Received: 2 April 2025 | Revised: 22 May 2025 | Accepted: 5 July 2025

DOI: 10.4218/etrij.2025-0152

ORIGINAL ARTICLE

ETRI Journal WILEY

Graph neural network-enhanced auxiliary classifier generative adversarial network framework for robust intrusion detection

Tianjing Wang | **Qi Liu** | **Hang Shen** | **Xiaokang Luo** | **Guangwei Bai**

College of Computer and Information Engineering (College of Artificial Intelligence), Nanjing Tech University, Nanjing, China

Correspondence
Hang Shen, College of Computer and Information Engineering (College of Artificial Intelligence), Nanjing Tech University, Nanjing, China.
Email: hshen@njtech.edu.cn

Funding information
National Natural Science Foundation of China, Grant/Award Numbers: 61501224, 61502230; Natural Science Foundation of Jiangsu Province, Grant/Award Number: BK20201357; Six Talent Peaks Project in Jiangsu Province, Grant/Award Number: RJFW-020

Abstract

To address the challenges of traffic diversity and data imbalance in network intrusion detection, we propose GraphACGAN, a novel detection framework that integrates auxiliary classifier generative adversarial networks (ACGANs) with graph neural networks (GNNs). In this architecture, the GNN is embedded in the ACGAN discriminator to exploit the latent graph structures inherent to network traffic, thereby improving the model's capacity to distinguish between benign and malicious behaviors. Simultaneously, the ACGAN generator was leveraged to synthesize minority-class attack traffic, effectively mitigating class imbalances and enhancing generalization. Comprehensive experiments conducted on three benchmark datasets, namely, NF-BoT-IoT-v2, NF-ToN-IoT-v2, and NF-UNSW-NB15-v2, demonstrate that GraphACGAN consistently outperforms all baselines, including E-GraphSAGE, GCN, ACGAN, LSTM, and KNN in terms of accuracy, precision, recall, and F_1-score. An evaluation of resource-limited computing platforms further demonstrates GraphACGAN's inference efficiency for real-world deployment.

KEYWORDS

auxiliary classifier generative adversarial network (ACGAN), graph neural network (GNN), multiclass classification, network intrusion detection

1 | INTRODUCTION

The internet has become the cornerstone of global connectivity, transforming the way individuals learn, work, communicate, and interact with their environments. Although its pervasive integration into modern life has facilitated unprecedented information exchange and digital innovation, it has also exposed users and organizations to an escalating landscape of cybersecurity threats [1]. The growing diversity and sophistication of cyberattacks, ranging from malware and phishing to advanced persistent threats, pose serious risks to personal privacy, critical infrastructure, and national security [2, 3]. Addressing these threats is a central concern in modern network systems.

Network intrusion detection systems (NIDS) serve as a critical line of defense by continuously monitoring traffic flows and identifying abnormal or malicious behaviors in real time [4]. Through early detection and response, an NIDS can trigger alerts or automatically initiate mitigation measures to prevent the spread of an attack. These systems are essential components of contemporary

cybersecurity architectures and have found applications in a wide range of domains, including military networks, industrial control systems, and the Internet of Things (IoT) [5, 6]. However, traditional NIDS approaches that rely on manually crafted rules or signature-based detection suffer from limited generalization capabilities and often fail to identify evolving threats.

Various deep learning models have been proposed to address the demand for real-time intrusion detection in high-throughput and dynamic network environments. [7], a directional long short-term memory (LSTM)-based framework called LSTM-cloud was developed to detect and mitigate distributed denial of service (DDoS) attacks in cloud-computing environments. By modeling sequential dependencies in traffic flows, the LSTM-cloud enables the rapid detection of burst attack patterns, which are common in high-volume cloud networks. Expanding upon this idea, subsequent research explored the use of bidirectional LSTM (BiLSTM) architectures, which process input sequences in both forward and backward directions. This bidirectional modeling allows for more comprehensive temporal context extraction, making BiLSTM models effective for both binary and multiclass classification of network traffic [8, 9]. Other researchers have investigated lightweight and hybrid models to enhance generalization and interpretability. For example, Mohy-Eddine and others [10] proposed an IoT-specific detection model based on *k*-nearest neighbors (KNN) and feature selection, whereas Zhao and others [11] combined gated recurrent units with ResNet modules to improve deep feature extraction for intrusion detection. Reinforcement learning has also been applied to the construction of adaptive intrusion detection agents capable of learning detection policies through environmental feedback [12]. In addition, deep neural networks [13] and deep autoencoders [14] have been employed to model complex feature distributions and reduce dimensionality in high-dimensional traffic datasets. These diverse approaches have collectively led to advanced research in this field. However, challenges remain in capturing relational structures and addressing data imbalances in realistic intrusion scenarios.

1.1 | Challenges and related works

Despite these advances, deep learning-based intrusion detection faces significant challenges owing to the ever-evolving nature of cyberattacks and imbalanced data distributions. These factors require robust, adaptive, and generalizable detection frameworks.

1.1.1 | Limited detection in Euclidean space

Most existing deep learning-based intrusion detection methods treat network traffic as independent samples in the Euclidean space and process each instance in isolation without accounting for the underlying relationships among them. However, in practice, network traffic exhibits rich structural dependencies such as shared communication endpoints, temporal proximity, and behavioral similarity, which are more naturally modeled in non-Euclidean graph structures. Ignoring these latent connections limits both the accuracy and efficiency of intrusion detection, particularly in large-scale network environments where batch detection becomes increasingly essential.

Graph neural networks (GNNs) [15], a class of deep learning models designed for unstructured and relational data, offer a promising solution to this problem. GNNs are powerful tools for capturing spatial and topological patterns in graph-structured data, making them particularly well-suited for network intrusion detection [16, 17]. GNNs can effectively model dependencies and contexts across traffic records by representing the network traffic as nodes and interactions as edges [18]. Early GNN-based approaches in the network security domain, such as those proposed by Xiao and others [19] and Cheng and others [20], modeled network traffic as graph structures and applied GNNs for analysis, achieving significant results in anomaly detection and botnet detection tasks, respectively. Recent studies have explored more sophisticated architectures, including ResACAG [21] with residual connections and attention mechanisms and FTG-Net-E [22] employing hierarchical ensemble learning with bagging and boosting strategies for DDoS detection.

Furthermore, some studies have applied graph convolutional networks (GCNs) [23] for intrusion detection, leveraging normalized adjacency matrices and convolutional operations to effectively aggregate neighborhood information. However, GCNs treat edges as homogeneous connections and neglect critical edge features in network traffic such as connection types, protocols, and interaction patterns. By contrast, Lo and others [24] proposed E-GraphSAGE, which incorporates both edge features and topological patterns into the graph-learning process, achieving improved detection performance for both benign and malicious traffic. Despite the potential to overcome the limitations of Euclidean-based methods by modeling traffic as structured graph data, these approaches still face challenges associated with class imbalance between common and rare attack types.

1.1.2 | Imbalanced network traffic data

In real-world network environments, malicious traffic typically accounts for only a small fraction of total network traffic, resulting in a significant class imbalance between benign and attack samples. Training deep learning models on such imbalanced datasets often leads to biased learning, where the model performs well on majority (benign) classes, but fails to detect minority (attack) instances accurately. This imbalance restricts the generalization capabilities of NIDS and hinders its practical applicability.

Generative adversarial networks (GANs) [25] have emerged as promising data augmentation techniques for mitigating this issue. By learning the distribution of real attack samples, GANs can generate synthetic attack traffic that closely resembles actual malicious behavior, thereby balancing the datasets and improving the detection performance. For example, in [26], synthetic abnormal traffic generated by a GAN was used to oversample the minority class, leading to improved binary classification accuracy when combined with random forest classifiers. Shahriar and others [27] proposed G-IDS, a GAN-based intrusion detection framework in which the inclusion of synthetic samples enhances both the detection accuracy and model robustness. Similarly, in [28], the authors introduced IDA-GAN, a classification framework that combines variational autoencoders with GANs. The autoencoder captures latent data distributions, enabling GANs to generate diverse and high-quality intrusion samples that help the model learn more precise classification boundaries.

Although GANs are effective in addressing binary imbalances, standard GAN models cannot differentiate between multiple attack categories, which limits their application in multiclass intrusion-detection scenarios. Conditional variants of GANs have been proposed to overcome this issue. Li and others [29] proposed a conditional GAN enhanced using bidirectional encoder representations from transformers to generate multiclass attack traffic for intrusion detection. [30], an auxiliary classifier GAN (ACGAN) was employed to incorporate class labels during the generation process, enabling the synthesis of various types of attack traffic to improve class balance in multi-category datasets.

To further refine the data augmentation, Ding and others [31] developed a hybrid method that integrates KNN for undersampling with table-aware ACGAN-based oversampling, thereby achieving more effective sample distributions and classification results. In a more advanced approach, Zhang and others [32] introduced ACGAN-GNN, which combines ACGAN for minority sample expansion with a heterogeneous GNN to model the relational structure among traffic flows. This integration has shown promising improvements in the classification performance by simultaneously addressing data imbalances and graph-based dependencies. However, these methods often overlook the influence of graph topology and edge features on the final detection performance. There is significant room for improvement in jointly leveraging adversarial learning and graph structural information to construct discriminative and balanced intrusion detection frameworks.

1.2 | Contributions and organization

In this study, we propose GraphACGAN, an intrusion detection framework that integrates ACGAN with a GNN to address the performance degradation caused by unstructured traffic features and imbalanced datasets. The main contributions of this study are summarized as follows.

- We embed E-GraphSAGE, a GNN variant, into the discriminator of ACGAN to capture the latent structural information within network flows effectively. This approach enhances the discriminator's capacity to recognize subtle variations in traffic patterns and improves its ability to distinguish between diverse attack types.
- The generator is conditioned on class labels to synthesize specific types of attack samples, enabling targeted data augmentation. This augmentation strategy mitigates class imbalance, reduces the false positive rate, and improves the generalization performance of the intrusion detection model.
- We conduct comprehensive experiments on three challenging benchmark datasets, namely NF-BoT-IoT-v2, NF-ToN-IoT-v2, and NF-UNSW-NB15-v2.[1] The results show that GraphACGAN consistently outperforms baseline models, including ACGAN, E-GraphSAGE, LSTM, GCN, and KNN, on both binary and multi-class classification tasks. On NF-UNSW-NB15-v2, GraphACGAN achieves F_1-score improvements of 7.55%, 27.47%, 9.48%, and 14.29%, respectively, over these methods.

The remainder of this paper is organized as follows: Section 2 introduces the preliminaries of E-GraphSAGE and ACGAN. The details of the implementation of the GraphACGAN detection approach are presented in Section 3. Section 4 discusses dataset selection and our experimental methodology. Section 5 presents the experimental results and their performances. Finally, our conclusions are summarized in Section 6.

[1] https://espace.library.uq.edu.au/view/UQ:38a2d07.

2 | PRELIMINARIES

2.1 | E-GraphSAGE

In E-GraphSAGE, the graph nodes represent the IP addresses of the hosts, whereas the edges represent the communication between the hosts (that is, network flows). This approach allows the integration of edge features and topological patterns into IoT network intrusion detection, thereby enabling the detection of malicious network flows. Graph sampling and aggregation techniques are used. The model first samples the neighbors of each node in the graph, and then learns a function to aggregate the representations of the neighboring vertices to produce the target vertex embedding. In particular, as shown in Figure 1, the training data are constructed as a graph for node sampling and edge feature aggregation. For example, the edge features of the selected one-hop neighbor nodes a and b and two-hop neighbor nodes e, f, and g are aggregated layer-by-layer to the endpoint u of edge $\widehat{uv}$ to derive the node features of u. For the other endpoint v, the same two layers of sampling and aggregation are used to obtain the node features of v. The node features u and v are concatenated to form an embedded representation of $\widehat{uv}$, which is input into the softmax activation function to derive the edge type.

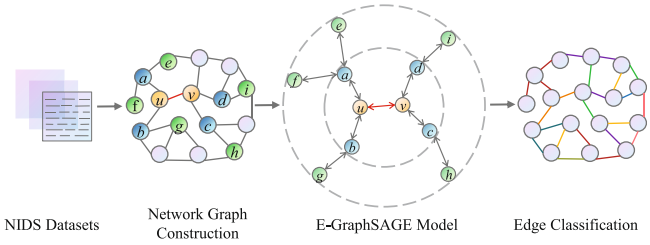

FIGURE 1 Edge classification based on E-GraphSAGE.

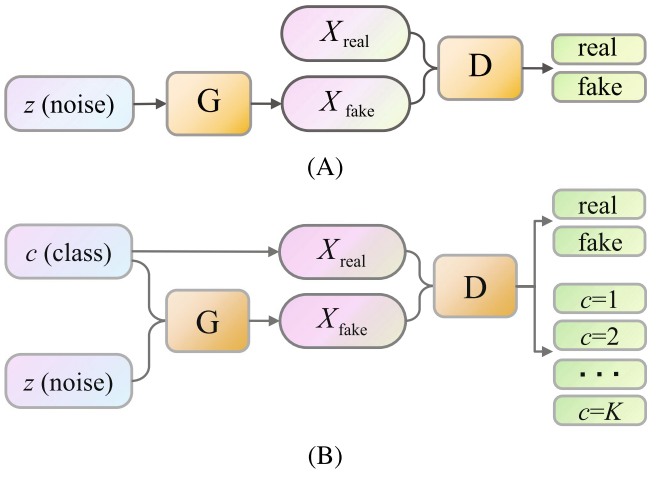

FIGURE 2 ACGAN versus GAN. (A) GAN framework and (B) ACGAN framework.

2.2 | ACGAN

A GAN consists of two adversarial neural networks: a generator (called G) and a discriminator (called D). As shown in Figure 2A, the task of the generator is to transform the received random noise into a fake sample similar to an actual sample, whereas the discriminator's task is to identify the fake and actual samples. When the adversarial game reaches the Nash equilibrium, both networks can learn the distribution of the real samples. The generator generates traffic samples, and the discriminator performs binary classification. However, it is necessary to generate multiple sample types or perform multi-classification in certain scenarios. ACGAN [33] extends the traditional GAN framework to generate different types of samples by inputting label information into G. Subsequently, D uses a multi-classification function to estimate the posterior probability of the samples to identify their types. Figure 2B shows that the given class label and random noise z are input into G to generate a fake sample X_{fake}, which is then input into D, along with the real sample X_{real}. The two cost functions are then formulated as:

$$L_S = \mathrm{E}[\log \mathrm{P}(S = \text{real}|X_{\text{real}})] + \mathrm{E}[\log \mathrm{P}(S = \text{fake}|X_{\text{fake}})], \tag{1}$$

and

$$L_C = \mathrm{E}[\log \mathrm{P}(C = c|X_{\text{real}})] + \mathrm{E}[\log \mathrm{P}(C = c|X_{\text{fake}})]. \tag{2}$$

The former determines whether the sample is real, and the latter determines whether the classification is true.

During adversarial training, G maximizes $L_C - L_S$ to identify as many generated samples as possible, whereas D maximizes $L_S + L_C$ to identify real, fake, and K types.

As shown in Figure 2, ACGAN demonstrates particular advantages for network intrusion detection, owing to its ability to precisely identify attack types, thereby enabling appropriate defensive countermeasures. By training the ACGAN on specific attack types, it can learn to produce accurate traffic patterns that represent different attack scenarios. Precise categorization of attack types is critical for deploying tailored defense strategies. Furthermore, because the data produced by the ACGAN generator are more realistic and accurate, they can train the discriminator better, thereby improving the accuracy and efficiency of the intrusion detection system.

3 | PROPOSED SOLUTION

As a common format for recording data communications, network traffic consists of an identified communication source, destination IP field, and traffic features. Traffic features include the number of incoming and outgoing bytes, multiple FLAG fields at the TCP layer, and traffic duration in the NF-ToN-IoT-v2 dataset. To model the network traffic as a graph structure using a GNN [34], we first mapped the source IP address to a random IP address between 172.16.0.1 and 172.31.0.1. This approach avoids the potential problem of source IP addresses by providing a small proportion of unintentional labels for attack traffic. Next, a binary combination of a source IP address and source port number acts as a source node "*IPV4_SRC_ADDR*," and a binary combination of a destination IP address and destination port number acts as a destination node "*IPV4_DST_ADDR*." The connection relationships between the source and destination nodes are used to construct a graph in which standardized traffic features are considered edge features, and the number of edges E in the graph represents the quantity of network traffic. The nodes in the graph contain no features; therefore, they must be embedded in a vector with dimensions equal to the number of edge features.

By integrating E-GraphSAGE and ACGAN, we designed an intrusion detection framework that applies adversarial learning to enhance the generation and discrimination abilities simultaneously. In Figure 3, network traffic data are directly constructed as a real graph $\mathcal{G}$. Random noise and a given class label are input into G to generate the edge feature $H_G = G(z, \text{label})$. By reusing the topology of a real graph, all the generated edge features are constructed as a generated graph $\tilde{\mathcal{G}}$. Both graphs are input into D and trained in E-GraphSAGE to implement multi-classification. D also implements binary classification if the detection requirements are reduced. The proposed framework can perform coarse and fine detection according to network situations and user requirements, exhibiting excellent flexibility and scalability.

In the following subsections, each functional module is described in detail.

3.1 | Generator

G is a five-layer neural network as shown in Figure 4. A 100-dimensional vector converted by the class label is multiplied by a 100-dimensional normal random noise vector **z**, which is input into an input layer with 100 neurons, followed by three hidden layers with 100 neurons each, and an output layer with n neurons, where n represents the dimensions of the traffic features. Therefore, the generator output is the generated edge feature, H_G. Combining the flows' source and destination IP addresses with the generated edge features creates the graph $\tilde{\mathcal{G}}$.

Based on (1) and (2), the goal of training G is to maximize

$$L_G = E[\log P(S = \text{real}|H_G)] + E[\log P(l = \text{label}|H_G)]. \quad (3)$$

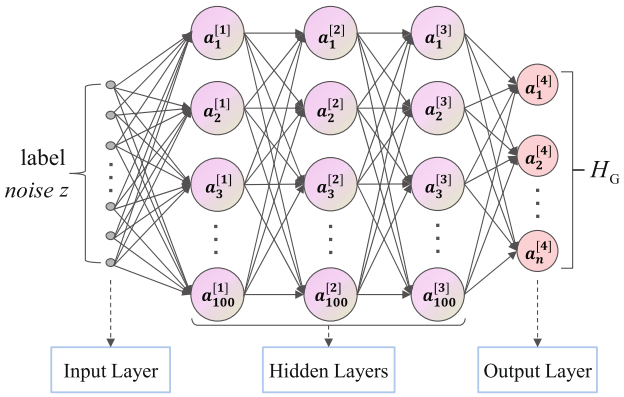

FIGURE 4 Generator network framework.

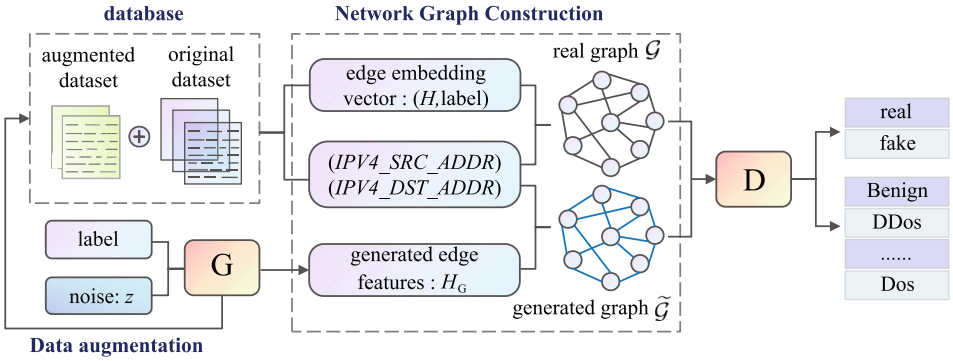

FIGURE 3 GraphACGAN intrusion detection framework.

Maximizing the first term makes the generated edge feature H_G similar to the real traffic feature H, whereas maximizing the second term makes the generated edge feature H_G more similar to the real traffic label, thereby enabling the trained generator to output usable traffic data according to the class labels.

3.2 | GNN-enhanced discriminator

The discriminator integrates E-GraphSAGE to perform the following sampling and aggregation steps on a real graph $\mathcal{G}$ and a generated graph $\tilde{\mathcal{G}}$ to identify the attack type.

3.2.1 | Full neighborhood random sampling

For vertices u and v of any edge $\widehat{uv}$ in $\mathcal{G}$ and $\tilde{\mathcal{G}}$, we randomly sample one- and two-hop neighbor nodes and their connecting edges.

3.2.2 | Edge information aggregation

A node v's feature vector is initialized as $h_v^0 = (1,\ldots,1)^T$.

$$e_{uv}^{k-1} = \sigma\left(W_1^k \cdot \mathrm{CONCAT}\left(h_v^{k-1}, e_{uv}\right)\right), k = 1,\ldots,K. \quad (4)$$

Based on (4), the neighbor node features and edge features of node v are concatenated to obtain the $(k-1)$th layer edge features e_{uv}^{k-1} through weighting and activation operations. The node features of the k-th neighbor nodes of v are then obtained by aggregating the side features e_{uv}^{k-1} as follows:

$$h_{N(v)}^k = \mathrm{AGG}_k\left(\{e_{uv}^{k-1}, \forall u \in N(v), \widehat{uv} \in \varepsilon\}\right) = \sum_{u \in N(v)} \frac{e_{uv}^{k-1}}{|N(v)|}, \quad (5)$$

where $N(v)$ is v's sampling neighborhood, ε is the set of sampled edges, and $|N(v)|$ is the number of v's sampled edges. Next, node feature h_v^{k-1} and aggregation feature $h_{N(v)}^k$ are concatenated according to (6). The concatenated vector is weighted and activated to obtain the kth layer node embedding.

$$h_v^k = \sigma\left(W_2^k \cdot \mathrm{CONCAT}\left(h_v^{k-1}, h_{N(v)}^k\right)\right). \quad (6)$$

Over K iterations, the K-th layer node embedding is derived as

$$z_v^K = h_v^K, \forall v \in V. \quad (7)$$

In the aggregation algorithm, K refers to the number of aggregations, weight matrices, or layers in the network. The number of layers in the network can be understood as the number of neighbor hops that require maximum access. K was set to two in this study because, in the aggregation algorithm, a 10% to 15% improvement in performance was achieved by setting $K = 2$ instead of $K = 1$. However, when K was set higher than two, the resulting improvement ranged from 0% to 5%, whereas the computational time increased by 10 to 100 times. Finally, the edge embedding is obtained by concatenating the node embeddings u and v as follows:

$$z_{uv}^K = \mathrm{CONCAT}\left(z_u^K, z_v^K\right), \widehat{uv} \in \varepsilon. \quad (8)$$

Then, z_{uv}^K is input into a softmax function to implement multi-classification.

According to the sampling and aggregation of $\mathcal{G}$ and $\tilde{\mathcal{G}}$, two loss functions are defined on the real and generated graphs as follows:

$$L_{\mathrm{real}} = \mathrm{E}[\log\ \mathrm{P}(S = \mathrm{real}|H)] + \mathrm{E}[\log\ \mathrm{P}(l = \mathrm{label}|H)], \quad (9)$$

and

$$L_{\mathrm{fake}} = \mathrm{E}[\log \mathrm{P}(S = \mathrm{fake}|H_G)] + \mathrm{E}[\log \mathrm{P}(l = \mathrm{label}|H_G)]. \quad (10)$$

Maximizing (9) can improve the learning of real graph information, whereas maximizing L_{fake} in (10) can increase the discriminator's accuracy. Therefore, the discriminator utilizes the following optimization objective function:

$$\max L_D = L_{\mathrm{real}} + L_{\mathrm{fake}}. \quad (11)$$

3.3 | Data augmentation

We use the generation ability of the trained generator to alleviate the lack of attack-traffic data. In Figure 3, the assigned attack traffic is generated to balance the original intrusion detection dataset. Retraining the GraphACGAN model with an augmented training dataset prevents overfitting and improves its generation and discrimination capabilities.

An intrusion-detection algorithm that combines E-GraphSAGE and ACGAN is summarized in

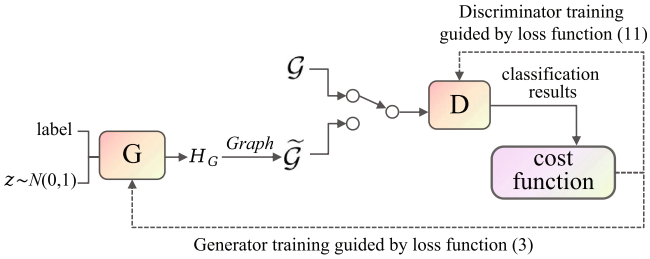

FIGURE 5 Algorithm workflow.

Algorithm 1, and the corresponding workflow is illustrated in Figure 5. First, G generates the edge feature H_G by inputting noise z and label. Then, the generated network traffic feature H_G, along with the source node "*IPV4_SRC_ADDR*," destination node "*IPV4_DST_ADDR*," and label, is used to generate a graph $\tilde{\mathcal{G}}$, which is input into D to obtain the output results and compute the cost. The loss of G is computed based on (3), and the parameters of G are updated accordingly. Similarly, "*IPV4_SRC_ADDR*," "*IPV4_DST_ADDR*," the network traffic feature H, and label form a graph $\mathcal{G}$ and are input into the discriminator to obtain classification results. By combining these two sets of output results, the loss of D is computed based on (11), and the parameters of D are updated accordingly.

Algorithm 1 Adversarial training algorithm.

Require: Network traffic feature H, Class label, $IPV4\_SRC\_ADDR$, $IPV4\_DST\_ADDR$, Normal distribution random noise z

Ensure: Optimize G and D

Init: G, D
$\mathcal{G} \leftarrow Graph(IPV4\_SRC\_ADDR,$
$IPV4\_DST\_ADDR, H,$ label).
for number of training iterations **do**
  #Train Generator
  $H_G \leftarrow$ G(z, label);
  $\tilde{\mathcal{G}} \leftarrow$ Graph($IPV4\_SRC\_ADDR,$
  $IPV4\_DST\_ADDR, H_G, label$);
  $D(\tilde{\mathcal{G}})$ gives the classification results of $\tilde{\mathcal{G}}$;
  Calculate the loss of G based on (3);
  Update the parameters of G via Adam;
  #Train Discriminator
  $D(\mathcal{G})$ gives the classification results of $\mathcal{G}$;
  $D(\tilde{\mathcal{G}})$ gives the classification results of $\tilde{\mathcal{G}}$;
  Calculate the loss of D based on (11);
  Update the parameters of D via Adam;
end for

The model training process must calculate and update the gradients of G and D. The number of floating-point operations required to update G and D in each training iteration is $O(m{\cdot}E)$, where m represents the sum of the dimensions of the two weight matrices, W_1^k and W_2^k, which are calculated as

$$m = \sum_{k=1}^{K} (\dim(W_1^k) + \dim(W_2^k)). \tag{12}$$

Accordingly, the overall computational complexity of the algorithm is approximated as $O((m{\cdot}\mathrm{E}){\cdot}I)$, where I denotes the total number of training iterations.

4 | EXPERIMENTAL SETUP

4.1 | Dataset selection

Three widely used authoritative datasets, namely NF-BoT-IoT-v2, NF-ToN-IoT-v2, and NF-UNSW-NB15-v2, with different degrees of data imbalance were used to evaluate the detection performance of GraphACGAN and the baseline approaches. An overview of these datasets is presented in Table 1 and their class distributions and proportions are listed in Table 2. The three datasets contained tens of millions of network traffic data points. Therefore, we randomly sampled 10% of the original datasets to form our network traffic datasets, where 70% of the traffic data were utilized to train the models and 30% were used to test the models.

4.2 | Baselines and hyperparameters

G consists of four fully connected layers with 100, 100, 100, and 39 neurons in each layer, respectively. D adopts a two-layer E-GraphSAGE model with 128 neurons in each layer using the mean function for aggregation, and edge classification is performed using a softmax classifier. For GraphACGAN, a rectified linear unit (ReLU) was used as the nonlinear activation function and a 2-hop domain was used for computation. The proposed method was implemented using the Adam optimizer in PyTorch. The learning rate (lr) was set to 0.0002 for all three datasets and the dropout parameter was set to 0.2.

The proposed GraphACGAN was compared with four baseline models.

- E-GraphSAGE, which is utilized for binary and multi–class intrusion detection tasks [24], is consistent with the E–GraphSAGE embedded in the proposed framework.

TABLE 1 Overview of three intrusion detection datasets.

Dataset	Count	Number of classes	Number of features	Benign to attack samples ratio
NF-BoT-IoT-v2	37 763 497	5	43	0.0 to 10.0
NF-ToN-IoT-v2	16 940 496	10	43	0.36 to 6.4
NF-UNSW-NB15-v2	18 893 708	10	43	9.6 to 0.4

TABLE 2 Class distributions and proportions of the three datasets.

Dataset	Classes				
NF-BoT-IoT-v2	Benign	Reconnaissance	DoS	DDoS	**Theft**
	0.36%	6.94%	44.15%	48.54%	**0.01%**
NF-ToN-IoT-v2	Benign	Backdoor	DoS	DDoS	Injection
	36.01%	0.1%	4.21%	11.96%	4.04%
	MITM	Password	Ransomware	Scanning	XSS
	0.04%	6.81%	0.02%	22.32%	14.49%
NF-UNSW-NB15-v2	Benign	Fuzzers	**Dos**	Reconnaissance	**Worms**
	96.02%	0.93%	**0.25%**	0.53%	**0.01%**
	Backdoor	Generic	Exploits	**Shellcode**	Analysis
	0.09%	0.69%	1.32%	**0.06%**	0.1%

Note: The class(es) with the smallest sample proportion in each dataset are highlighted in bold.

- The GCN [35, 36], which was structured with comparable parameters to E-GraphSAGE for equitable evaluation, utilizes two convolutional layers with 256 neurons each, followed by ReLU activation with a dropout rate of 0.2.
- ACGAN is used to perform the multi-classification of network traffic, as described in [32]. The generator network structure of ACGAN is identical to that employed in GraphACGAN. The discriminator network is structured with four fully connected layers of 512, 512, 512, and 39 neurons, in order. The LeakyReLU activation function is employed.
- LSTM is used to perform binary and multi-class classification for network traffic, as described in [7]. The LSTM benchmark consists of six LSTM layers. The number of neurons in the first five layers are 39, 20, 60, 80, and 90, in order. The number of neurons in the last layer corresponds to the number of categories in the NF-BoT-IoT-v2, NF-ToN-IoT-v2, and NF-UNSW-NB15-v2 datasets (5, 10, and 10, respectively). The total number of parameters is approximately 21K.
- KNN [10], one of the classic intrusion detection algorithms with the hyperparameter K set to three.

4.3 | Evaluation metrics

Indicators such as accuracy, precision, recall, and F_1-score are commonly used to evaluate detection performance. Let TP, TN, FP, and FN denote the true positive, true negative, false positive, and false negative, respectively. The formulas for the four indicators are as follows:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}, \tag{13}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{14}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{15}$$

$$F_1\text{-score} = \frac{2 \times \text{TP}}{2 \times \text{TP} + \text{FP} + \text{FN}}. \tag{16}$$

Accuracy intuitively reflects the proportion of correct detection results. Precision and recall emphasize the proportion of correct classifications among positive predictions and actual positives, respectively, whereas the F_1-score is a comprehensive evaluation metric for comparison. Because all three datasets exhibit data imbalance, accuracy and weighted F_1-score were selected as the main performance indicators.

5 | EXPERIMENTAL RESULTS

This section compares the binary and multi-class classification performances of different models on the three

datasets. In addition, we present two data augmentation experiments conducted on the NF-ToN-IoTv2 and NF-UNSW-NB15-v2 datasets. Finally, we present the visualization results for the NF-BoT-IoT-v2 dataset.

5.1 | Detection performance analysis

As shown in Table 3, the binary and multi-classification indicators of GraphACGAN were higher than those of E-GraphSAGE and ACGAN. In particular, GraphACGAN's two indicators for the binary classification approach were 100%. On the NF-UNSW-NB15-v2 dataset, which contains a lower proportion of attack traffic, GraphACGAN achieved weighted F_1-score that were 7.55%, 8.1%, 27.47%, 9.48%, and 14.29% higher than those of E-GraphSAGE, GCN, ACGAN, LSTM, and KNN, respectively. Although GCN demonstrates competitive performance by effectively capturing topological patterns, it fails to model edge features, which are crucial for traffic analysis. ACGAN fails to take advantage of the non-Euclidean graph structure of network traffic. LSTM and KNN use a single traffic source, leading to low training efficiency and a failure to mine the common features of multiple traffic sources to improve the detection accuracy. Although E-GraphSAGE converts network traffic into a graph structure, it cannot improve discriminator performance without additional auxiliary modules. By integrating E-GraphSAGE and ACGAN, we constructed a new deep learning framework with enhanced attack recognition capabilities.

The precision, recall, and F_1-score for benign and attack traffic detection across the three datasets are presented in Tables 4,5, and 6. GraphACGAN consistently achieved superior performance, with attack traffic F_1-score of 99.12%, 95.91%, and 97.65%, and benign traffic F_1-scores of 99.84%, 98.97%, and 99.59%, respectively. In contrast, GCN struggled significantly with benign traffic detection on NF-BoT-IoT-v2 (with an F_1-score of only 37.09%). ACGAN performed poorly in detecting benign traffic (F_1-score < 71%) on NF-UNSWNB15-v2. LSTM and KNN achieved moderate F_1-scores (70%–88%) for benign and attack traffic under imbalanced proportions. These results demonstrate that GraphACGAN provides superior performance in terms of both precision and recall, benefiting from the strong feature extraction ability of E-GraphSAGE and strong generalization ability of ACGAN.

Figures 6,7, and 8 present three types of multi-classification indicators for the five models of the three datasets. One can see that GraphACGAN was superior to the other four models in terms of every indicator. Notably, the NF-BoT-IoT-v2 dataset has the highest proportion of attack traffic, followed by the NF-ToN-IoT-v2 dataset, whereas the NF-UNSW-NB15-v2 dataset has the

TABLE 3 Comparison of binary and multi-classification.

Algorithm	Binary classification						Multi-classification					
	NF-BoT-IoT-v2		NF-ToN-IoT-v2		NF-UNSW-NB15-v2		NF-BoT-IoT-v2		NF-ToN-IoT-v2		NF-UNSW-NB15-v2	
	Acc.	W-F1	Acc.	W-F1	Acc.	W-F1	Acc.	W-F1	Acc.	W-F1	Acc.	W-F1
GraphACGAN	99.73%	99.73%	98.37%	98.97%	99.30%	99.30%	96.10%	96.08%	92.35%	92.99%	95.56%	96.15%
E-GraphSAGE	98.76%	98.58%	96.91%	96.98%	98.02%	97.99%	93.29%	93.43%	89.00%	89.36%	83.80%	88.60%
GCN	92.20%	94.49%	95.27%	95.43%	95.53%	95.48%	91.37%	92.85%	85.71%	83.05%	86.02%	88.05%
ACGAN	96.67%	96.48%	93.89%	93.94%	89.90%	90.90%	92.47%	92.18%	68.48%	63.38%	73.10%	68.68%
LSTM	95.13%	94.96%	89.16%	88.79%	95.54%	95.25%	88.20%	87.84%	80.07%	77.87%	87.34%	86.67%
KNN	87.50%	88.68%	98.29%	89.12%	89.60%	89.76%	82.37%	83.71%	73.47%	73.12%	82.11%	81.86%

TABLE 4 Benign and attack traffic detection on NF-BoT-IoT-v2.

Model	Class	Precision	Recall	F_1-score
GraphACGAN	Benign	99.96%	99.73%	99.84%
	Attack	98.47%	99.78%	99.12%
E-GraphSAGE	Benign	92.12%	50.55%	65.29%
	Attack	98.84%	99.90%	99.37%
GCN	Benign	22.79%	99.52%	37.09%
	Attack	99.97%	92.03%	95.84%
ACGAN	Benign	98.32%	75.45%	85.38%
	Attack	96.49%	99.18%	98.12%
LSTM	Benign	86.85%	73.39%	79.55%
	Attack	96.15%	98.35%	97.24%
KNN	Benign	50.96%	82.43%	62.98%
	Attack	97.14%	88.25%	92.48%

TABLE 5 Benign and attack traffic detection on NF-ToN-IoT-v2.

Model	Class	Precision	Recall	F_1-score
GraphACGAN	Benign	99.83%	98.12%	98.97%
	Attack	92.74%	99.32%	95.91%
E-GraphSAGE	Benign	94.97%	98.72%	96.81%
	Attack	96.93%	99.15%	92.58%
GCN	Benign	81.28%	98.53%	89.08%
	Attack	99.62%	94.47%	96.98%
ACGAN	Benign	96.76%	95.13%	95.94%
	Attack	85.49%	90.00%	87.69%
LSTM	Benign	91.68%	95.17%	93.39%
	Attack	76.27%	64.27%	69.76%
KNN	Benign	91.80%	94.30%	93.03%
	Attack	80.43%	73.54%	76.83%

lowest proportion. The lower the proportion, the higher the recognition difficulty. Precisely identifying attack types with a small proportion of attack data is an important method for evaluating the detection effectiveness of the five models.

In the NF-BoT-IoT-v2, a high proportion of attack traffic is present, particularly DoS and DDoS attacks. As shown in Figure 6, all five methods performed well at detecting these two categories. However, ACGAN and LSTM could not identify theft attacks, which accounted for a small proportion of the dataset (only 0.01%). The accuracy, recall, and F_1-scores of GraphACGAN for this type of attack reached 87.85%, 99.99%, and 93.53%, respectively, which are higher than those of

TABLE 6 Benign and attack traffic detection on NF-UNSW-NB15-v2.

Model	Class	Precision	Recall	F_1-score
GraphACGAN	Benign	99.45%	99.73%	99.59%
	Attack	98.42%	96.90%	97.65%
E-GraphSAGE	Benign	98.19%	99.51%	98.84%
	Attack	97.01%	89.58%	93.15%
GCN	Benign	94.97%	98.72%	96.81%
	Attack	96.93%	88.57%	92.56%
ACGAN	Benign	56.50%	94.32%	70.67%
	Attack	98.07%	90.25%	93.90%
LSTM	Benign	95.25%	99.73%	97.43%
	Attack	97.89%	71.84%	82.86%
KNN	Benign	95.20%	89.34%	92.18%
	Attack	79.45%	90.15%	84.46%

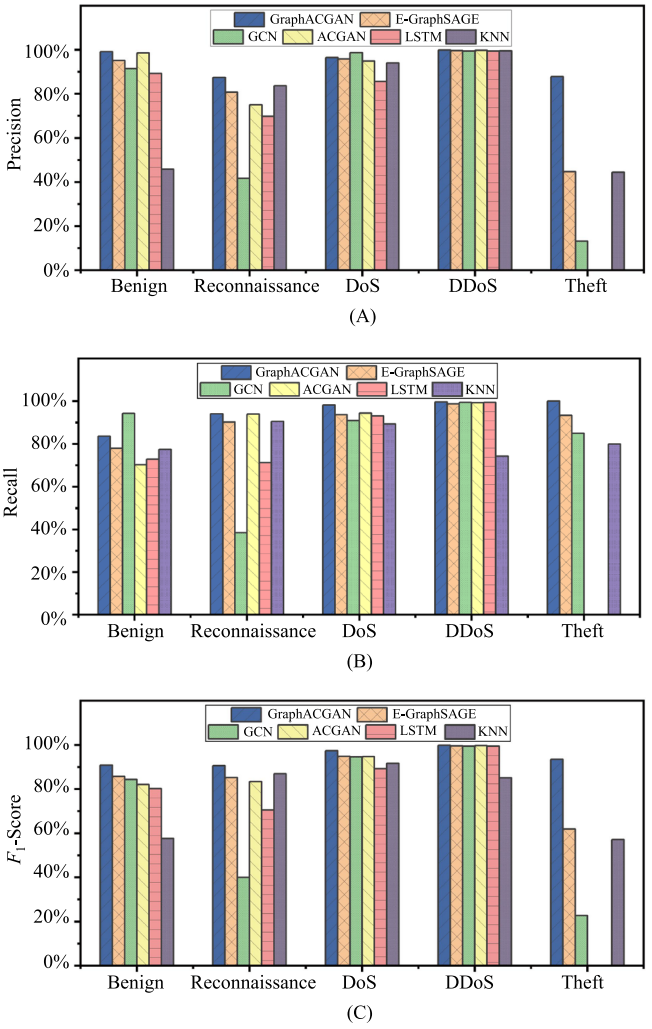

FIGURE 6 Multi-classification performance comparison on NF-BoT-IoT-v2. (A) Precision for each class, (B) recall for each class, and (C) F_1-scores for each class.

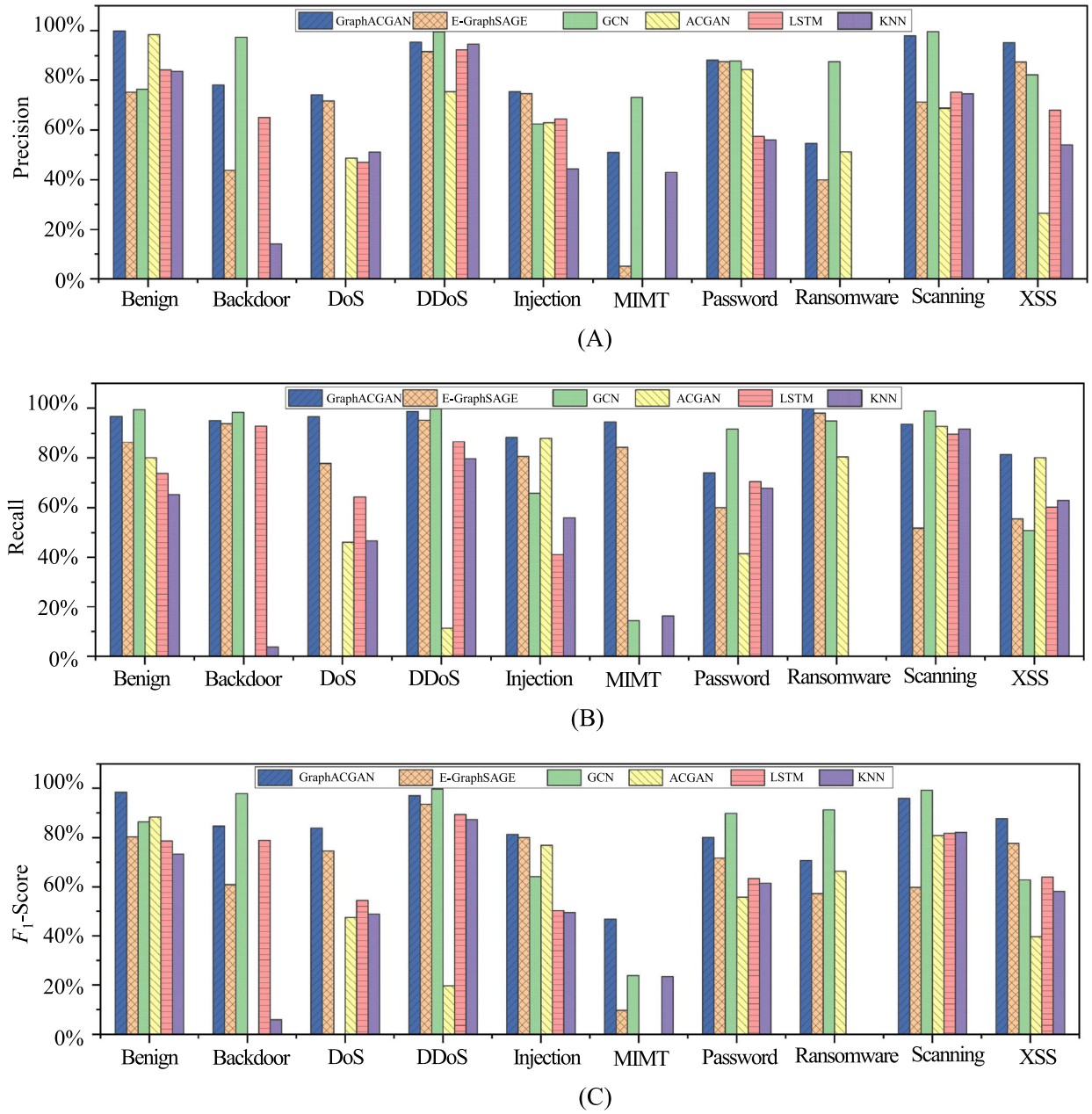

FIGURE 7 Multi-classification performance comparison on NF-ToN-IoT-v2. (A) Precision for each class, (B) recall for each class, and (C) F_1-scores for each class.

E-GraphSAGE, GCN, and KNN. Because of the mutual benefits of E-GraphSAGE and ACGAN, GraphACGAN's detection performance was further improved, particularly for attack types that were previously difficult to identify.

Figure 7 illustrates the model performance on the NF-ToN-IoT-v2 dataset. For MIMT attacks, GraphACGAN achieved the highest F_1-score, significantly outperforming the other models, whereas ACGAN and LSTM failed entirely in this category. Similarly, for Ransomware attacks (comprising only 0.02% of the samples), GNN models (GCN, GraphACGAN, and E-GraphSAGE) exhibited superior detection capabilities compared to traditional approaches. Notably, the GCN completely failed to detect DoS attacks. In contrast, GraphACGAN successfully detected all attack types without exception, demonstrating comprehensive coverage and balanced performance across the entire threat spectrum.

Compared to the NF-BoT-IoT-v2 and NF-ToN-IoT-v2 datasets, the imbalance in the class distribution was more pronounced in the NF-UNSW-NB15-v2 dataset, where the total proportion of all attack types was less than 4%. However, as shown in Figure 8, GraphACGAN still

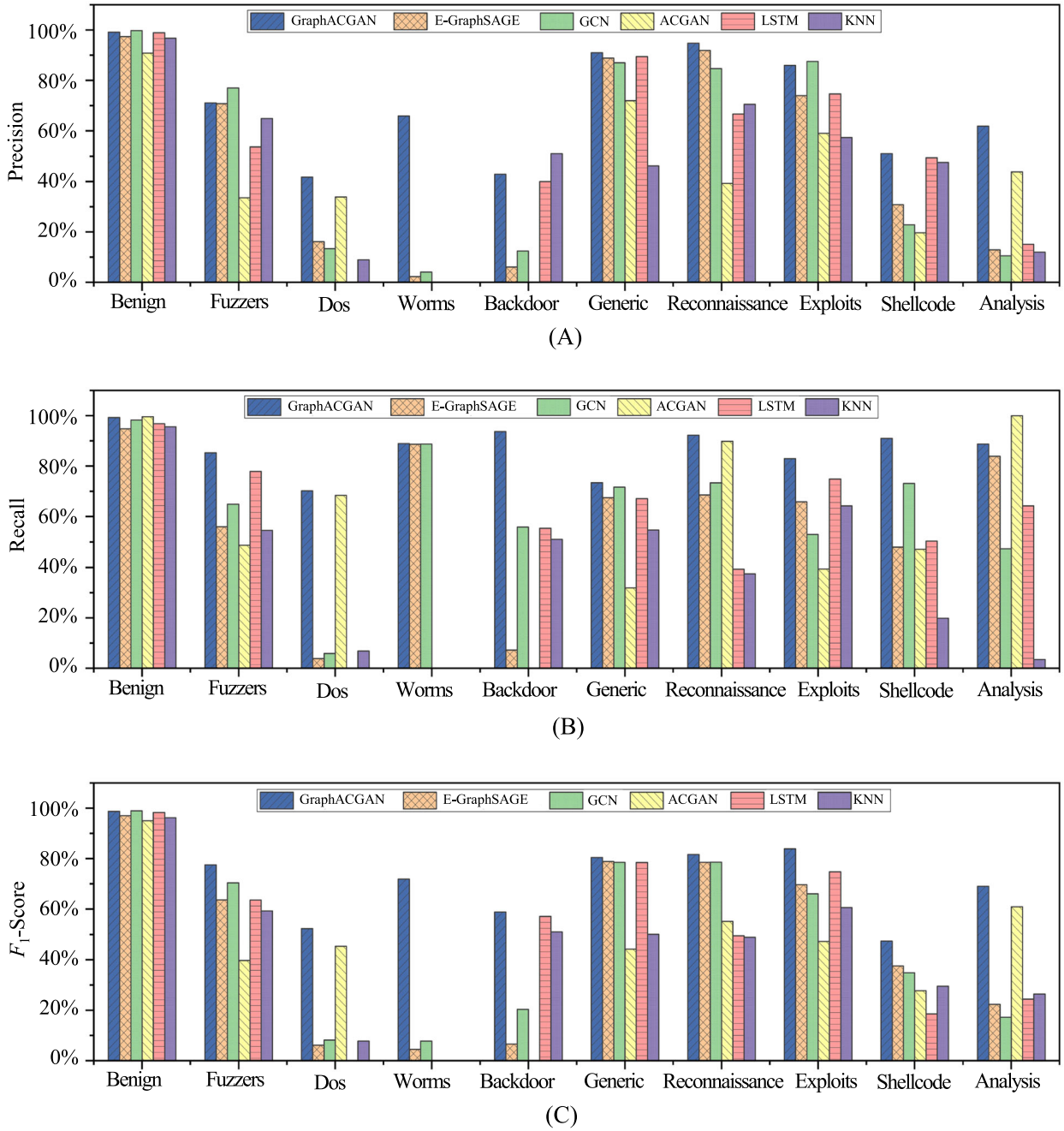

FIGURE 8 Multi-classification performance comparison on NF-UNSW-NB15-v2. (A) Precision for each class, (B) recall for each class, and (C) F_1-scores for each class.

outperformed the other four methods, particularly in Worm and DoS attacks, which have very low proportions. The accuracy, recall, and F_1-scores of GraphACGAN for Worm detection were 65.93.

5.2 | Impact of data augmentation

To alleviate the data imbalance problem observed in the above experiments, we augmented all types of attack traffic according to their proportions to improve the multi-classification performance of GraphACGAN. Specifically, 500, 1000, 2000, 3000, 4000, and 5000 attack samples generated by G were added to the NF-ToN-IoT-v2 and NF-UNSW-NB15-v2 datasets to observe the changes in detection accuracy and F_1-score.

As shown in Figure 9, with a gradual increase in the number of generated attack samples, the accuracy trend for the augmented dataset first increases and then decreases compared with the original accuracy. In

particular, the multi-classification accuracy of GraphACGAN was maximized when the number of added attack samples was 2000 and 3000 on the NF-ToN-IoT-v2 and NF-UNSW-NB15-v2 datasets, respectively. Figure 10 reveals that the F_1-scores for MIMT and Shellcode attacks improved compared with the low detection rates observed in Figures 6–8, indicating that adding sufficient attack samples can help imbalanced datasets achieve better class balance and improve detection accuracy. However, too many generated samples cause the augmented dataset to deviate from the actual data distribution, causing GraphACGAN to learn many biased features and negatively affecting accuracy.

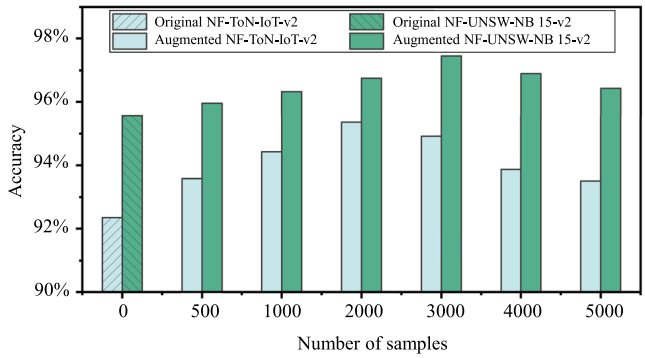

FIGURE 9 GraphACGAN's multi-classification accuracy on original and augmented datasets.

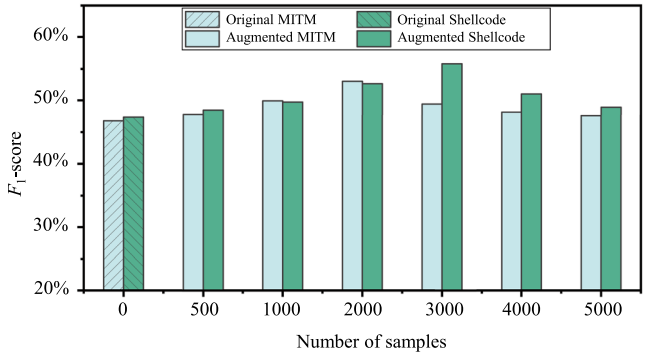

FIGURE 10 F_1-scores for MIMT and Shellcode attacks on original and augmented datasets.

Further analysis of GraphACGAN's weighted F_1-scores with optimal sample additions (2000 for NF-ToN-IoT-v2 and 3000 for NF-UNSW-NB15-v2) revealed significant improvements in both the binary and multi-classification metrics, as presented in Table 7. Overall, the experimental results demonstrate that expanding the attack traffic by a small proportion mitigates data imbalance and improves the discriminator's performance. In addition, random noise input facilitates the generation of attack variants that are absent from the original training data but consistent with known category distributions, thereby enhancing GraphACGAN's generalization capabilities when confronted with variations in established attack types.

5.3 | Performance under resource constraints

To assess real-world applicability, we emulated edge computing constraints using Docker containerization based on Raspberry Pi 4B specifications (4 CPU cores@1.5 GHz, 4 GB RAM, and 1 Gbps network). We then deployed our model, pretrained on the NF-BoT-IoT-v2 dataset in this environment. As shown in Tables 8 and 3, GraphACGAN achieves a classification accuracy of 96.1% with a

TABLE 7 Weighted F_1-scores of GraphACGAN on original and augmented datasets.

		Weighted F_1-score	
Dataset	**Classification**	**Original**	**Augmented**
NF-ToN-IoT-v2	Binary	98.37%	99.74%
	Multiclass	92.99%	95.74%
NF-UNSW-NB15-v2	Binary	99.30%	99.41%
	Multiclass	96.15%	97.62%

TABLE 8 Performance comparisons in terms of inference speed and throughput.

Method	Params	Inference time (s)	Throughput (samples per second)
GCN	70.6 K	2.975	121 022
GraphACGAN	71.4 K	3.306	108 903

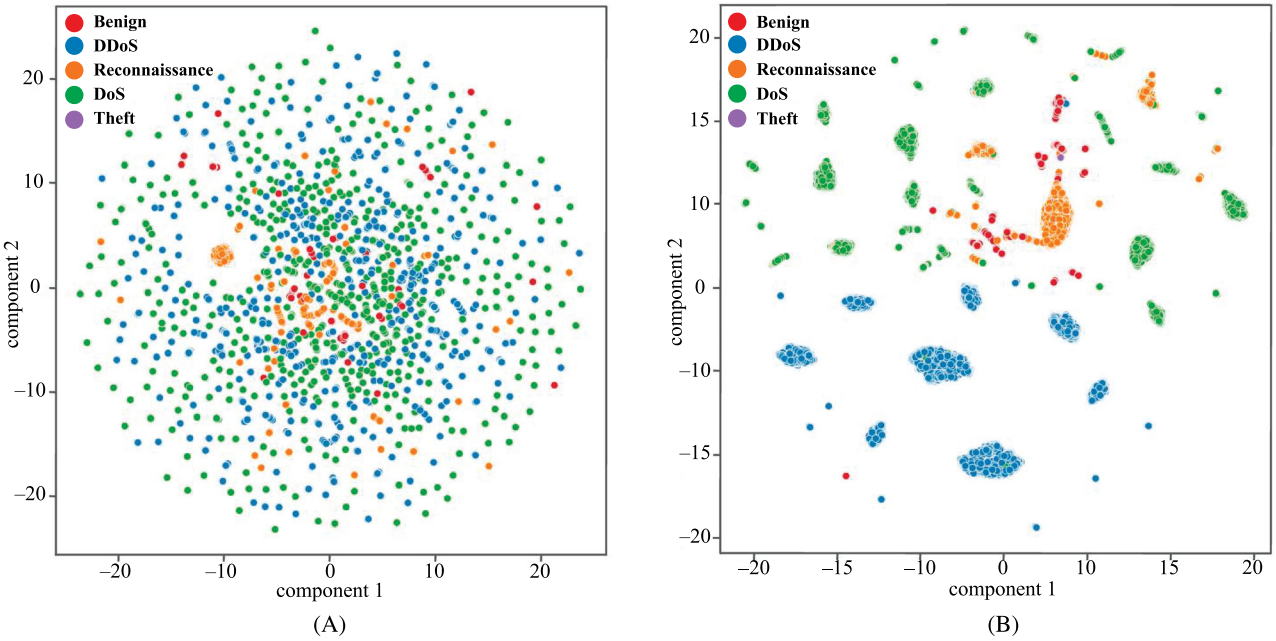

FIGURE 11 Visualization of attack traffic classification. (A) 10 000 random samples on NF-BoT-IoT-v2 and (B) separated high-dimensional edge features.

throughput of 108 903 samples per second, demonstrating its viability for real-time network monitoring in resource-constrained environments. Although GCN exhibits a marginally higher throughput, but suffers from a 4.73% lower detection accuracy because of the lack of hierarchical neighborhood aggregation and advanced message-passing mechanisms. In network security applications, maintaining high accuracy while ensuring sufficient processing speed is essential. GraphACGAN achieves this optimal balance by delivering superior detection performance with a throughput that satisfies practical deployment requirements.

5.4 | Feature distribution visualization

To intuitively understand its excellent performance, we tested the proposed GraphACGAN on the NF-BoT-IoT-v2 dataset to visualize the edge embeddings of a graph using 10000 random samples. Using the unified manifold approximation and projection dimensionality reduction algorithm [37], we mapped 10000 samples of the original features and high-dimensional edge embeddings onto two dimensions (components one and two) for visualization. Figure 11A shows the initial state, and Figure 11B shows that the four attack types and benign traffic are separated, confirming that GraphACGAN can leverage the inherent graph structure of the network flows. GNN and ACGAN mutually reinforce each other to separate attack traffic from normal traffic.

6 | CONCLUSION

We proposed a multi-class network intrusion detection framework called GraphACGAN. This framework is based on ACGAN, where E-GraphSAGE is embedded in the discriminator of ACGAN. This integration enables the discriminator to capture additional hidden feature information, thereby enhancing its ability to detect network attacks. An enhanced discriminator also guides the generator to promote the generation of high-dimensional feature representations of network traffic that approximate the real probability distribution, thereby improving the intrusion detection performance and generalization capability of the model. The experimental results demonstrated that, compared with E-GraphSAGE, GCN, ACGAN, LSTM, and KNN, GraphACGAN has advantages in terms of accuracy, precision, and other indicators. The containerized deployment tests revealed that GraphACGAN maintained its performance advantage while achieving sufficient throughput in constrained computing environments. Our ongoing study focuses on detecting unknown attacks in zero-shot scenarios.

CONFLICT OF INTEREST STATEMENT

The authors declare that there are no conflicts of interest.

ORCID

Hang Shen https://orcid.org/0000-0002-8804-2787

REFERENCES

1. T. Bouraffa and K.-L. Hui, *Regulating information and network security: review and challenges*, ACM Comput. Surv. **57** (2025), no. 5, 1–38.
2. H. Shen, X. Li, Y. Wang, T. Wang, and G. Bai, *Collaborative path penetration in 5G-IoT networks: a multi-agent deep reinforcement learning approach*, Peer Peer Netw. Appl. **18** (2025), no. 3, 113.
3. A. Alqahtani and S. B. Khan, *An optimal hybrid cascade regional convolutional network for cyberattack detection*, Int. J. Netw. Manag. **34** (2024), no. 5, e2247.
4. H. Shen, Y. Zhou, T. Wang, Y. Zhang, J. Huang, G. Bai, and X. Miao, *Blockchain-assisted cross-silo graph federated learning for network intrusion detection*, (Proc. IEEE Global Blockchain Conference (GBC)), 2025, pp. 1–8.
5. S. Agrawal, S. Sarkar, O. Aouedi, G. Yenduri, K. Piamrat, M. Alazab, S. Bhattacharya, P. K. R. Maddikunta, and T. R. Gadekallu, *Federated learning for intrusion detection system: concepts, challenges and future directions*, Comput. Commun. **195** (2022), 346–361.
6. P. Kumar, A. A. Kumar, C. Sahayakingsly, and A. Udayakumar, *Analysis of intrusion detection in cyber attacks using DEEP learning neural networks*, Peer Peer Netw. Appl. **14** (2021), 2565–2584.
7. H. Aydn, Z. Orman, and M. A. Aydn, *A long short-term memory (LSTM)-based distributed denial of service (DDoS) detection and defense system design in public cloud network environment*, Comput. Secur. **118** (2022), 102725.
8. Y. Imrana, Y. Xiang, L. Ali, and Z. Abdul-Rauf, *A bidirectional LSTM deep learning approach for intrusion detection*, Expert Syst. Appl. **185** (2021), 115524–115535.
9. O. M. A. Alsyaibani, E. Utami, and A. D. Hartanto, *An intrusion detection system model based on bidirectional LSTM*, (International Conference on Cybernetics and Intelligent System (ICORIS), Makasar, Indonesia), 2021, pp. 1–6.
10. M. Mohy-eddine, A. Guezzaz, S. Benkirane, and M. Azrour, *An efficient network intrusion detection model for IoT security using K-NN classifier and feature selection*, Multimed. Tools Appl. (2023), 1–19.
11. G. Zhao, C. Ren, J. Wang, Y. Huang, and H. Chen, *IoT intrusion detection model based on gated recurrent unit and residual network*, Peer Peer Netw. Appl. **16** (2023), no. 4, 1887–1899.
12. M. Y. Saeed, J. He, N. Zhu, M. Farhan, S. Dev, T. R. Gadekallu, and A. Almadhor, *An intelligent reinforcement learning–based method for threat detection in mobile edge networks*, Int. J. Netw. Manag. **35** (2025), e2294.
13. S. P. Thirimanne, L. Jayawardana, L. Yasakethu, P. Liyanaarachchi, and C. Hewage, *Deep neural network based real-time intrusion detection system*, SN Comput. Sci. **3** (2022), no. 2, 145–156.
14. A. Basati and M. M. Faghih, *PDAE: efficient network intrusion detection in IoT using parallel deep auto-encoders*, Inform. Sci. **598** (2022), 57–74.
15. F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, *The graph neural network model*, IEEE Trans. Neural Netw. **20** (2008), no. 1, 61–80.
16. T. Bilot, N. El, K. Al Agha, and A. Zouaoui, *Graph neural networks for intrusion detection: a survey*, IEEE Access, 2023, 49114–49139.
17. M. Zhong, M. Lin, C. Zhang, and X. Zeshui, *A survey on graph neural networks for intrusion detection systems: methods, trends and challenges*, Comput. Secur. **141** (2024), 103821.
18. W. Jiang, *Graph-based deep learning for communication networks: a survey*, Comput. Commun. **185** (2022), 40–54.
19. Q. Xiao, J. Liu, Q. Wang, Z. Jiang, X. Wang, and Y. Yao, *Towards network anomaly detection using graph embedding*, (Proc. International Conference on Computational Science, Amsterdam, The Netherlands), 2020, pp. 156–169.
20. J. Zhou, Z. Xu, A. M. Rush, and M. Yu, *Automating botnet detection with graph neural networks*, arXiv preprint, 2020. DOI arXiv:2003.06344
21. A. Zhang, Y. Zhao, C. Zhou, and T. Zhang, *Resacag: a graph neural network based intrusion detection*, Comput. Electr. Eng. **122** (2025), 109956.
22. R. A. Bakar, L. De Marinis, F. Cugini, and F. Paolucci, *FTG-Net-E: a hierarchical ensemble graph neural network for DDoS attack detection*, Comput. Netw. **250** (2024), 110508.
23. Y. Cao, H. Jiang, Y. Deng, W. Jing, P. Zhou, and W. Luo, *Detecting and mitigating DDoS attacks in SDN using spatial-temporal graph convolutional network*, IEEE Trans. Dependable Secure Comput. **19** (2022), no. 6, 3855–3872.
24. W. W. Lo, S. Layeghy, M. Sarhan, M. Gallagher, and M. Portmann, *E-graphsage: a graph neural network based intrusion detection system for IoT*, (Proc. IEEE/IFIP Network Operations and Management Symposium), 2022, pp. 1–9.
25. I. Goodfellow, J. Pouget-Abadie, M. Mirza, X. Bing, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, *Generative adversarial networks*, Commun. ACM **63** (2020), no. 11, 139–144.
26. J. H. Lee and K. H. Park, *GAN-based imbalanced data intrusion detection system*, Pers. Ubiquitous Comput. **25** (2021), 121–128.
27. M. H. Shahriar, N. I. Haque, M. A. Rahman, and M. Alonso, *G-IDS: generative adversarial networks assisted intrusion detection system*, (Proc. IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC)), 2020, pp. 376–385.
28. H. Yang and Y. Zhou, *IDA-GAN: a novel imbalanced data augmentation GAN*, (Proc. 25th International Conference on Pattern Recognition (ICPR)), 2021, pp. 8299–8305.
29. F. Li, H. Shen, J. Mai, T. Wang, Y. Dai, and X. Miao, *Pre-trained language model-enhanced conditional generative adversarial networks for intrusion detection*, Peer Peer Netw. Appl. **17** (2024), no. 1, 227–245.
30. J. Liao, *An intrusion detection model based on improved ACGAN in big data environment*, Security Commun. Netw. (2022), no. 1, 6821174.
31. H. Ding, L. Chen, L. Dong, F. Zhongwang, and X. Cui, *Imbalanced data classification: a KNN and generative adversarial networks-based hybrid approach for intrusion detection*, Future Gener. Comput. Syst. **131** (2022), 240–254.
32. K. Zhang, H. Qin, Y. Jin, H. Wang, and X. Yu, *Auxiliary classifier generative adversarial network assisted intrusion detection system*, (Proc. International Conference on Intelligent Information Processing (IIP)), 2022, pp. 307–311.

33. A. Odena, C. Olah, and J. Shlens, *Conditional image synthesis with auxiliary classifier GANs*, (Proc. International Conference on Machine Learning), 2017, pp. 2642–2651.
34. J. Zhou, G. Cui, H. Shengding, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, *Graph neural networks: a review of methods and applications*, AI Open **1** (2020), 57–81.
35. A. Ylmaz and R. Das, *A novel hybrid approach combining GCN and GAT for effective anomaly detection from firewall logs in campus networks*, Comput. Netw. **259** (2025), 111082.
36. X. Pei, X. Deng, S. Tian, P. Jiang, Y. Zhao, and K. Xue, *A privacy-preserving graph neural network for network intrusion detection*, IEEE Trans. Dependable Secure Comput. **22** (2025), no. 1, 740–756.
37. L. McInnes, J. Healy, and J. Melville, *UMAP: uniform manifold approximation and projection for dimension reduction*, arXiv preprint, 2018. DOI arXiv:1802.03426

AUTHOR BIOGRAPHIES

Tianjing Wang received a BSc degree in mathematics from Nanjing Normal University in 2000, MSc degree in mathematics from Nanjing University in 2002, and PhD in signal and information systems from Nanjing University of Posts and Telecommunications (NUPT) in 2009, all in Nanjing, China. From 2011 to 2013, she was a full-time postdoctoral fellow with the School of Electronic Science and Engineering at NUPT. From 2013 to 2014, she served as a visiting scholar in the Electrical and Computer Engineering Department of the State University of New York at Stony Brook. She is currently an associate professor with the Department of Communication Engineering at Nanjing Tech University. Her research interests include cybersecurity and vehicular networks.

Qi Liu received a BEng degree in computer science from Shenyang Ligong University, Shenyang, China, in 2023. He is currently pursuing an MEng degree in computer science at Nanjing Tech University, Nanjing, China. His research interests include dynamic neural networks and large-language cybersecurity models in cybersecurity. He is a CCF student.

Hang Shen received a PhD with honors in computer science from Nanjing University of Science and Technology, Nanjing, China, in 2015. He worked as a full-time postdoctoral fellow with the Broadband Communications Research (BBCR) Lab, Electrical and Computer Engineering Department, University of Waterloo, Waterloo, ON, Canada, from 2018 to 2019. He is an associate professor with the Department of Computer Science and Technology at Nanjing Tech University, Nanjing, China. His research interests include cybersecurity and vehicular networks. He is an associate editor for *Journal of Information Processing Systems*, *Frontiers in Blockchain*, and *IEEE Access* and a guest editor for *Peer-to-Peer Networking and Applications*. He is a program committee member of the IEEE International Conference on High Performance Computing and Communications (HPCC) 2024 and the Annual International Conference on Privacy, Security, and Trust (PST) 2021. He is an executive committee member of the ACM Nanjing Chapter and supervisory committee member of the CCF Nanjing Chapter.

Xiaokang Luo received a BEng degree in Internet of Things engineering from Changsha University, Changsha, China, and an MEng degree in computer science at Nanjing Tech University, Nanjing, China. Her research interests include graph neural networks, adversarial learning, and network security applications.

Guangwei Bai received BEng and MEng degrees in computer engineering from Xi'an Jiaotong University, Xi'an, China, in 1983 and 1986, respectively, and PhD in computer science from the University of Hamburg, Hamburg, Germany, in 1999. From 1999 to 2001, he worked as a Research Scientist at the German National Research Centre for Information Technology. In 2001, he joined the University of Calgary, Calgary, Canada as a research associate. Since 2005, he has worked at the Nanjing Tech University, Nanjing, China as a professor of computer science. From October to December 2010, he was a visiting professor with the Electrical and Computer Engineering Department at the University of Waterloo, Waterloo, ON, Canada. His research interests include wireless networking and cybersecurity.