovation advances the application of graph neural networks in network security, significantly improving their effectiveness in real-world environments.

This outcome illustrates that EMG-GraphSAGE attains elevated Accuracy in the classification test while proficiently balancing Precision and Recall. The elevated precision signifies a minimal false alarm rate for attack detection, whilst the high F1 score validates the model's proficient equilibrium between identifying attacks and regular traffic. The graph neural network's capacity to identify correlations among nodes and capture edge topology information allows EMG-GraphSAGE to excel in intricate intrusion detection tasks. In summary, EMG-GraphSAGE, utilizing a graph neural network, surpasses alternative models in the experiments, underscoring the benefits of graph neural network in network traffic classification and attack detection. Graph neural network surpass classic machine learning methods in their ability to capture edge topology in network data, hence improving model performance in intricate intrusion detection tasks.

VI. CONCLUSIONS AND FUTURE WORK

This paper introduces EMG-GraphSAGE, an IoT intrusion detection model utilizing GNN, which integrates both edge and node characteristics of the network flow graph to enhance the Accuracy of attack flow identification. This research concentrates on the application of EMG-GraphSAGE for detecting malicious network flows in IoT intrusion detection. Experimental findings on the NF-BoT-IoT dataset indicate that our models surpass current network intrusion detection models, attaining superior Accuracy, Precision, Recall, and F1 scores. Nevertheless, owing to the intricacy of the traffic data, some loss may transpire. In the future, we will prioritize minimising data loss and investigating the amalgamation of GNN with other deep learning algorithms to enhance intrusion detection efficacy.

This paper presents EMG-GraphSAGE, an IoT intrusion detection model utilizing a Graph Neural Network. It comprehensively utilizes both node and edge data in the network flow graph to markedly enhance attack flow detection accuracy. Experimental findings indicate that EMG-GraphSAGE surpasses current network intrusion detection models on the NF-BoT-IoT dataset, attaining superior performance in Accuracy, Precision, Recall, and F1 scores. The model has demonstrated resilience in managing intricate network traffic data, signifying its substantial promise in the IoT security sector.

The intricacy of network traffic is a considerable problem, frequently resulting in data loss that can obstruct the identification of severe attack patterns. Subsequent studies will emphasize the following optimization measures to resolve these challenges. Advanced data preprocessing and feature extraction strategies will be created in conjunction with the investigation of more refined data augmentation approaches to reduce information loss. Furthermore, the amalgamation of Graph Neural Network with sophisticated deep learning architectures, like Transformers and Convolutional Neural Network, will be undertaken to

develop hybrid models that capitalize on the distinct advantages of each framework.

The primary emphasis will be on diminishing the computing complexity of the model to enhance its real-time detection efficacy, hence rendering it more feasible for implementation in industrial environments. A thorough study of supplementary IoT traffic dataset will be conducted to assess the model's generalization ability and cross-domain resilience. To address the evolving nature of cyber threats, sophisticated methodologies such as adaptive learning and incremental learning will be explored, guaranteeing the model's efficacy against changing attack techniques.

These research efforts aim to advance the performance of EMG-GraphSAGE, delivering a robust and efficient solution for safeguarding IoT network against sophisticated security threats.

REFERENCES

[1] L. Shi, Q. Yang, L. Gao, and H. Ge, "An ensemble system for machine learning IoT intrusion detection based on enhanced artificial hummingbird algorithm," The Journal of Supercomputing, vol. 81, pp110-152, 2024.

[2] V. Holubenko, D. Gaspar, and R. Leal, "Autonomous intrusion detection for IoT: a decentralized and privacy-preserving approach," International Journal of Information Security, vol. 24, pp7-10, 2024.

[3] R. Bensaid, N. Labraoui, H. Saidi, "Securing fog-assisted IoT smart homes: a federated learning-based intrusion detection approach," Cluster Computing, vol. 28, pp50-67, 2024.

[4] A. Deshmukh, and K. Ravulakollu, "An Efficient CNN-Based Intrusion Detection System for IoT: Use Case Towards Cybersecurity," Technologies, vol. 12, pp203-223, 2024.

[5] K. Khalil, N. Afiza, N. Atiqah, G. Sulong, and M. Johar, "Smart Cities' Cybersecurity and IoT: Challenges and Future Research Directions," IAENG International Journal of Computer Science, vol. 51, no.7, pp725-737, 2024.

[6] N. Almiani, M. Anbar, S. Karuppayah, Y. Sanjalawe, H. Alrababah, F. Zwayed, "Feature Selection and 1DCNN-based DDOS Detection in Software-Defined Networking," Engineering Letters, vol. 32, no. 7, pp1529-1544, 2024.

[7] Q. Li, J. Huang, L. Sihan, and H. Chenze, "A Sustainable Data Encryption Storage and Processing Framework via Edge Computing-Driven IoT," Engineering Letters, vol. 32, no.7, pp1510-1520, 2024.

[8] Y. Xiao, Y. Feng, and K. Sakurai, "An Efficient Detection Mechanism of Network Intrusions in IoT Environments Using Autoencoder and Data Partitioning," Computers, vol. 13, pp269-286, 2024.

[9] S. Walling, and S. Lodh, "Enhancing IoT intrusion detection through machine learning with AN-SFS: a novel approach to high performing adaptive feature selection," Discover Internet of Things, vol. 4, pp16-25, 2024.

[10] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," Advances in neural information processing systems," pp26-36, 2017.

[11] Q. Xiao, J. Liu, Q. Wang, Z. Jiang, X. Wang, and Y. Yao, "Towards Network Anomaly Detection Using Graph Embedding, in International Conference on Computational Science," pp156-169,2020.

[12] Q. Cheng, C. Wu, and S. Zhou, "Discovering attack scenarios via intrusion alert correlation using graph convolutional networks," IEEE Communications Letters, vol. 25, no.5, pp1564–1567, 2021.

[13] J. Zhou, Z. Xu, A. Rush, and M. Yu, "Automating Botnet Detection with Graph Neural Networks," Machine Learning and Systems, pp1-8, 2020.

[14] M. Sarhan, S. Layeghy, N. Moustafa, and M. Portmann, "Netflow datasets for machine learning-based network intrusion detection systems," Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol. 371, pp117-135, 2020.

[15] Lo. Waiweng, S. Layeghy, M. Sarhan, M. Gallagher, and M. Portmann, "E-graphs: A graph neural network-based intrusion detection system for IoT," NOMS 2022-2022 IEEE/IFIP network operations and Management Symposium, pp1-9, 2022.

[16] L. Qixu, X. Juxin, T. Yaokang, W. Chengchun, H. Hao, and Z. Fangjiao, "Survey of industrial Internet traffic analysis technology," Communications Letters, vol. 45, pp221-237, 2024.

[17] Kan Hong, "Facial Expression Recognition Based on Anomaly Detection and Multispectral Imaging," IAENG International Journal of Computer Science, vol. 51, no.10, pp1627-1641, 2024.

[18] D. Miaolei, K. Yupei1, S. Chuanchuan1, X. Haihang, F. Shaoju, and Z.Xin, "Summary of network intrusion detection systems based on deep learning," computer application, pp1-13, 2024.

[19] S. Xuli, and L. Shifeng, "An Improved Intrusion Detection Method for GraphSAGE," Microelectronics and computers, pp1-10, 2024.

[20] S. Thangam, and S. Sibi Chakkaravarthy, "An Edge-enabled Virtual Honeypot Based Intrusion Detection System for Vehicle-to-Everything (V2X) Security using Machine Learning," IAENG International Journal of Computer Science, vol. 51, no.9, pp1374-1384, 2024.

[21] L. Congyu, Z. Lihui, and A. Yang, "Research on Intrusion Detection of Internet of Things Based on Graph Neural Network," Journal of North Central University, vol. 45, pp194-204, 2024.

[22] A. Alsoufi, M. Siraj, and A. Ghaleb, "Anomaly-Based Intrusion Detection Model Using Deep Learning for IoT Networks," Computer Modeling in Engineering & Sciences, vol. 141, pp823-845, 2024.

[23] H. Asgharzadeh, A. Ghaffari, and M. Masdari, "An Intrusion Detection System on The Internet of Things Using Deep Learning and Multi-objective Enhanced Gorilla Troops Optimizer," Journal of Bionic Engineering, vol. 21, pp2658-2684, 2024.

[24] L. Guoyu, W. Xueshun, and D. Jinyou, "Random Feature Graph Neural Network for Intrusion Detection in Internet of Things," Computer Engineering and Applications, vol. 60, pp264-273, 2024.

[25] S. Lupart, and S. Clinchant, "A Study on FGSM Adversarial Training for Neural Retrieval," Conference paper, pp484–492, 2023.

[26] W. Maximilian, L. Dieter, H. Andreas, S. Daniel, "Systematic Evaluation of Synthetic Data Augmentation for Multi-class NetFlow Traffic," Cryptography and Security, pp1-13, 2024.