

Encrypted Malicious Traffic Detection Fusing Cross-Attention with Lightweight Graph Convolution

Xiaogang Yuan, Minghang Ni, Dezhi An, Jianxin Wan

School of Cyber Security, Gansu University of Political Science and Law, Lanzhou 730070, China

Abstract—With the widespread adoption of encryption technologies, traditional deep packet inspection methods fail to parse encrypted payloads, making it difficult to effectively identify malicious traffic. To address this, this paper proposes a Dual-Level Cross-Attention Lightweight Graph Convolutional Network (DCA-LGCN). This method employs node-level attention to extract intra-flow features and edge-level attention to model inter-flow relationships, introducing a cross-attention mechanism for adaptive fusion of node and edge information. Simultaneously, it incorporates depthwise separable graph convolution and feature bottleneck layers to reduce computational overhead while maintaining detection accuracy. Experiments on the USTC-TFC2016 and CIC-Bell-DNS-EXF-2021 datasets demonstrate that DCA-LGCN significantly outperforms mainstream baseline models in terms of accuracy, inference latency, and model size. These results validate its efficiency and robustness in encrypted malicious traffic detection, achieving an effective balance between precision and efficiency.

Keywords—Encrypted traffic; malicious traffic detection; graph neural network; attention mechanism

I. INTRODUCTION

Network communication security relies on the support of encryption protocols. Google's Transparency Report from April 2024 indicates that 94% of traffic across all its products and services is now encrypted; moreover, web pages loaded via HTTPS in the Chrome browser on Windows systems have fully adopted encrypted transmission[1]. While encryption protects data privacy, it also provides a covert channel for malicious activities. Attackers often leverage encrypted traffic to build DNS covert tunnels, encrypted command and control (C2) communications, and even launch higher-level attacks by exploiting complex ecosystems like the cyber-military-industrial complex[2]. This renders traditional Deep Packet Inspection (DPI) methods ineffective as they cannot parse encrypted payloads[3]. Consequently, achieving accurate detection of encrypted malicious traffic without decryption has become a key challenge in the current cybersecurity landscape.

Traditional detection methods primarily rely on DPI or machine learning models based on handcrafted features. DPI requires analyzing plaintext content, making it ineffective against fully encrypted traffic. Although machine learning-based approaches can utilize statistical features (e.g., packet length, timing), their performance heavily depends on feature engineering and struggles to capture the complex dynamic associations and topological dependencies among traffic entities[4],[5]. In recent years, deep learning models have alleviated the reliance on manual design by automatically learning features. However, architectures like CNNs and RNNs

inherently process sequential or grid data and are inadequate for effectively modeling the graph-structured relationships naturally present in network flows.

Graph Neural Networks (GNNs), through message-passing mechanisms that aggregate neighborhood information, provide a framework for modeling traffic as graph structures (where nodes represent hosts or sessions, and edges represent interaction relationships). This makes GNNs an important development direction for encrypted traffic identification[6]. Nevertheless, existing GNN-based detection methods still face three main limitations: First, node features are often limited to quintuples or basic statistics, lacking the characterization of encrypted semantics and fine-grained behavioral patterns[7]. Second, they often rely on static graph construction, failing to capture the dynamically evolving relationships within traffic interactions[8]. Third, model complexity is high, leading to substantial computational costs that struggle to meet real-time requirements in practical deployment [9].

To address these challenges, this paper proposes a Dual-Level Cross-Attention Lightweight Graph Convolutional Network (DCA-LGCN). The main contributions of this paper are as follows:

- A multi-level graph feature representation method is proposed, fusing statistical, temporal, and protocol semantic features to enhance the expressive power of node and edge features.
- A dual-level attention mechanism (node-level and edge-level) is designed, focusing on intra-flow patterns and inter-session interactions respectively, and achieving adaptive fusion through cross-attention, thereby dynamically capturing complex relationships.
- A lightweight design incorporating depthwise separable graph convolution and feature bottleneck layers is introduced, significantly reducing model parameters and inference latency without compromising detection accuracy.

II. RELATED WORK

Malicious traffic detection is a core task in cybersecurity, aiming to accurately identify attack behaviors from vast amounts of network data. With the prevalence of encryption, traditional detection methods reliant on plaintext analysis have gradually become ineffective. This has driven the evolution of detection techniques from early machine learning models based on handcrafted features to deep learning models capable of automatic representation learning, and more recently to Graph

Neural Networks (GNNs) — particularly Graph Convolutional Networks (GCNs) and their variants — which have garnered widespread attention. These methods exhibit different advantages and limitations in terms of classification accuracy

and computational efficiency. Table I summarizes related work based on machine learning, deep learning, and graph neural networks.

TABLE I. SUMMARY OF RELATED WORK

Method Category	Representative Methods	Advantages	Limitations
Machine Learning	Unsupervised system based on flow interaction graphs [10]	High accuracy and throughput	Strong dependency on feature engineering; difficulty capturing deep temporal and protocol semantic patterns in encrypted traffic
	Improved method based on LightGBM [11]	Alleviates class imbalance	
	Retrieval-augmented framework based on LLMs [12]	Strong detection performance, can adapt to new threats without fine-tuning	
	CertTA framework [13]	Provides provable robustness against various attacks	
Deep Learning	Unsupervised method based on CNN [14]	Detection performance close to supervised methods	Strong at extracting local spatio-temporal features, but weak at modeling global topological relationships
	Fusion of deep learning and traditional machine learning [15]	Focuses on encrypted malicious traffic detection, improves accuracy	
	LSTM network integrated with meta-heuristic algorithms [16]	Achieves high-precision identification of various attacks like DDoS and port scanning	
	Method based on DenseNet [17]	Improves multi-class classification performance	
	Fusion model of 1D-CNN and BiGRU [18]	Mitigates detection issues caused by class imbalance	
	CNN-BiGRU model [19]	Adapts to encrypted and dynamic environments	
Graph Neural Networks	GNN framework integrating neighborhood filtering and stable learning [22]	Enhances robustness in industrial internet scenarios	High computational cost, dependent on graph structure quality
	End-to-end detection model based on GNN [23]	Enhances adaptability in encrypted traffic environments	
	Innovative end-to-end framework Trace2vec [24]	Provides good interpretability	
	Integration of group reversible residual connections and graph isomorphism networks [25]	Improves botnet detection performance	
	Combination of GraphNet framework and Multi-Layer Perceptron [26]	Improves detection performance for complex attack patterns	
	Combination of METIS algorithm and Cluster-GCN framework [27]	Efficiently handles large-scale data	
	Fusion of causal reasoning, conditional generative adversarial networks, and transfer learning [28]	Addresses detection challenges from data heterogeneity and sparsity	
	Integration of Gaussian Mixture Model for probabilistic label correction and conditional GAN for data augmentation [29]	Simultaneously addresses data scarcity and label noise problems	
	Self-supervised learning model based on GNN [5]	Suitable for unsupervised scenarios	

A. Malicious Traffic Detection Methods Based on Machine Learning

Machine learning-based malicious traffic detection methods employ machine learning models, random forests, etc., using single or multi-layer combination techniques to learn features from network traffic, thereby distinguishing between normal and malicious traffic.

For instance, HyperVision [10] is an unsupervised real-time detection system based on flow interaction graphs for identifying unknown encrypted malicious traffic. It boasts high accuracy and throughput but demands substantial computational resources. Hu et al. [11] proposed an improved method based on LightGBM, enhancing detection performance through focal loss and fruit fly optimization algorithms. This method can alleviate class imbalance but incurs high computational costs. MalRAG [12] is a retrieval-augmented framework based on Large Language Models (LLMs) that requires no fine-tuning. It demonstrates strong detection performance but depends on retrieval database quality and

involves significant computational overhead. CertTA [13] is a defense framework providing provable robustness for traffic analysis models, capable of resisting multimodal adversarial attacks. However, it suffers from limitations such as high computational cost and a limited certification scope.

While machine learning methods improve detection performance by fusing models or algorithms and continually attempt to introduce new machine learning concepts and techniques, they also lead to issues like high computational resource consumption, the need for high-quality datasets, strong dependence on base model selection and parameter tuning, and reduced interpretability due to model complexity.

B. Malicious Traffic Detection Methods Based on Deep Learning

The breakthrough of deep learning methods lies in their ability to learn hierarchical feature representations from raw or shallowly processed traffic data, thereby significantly reducing reliance on manual feature engineering. For example, Shumon et al.[14] validated that an unsupervised CNN-based method

achieves performance close to supervised methods in DDoS detection by constructing a high-quality dataset, although data generalization capability remains limited. To further enhance performance, Wang et al.[15] integrated multiple base models to fuse discriminative information from different perspectives, while Dash et al.[16] introduced meta-heuristic algorithms for automatic optimization of LSTM hyperparameters. Kang et al.[17] employed DenseNet to process raw traffic bytes directly, demonstrating the potential for automatically learning effective features from data. Chen et al.[18] and Yang et al.[19] designed a 1D-CNN-BiGRU and a CNN-BiGRU hybrid model respectively, aiming to synergistically capture local spatial patterns and long-range temporal dependencies in traffic data.

Although deep learning models excel at learning spatio-temporal features at the single-flow level and achieve better accuracy and generalization than traditional methods, they primarily focus on temporal and spatial characteristics of traffic and struggle to capture the naturally formed graph-structural relationships between flows.

C. Malicious Traffic Detection Methods Based on Graph Neural Networks

The network environment is inherently a complex graph-structured system. In the broader field of threat analysis, graph structures are also widely used to construct cybersecurity knowledge graphs for organizational attribution and intelligence correlation analysis of Advanced Persistent Threats (APTs)[20],[21]. For network traffic, nodes and edges form rich topological relationships, and malicious traffic often exhibits specific propagation patterns through these relationships. Graph Neural Networks provide an effective means to capture such complex relationships by directly modeling the traffic graph structure.

For example, Zeng et al.[22] proposed a GNN framework integrating neighborhood filtering and stable learning, enhancing the robustness of identifying unknown attacks in Industrial Internet scenarios. Tang et al.[23] reduced reliance on manual feature engineering by modeling traffic sessions as graphs and employing hierarchical pooling and attention mechanisms to automatically learn features. The Trace2vec framework[24] achieved high-accuracy multi-step attack detection through data augmentation, dynamic heterogeneous graph modeling, and a Monte Carlo Tree Search explainer. The XG-BoT framework[25] integrated grouped reversible residual connections with Graph Isomorphism Networks, providing automated forensic support. GraphNet-MLP[26] is a GNN method based on edge classification, aiming to improve detection performance for complex attack patterns. Kexiang et al.[27] utilized the METIS algorithm and Cluster-GCN framework to process large-scale traffic graphs in parallel, employing diagonal enhancement techniques to optimize feature learning and detection in IoT environments. GATL[28] fused causal inference, conditional Generative Adversarial Networks, and transfer learning to address detection challenges arising from data heterogeneity and sparsity. GMM-cGAN[29] performed probabilistic label correction using a Gaussian

Mixture Model (GMM) and data augmentation using a conditional Generative Adversarial Network (cGAN) to collaboratively tackle data scarcity and label noise challenges. NEGSC [5] is a self-supervised detection framework based on GNNs, employing a NEGAT encoder and NEGSC contrastive learning for multi-classification without requiring labeled data, albeit with high computational complexity.

In recent years, Graph Neural Networks have made significant progress in cybersecurity detection fields such as malicious traffic identification, botnet detection, and industrial control system defense. However, current methods still face issues such as insufficient synergistic utilization of the temporal characteristics and topological structure inherent in traffic data, leading to incomplete information exploitation, or high computational complexity in some high-performance models.

III. DCA-LGCN MODEL

To address the aforementioned challenges in Graph Neural Networks, a Dual-Level Cross-Attention Lightweight Graph Convolutional Model is proposed, further strengthening the dynamic interaction between node and edge features and optimizing the computational efficiency of lightweight graph convolution.

The overall workflow is illustrated in Fig. 1. It begins with multi-level feature engineering, extracting and standardizing node and edge features from raw traffic. Subsequently, in the graph construction stage, graph topology is built based on node and edge features using the link homogeneity principle, generating a weighted adjacency matrix and completing graph normalization.

The constructed graph data is then fed into the DCA-LGCN model for inference. The core DCA-LGCN module integrates node-level attention to refine the node's own feature representation, edge-level attention to evaluate and aggregate neighboring edge relationship information, and cross-attention between them to enable deep interaction between node and edge information. This facilitates dynamic weighted learning of key features and important topological relationships, while computational overhead is controlled through lightweight graph convolution design. This optimization for efficiency is crucial for deploying detection models in resource-constrained edge environments, representing a current research trend in cybersecurity[30].

A. Data Preprocessing and Multi-level Feature Extraction

Before feature extraction, targeted preprocessing and segmentation strategies were applied to the datasets to ensure that the generated feature units retained the complete context of attacks while being suitable for graph structure modeling.

For the USTC-TFC2016 dataset, which contains complete bidirectional network flows, consecutive packet sequences sharing the same quintuple were merged into a single network session, with each session representing a complete network interaction process.

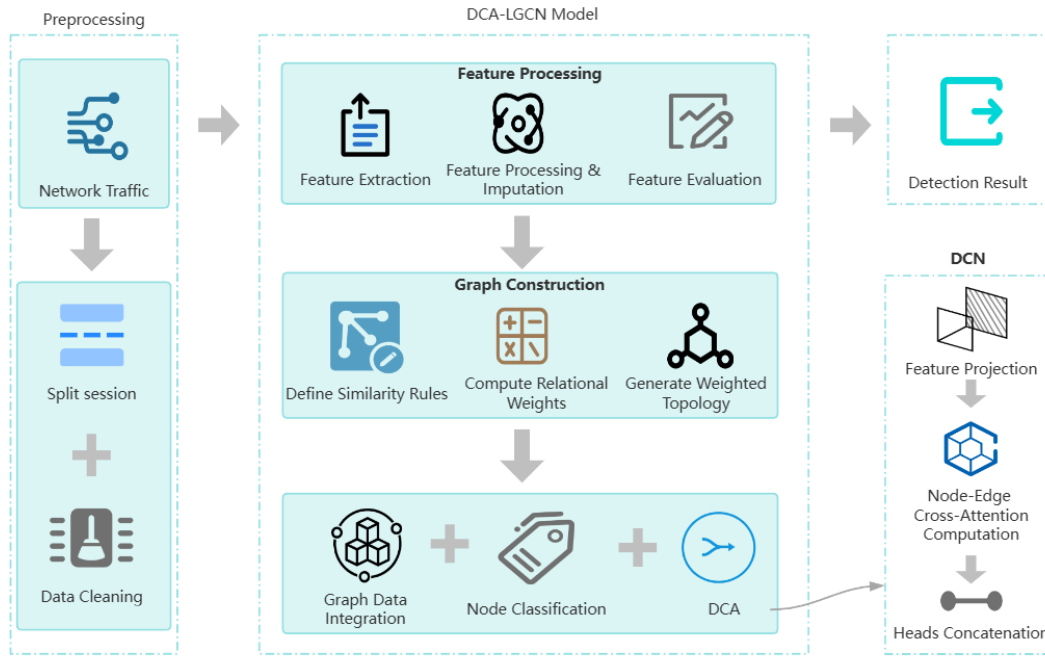


Fig. 1. Workflow of the DCA-LGCN method.

For the CIC-Bell-DNS-EXF-2021 dataset, a sliding window approach was employed for segmentation. A window size of 100 consecutive DNS packets with a sliding stride of 100 was set, generating mutually independent analysis windows, with each window serving as a basic feature unit:

$$W_k = \{p_{1+100k}, p_{2+100k}, \dots, p_{100+100k}\}, \quad k=0, 1, \dots, \lfloor \frac{N-100}{100} \rfloor \quad (1)$$

where, W_k represents the k -th window, p_i is the packet sequence, N is the total number of packets, and both window size and step are 100. During this process, absolute timestamps of packets are removed to prevent the model from overfitting to specific collection time points, thereby enhancing the model's ability to learn generalized features of attack behavior.

To effectively identify malicious behavior from the differentially preprocessed sessions described above, the key lies in constructing more discriminative feature representations for them. To better capture covert malicious behavior, extracted features are extended from static quintuples to multi-level graph structure features, including session duration, packet length statistics, and protocol interaction complexity, aiming to finely characterize the intrinsic behavioral attributes of each analysis unit (node).

The session duration formula calculates the temporal span of a session to capture the persistence of the interaction. The formula directly computes the difference as a feature, providing a basis for subsequent temporal analysis.

$$T_{\text{duration}} = t_{\text{last}} - t_{\text{first}} \quad (2)$$

where, t_{first} and t_{last} are the timestamps of the first and last packet in the session, respectively.

The formulas for mean, variance, and standard deviation are as follows, jointly characterizing the fluctuation

characteristics of traffic load. These statistics are efficiently computed to ensure feature scalability.

$$\begin{aligned} \text{Mean: } \mu_L &= \frac{1}{N} \sum_{i=1}^N L_i \\ \text{Variance: } \sigma_L^2 &= \frac{1}{N} \sum_{i=1}^N (L_i - \mu_L)^2 \\ \text{Standard Deviation: } \sigma_L &= \sqrt{\sigma_L^2} \end{aligned} \quad (3)$$

where, L_i is the length of the i -th packet, and N is the number of packets.

$$w_{\text{time}}^{(ij)} = \exp\left(-\frac{|t_i - t_j|}{\tau}\right) \quad (4)$$

where, t_i and t_j are the start times of sessions i and j , and τ is the decay factor.

The behavioral correlation formula quantifies node similarity from a content dimension. The cosine similarity formula measures the directional consistency of node feature vectors, identifying session pairs with similar behavioral patterns:

$$\text{sim}(i, j) = \frac{f_i \cdot f_j}{\|f_i\| \|f_j\|} \quad (5)$$

where, f_i and f_j are node feature vectors.

The IP co-occurrence frequency formula counts the interaction frequency of IP pairs, normalized to serve as edge weight, effectively mitigating graph sparsity.

$$w_{ij}^{\text{IP}} = \frac{\text{count}(IP_i, IP_j)}{\max(\text{count}(IP))} \quad (6)$$

The numerator is the co-occurrence count of the IP pair, and the denominator is the normalization factor.

B. Lightweight Graph Structure Modeling Based on Link Homophily

Building upon the completed data preprocessing and multi-level feature extraction, a weighted graph topology is constructed through the lightweight graph structure modeling based on link homophily illustrated in Fig. 2, to more accurately capture the complex associations within network traffic.

Link homophily originates from a fundamental principle in social network analysis: similar nodes are more likely to connect. In the context of malicious traffic detection, this implies that traffic sessions sharing common behavioral

characteristics are more likely to form edge connections, naturally reflecting the propagation paths of network attacks. By quantifying the multi-dimensional similarity between nodes as edge weights, an efficient and expressive graph structure is established. Its advantages include: enhancing graph connectivity and robustness to avoid detection blind spots caused by sparse connections; strengthening the ability to identify covert threats (e.g., DNS tunnels) in encrypted traffic through a dynamic weighting mechanism that adaptively learns key topological features. Simultaneously, the lightweight design ensures the model maintains high inference efficiency even in resource-constrained environments, laying the foundation for subsequent Graph Neural Network processing.

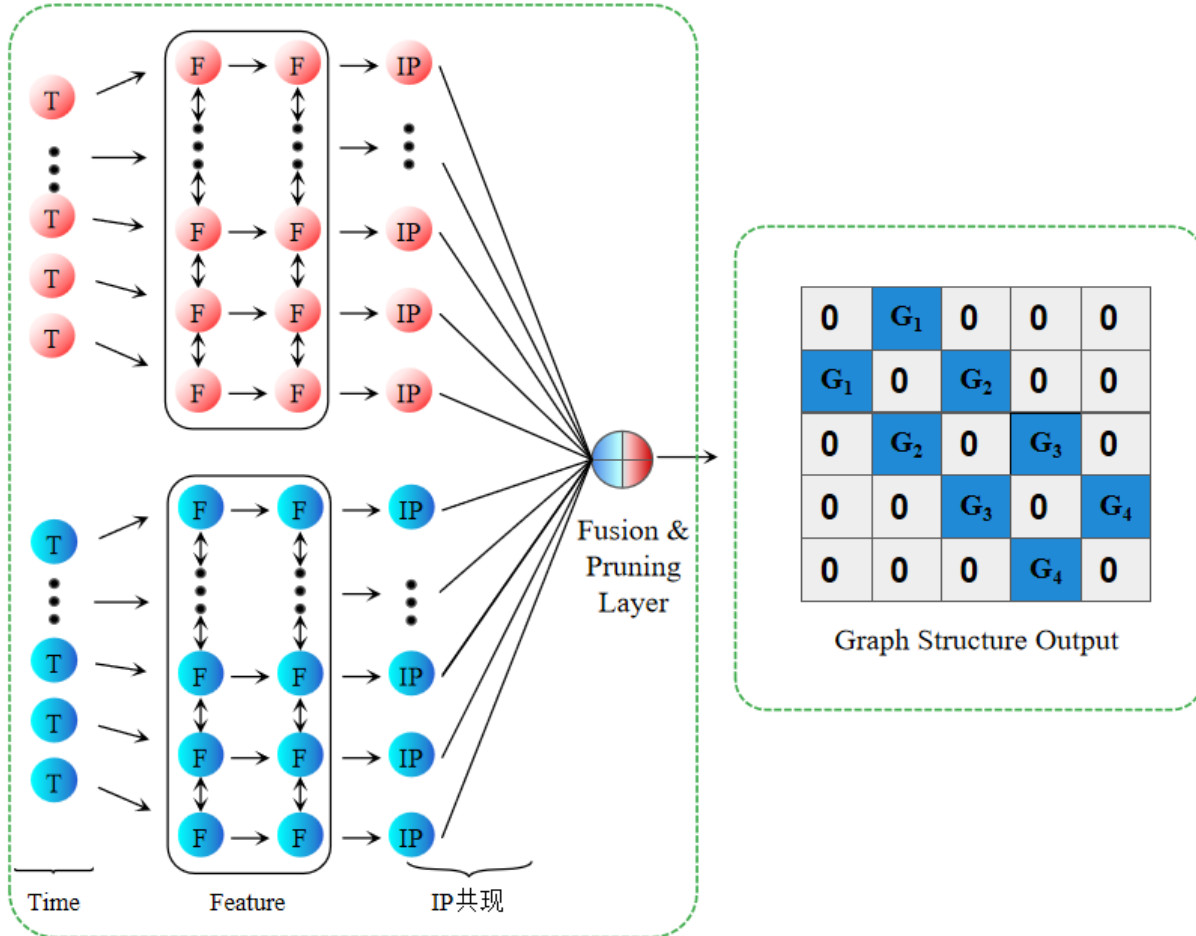


Fig. 2. Lightweight graph structure modeling based on link homophily.

Fig. 2 illustrates the lightweight graph structure modeling method based on link homophily, a core component of the DCA-LGCN model, aiming to construct an efficient weighted graph topology by quantifying the similarity between nodes. The principle of link homophily, derived from social network analysis, emphasizes that similar nodes are more prone to connect. In malicious traffic detection, this means traffic sessions with common behavioral characteristics (e.g., protocol patterns, temporal proximity, or IP interactions) are prioritized in forming edge connections, thereby naturally mapping attack propagation paths.

Fig. 2 specifically depicts the node and edge construction process: graph construction employs a multi-modal edge fusion strategy, integrating temporal edges, content edges, and feature edges. Edge weights are calculated based on multi-dimensional features (e.g., temporal proximity, behavioral correlation, and IP co-occurrence frequency). The fusion weight is computed as:

$$G_{ij} = \alpha \cdot w_{ij}^{\text{time}} + \beta \cdot w_{ij}^{\text{feature}} + \gamma \cdot w_{ij}^{\text{IP}} \quad (7)$$

where, $\alpha + \beta + \gamma = 1$. Edges with weights above a threshold are retained.

We set $\alpha=0.3$, $\beta=0.4$, and $\gamma=0.3$ for USTC-TFC2016, with an edge retention threshold of 0.2, determined via grid search on the validation set. For CIC-Bell-DNS-EXF-2021, an adaptive strategy is used where edges with positive fused weights are retained. In both cases, the threshold is chosen to balance graph connectivity and sparsity by preserving strong associations while discarding weak connections.

To prevent data leakage, the graph topology (similarity thresholds and edge weights) was constructed strictly using only the training set. Test nodes were isolated during the training phase and evaluated as independent subgraphs during inference. Furthermore, to prevent the model from relying on specific IP addresses as a shortcut, we applied anonymization to the IP features before graph construction.

This graph modeling method significantly enhances graph connectivity and robustness, avoiding detection blind spots due to sparse connections. Its lightweight structure ensures efficient inference in resource-constrained environments, lays the topological foundation for subsequent GNN processing, and strengthens the ability to identify covert threats (e.g., DNS tunnels) within encrypted traffic.

C. Framework and Algorithm of the DCA-LGCN Model

The overall framework of the DCA-LGCN model comprises three core components: a node-edge dual-level attention mechanism, a cross-attention fusion layer, and a lightweight graph convolution module. As shown in Fig. 3, the model framework adopts a bottom-up data flow design, forming a complete closed loop from feature input to classification output. The workflow begins with multi-level feature input. The node-level GCN path extracts intra-flow feature patterns, while the edge-level GCN path captures inter-session interaction relationships. The two paths achieve deep information interaction in the cross-attention layer, and the final detection result is output by the classifier.

As shown in Fig. 3, the model can be divided into the Input Layer, Dual-Path Attention Paths, Cross Fusion Layer, and Classification Output Layer. The Input Layer includes the node feature matrix and edge features. Node features are derived from statistical and protocol semantic information of traffic sessions, while edge features quantify the interaction relationships between sessions.

The model processes the input through two parallel paths: the left node-level attention path employs lightweight GCNs to focus on intra-flow feature patterns; the right edge-level attention path captures dynamic inter-flow associations through an edge feature update module. The outputs of the two paths undergo deep interaction in the Cross-Attention Fusion Layer, where a multi-head scaled dot-product attention mechanism is used to achieve adaptive weighted fusion of node and edge information. The fused features are then aggregated hierarchically, and the final prediction is generated by the classifier.

This hierarchical design and modular lightweight components effectively control computational overhead while ensuring the model's ability to perceive complex topological structures. The specific computational processes for each core component of the framework are defined as follows.

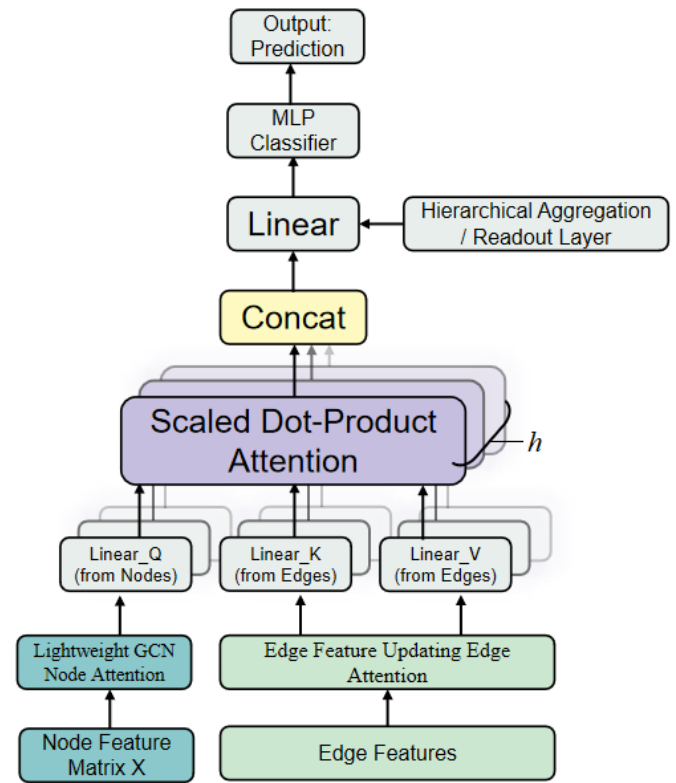


Fig. 3. Framework of the DCA-LGCN model.

1) Feature bottleneck layer performs dimensionality reduction on input features:

$$\begin{aligned} X_{\text{hidden}} &= \text{ReLU}(\text{Dropout}_{0.3}(W_1 \cdot X + b_1)) \\ X_{\text{bottleneck}} &= W_2 \cdot X_{\text{hidden}} + b_2 \end{aligned} \quad (8)$$

where, $X \in \mathbb{R}^{N \times F}$ is the input feature matrix, N is the number of nodes, F is the input feature dimension. This layer enhances robustness through the ReLU activation function and Dropout regularization.

2) Node-level depthwise separable graph convolution layer reduces parameters by decomposing dense weight matrices. It reduces the parameter count of standard GCN from $O(d^2)$ to $O(d)$, where d is the feature dimension, thereby significantly lowering model complexity and computational cost. Its computation is divided into two steps:

$$\begin{aligned} \text{Depthwise Aggregation Transform:} \\ H_{\text{agg}}^{(1)} &= \hat{A} \cdot H^{(1)} \cdot W_{\text{depth}} \\ \text{Pointwise Transform:} \\ H^{(1+1)} &= \text{ReLU}(H_{\text{agg}}^{(1)} \cdot W_{\text{point}}) \end{aligned} \quad (9)$$

The key innovation lies in the cross-attention layer, which employs a bidirectional interaction mechanism between nodes and edges—particularly suited for jointly modeling intra-session features and inter-session relations.

3) To initialize edge features to encode relationships between nodes, they are initialized by concatenating adjacent node features and applying linear projection:

$$E_{ij}^{(0)} = W_e \cdot [H_i || H_j] + b_e \quad (10)$$

where, $[\cdot || \cdot]$ denotes feature concatenation, and $W_e \in \mathbb{R}^{256 \times 128}$, $b_e \in \mathbb{R}^{128}$ project the 256-dimensional concatenated features into a 128-dimensional hidden space.

4) The Edge-Level Depthwise Separable Graph Convolution Layer has a structure similar to the node-level one. Its function is to efficiently propagate edge features within the graph structure:

$$\begin{aligned} &\text{Depthwise Aggregation Transform:} \\ &E_{agg}^{(1)} = \hat{A} \cdot E^{(1)} \cdot W_{\text{depth_edge}} \\ &\text{Pointwise Transform:} \\ &E^{(l+1)} = \text{ReLU}(E_{agg}^{(1)} \cdot W_{\text{point_edge}}) \end{aligned} \quad (11)$$

This layer ensures edge information propagation remains lightweight.

5) The core innovation of the model lies in the Cross-Attention Layer, designed to achieve adaptive fusion of node and edge features. In malicious traffic detection, the maliciousness of a session (node) may depend not only on its own content but also highly on which other sessions (edges) it is associated with. Therefore, the model's cross-attention specifically designs a bidirectional interaction mechanism between nodes and edges, making it more suitable for the collaborative modeling of intra-session features and inter-session relationships in traffic analysis. Compared to simple feature concatenation, cross-attention dynamically computes correlation weights between node and edge features, thereby more effectively fusing heterogeneous information, avoiding potential redundancy or conflict from direct concatenation, and achieving a synergistic enhancement effect.

This layer employs a multi-head mechanism. The attention weights from nodes to edges are calculated using the following formula, allowing nodes to attend to the most relevant edge information, achieving information complementarity:

$$\alpha_{ij} = \text{softmax} \left(\frac{Q_{\text{node}(i)} \cdot K_{\text{edge}(j)}^T}{\sqrt{d}} \right) \quad (12)$$

Edge-to-node attention weights:

$$\beta_{ji} = \text{softmax} \left(\frac{Q_{\text{edge}(j)} \cdot K_{\text{node}(i)}^T}{\sqrt{d}} \right) \quad (13)$$

First, linear projection layers map node and edge features into Query (Q), Key (K), and Value (V) spaces, respectively. Q, K, V are reshaped into multi-head forms. Attention weights are computed via scaled dot-product, and an edge index mask is applied to ensure only connected nodes participate in the computation. The final output is enhanced with LayerNorm and residual connections for stability.

6) Classifier performs final classification on the fused node and edge features:

$$\begin{aligned} z &= \text{ReLU}(\text{Dropout}_{0.3}(W_1 \cdot [H_{\text{final}} || E_{\text{final}}] + b_1)) \\ \hat{y} &= \sigma(W_2 \cdot z + b_2) \end{aligned} \quad (14)$$

where, $W_1 \in \mathbb{R}^{256 \times 128}$, $b_1 \in \mathbb{R}^{128}$, $W_2 \in \mathbb{R}^{128 \times 1}$, $b_2 \in \mathbb{R}$, σ is the Sigmoid activation function, and $\hat{y} \in [0,1]$ is the malicious probability of the node.

7) To address class imbalance in the data and prevent overfitting, weighted binary cross-entropy loss is employed to handle class imbalance:

$$\mathcal{L} = -\frac{1}{N} \sum_{i=1}^N [w_{y_i} (y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i))] + \lambda ||\theta||_2^2 \quad (15)$$

where, w_{y_i} is the class weight calculated based on the positive/negative sample ratio in the training set, $\lambda = 0.001$ is the L2 regularization coefficient, and label smoothing is supported.

IV. EXPERIMENTAL SETUP

A. Dataset Description

Two public and representative encrypted traffic detection datasets were selected for the experiments: the USTC-TFC2016 dataset and the CIC-Bell-DNS-EXF-2021 dataset. These target mixed attack traffic and DNS covert exfiltration attack scenarios respectively, effectively testing the model's detection capabilities across different attack types and levels of stealth.

USTC-TFC2016 Dataset: Released by the University of Science and Technology of China (USTC) in 2016, this dataset aims to provide standardized data resources for network traffic classification and malicious traffic detection research. Collected from real network environments, it contains normal traffic from various common application types as well as malicious traffic generated by real malware samples, covering a comprehensive range of attack types. It is suitable for evaluating a model's comprehensive detection capability in diverse traffic backgrounds.

CIC-Bell-DNS-EXF-2021 Dataset[31]: Jointly released by the Canadian Institute for Cybersecurity (CIC) and Bell Labs in 2021, this dataset focuses on detecting data exfiltration attacks within the DNS domain. It simulates various covert channels and data leakage attacks based on the DNS protocol, aiming to advance research on detecting advanced threats in encrypted DNS traffic. Its design particularly emphasizes the stealthiness and protocol compliance of attacks, posing higher demands on a detection model's feature extraction and behavioral analysis capabilities.

B. Evaluation Metrics

Several evaluation metrics were chosen to comprehensively assess the performance of the proposed DCA-LGCN model, as detailed in Table II.

TABLE II. EVALUATION METRICS

Metric	Formula	Meaning
Precision	$Pre = \frac{TP}{TP + FP}$	The proportion of samples predicted as malicious that are truly malicious, measuring the reliability of predictions.
Accuracy	$Acc = \frac{TP + TN}{TP + TN + FP + FN}$	The proportion of correctly classified samples among all samples, measuring overall performance.
Recall	$Recall = \frac{TP}{TP + FN}$	The proportion of all truly malicious samples that are correctly identified, measuring the model's ability to discover threats.
F1-measure	$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$	The harmonic mean of Precision and Recall, used to balance the two.

Here, TP, TN, FP, FN represent the number of true positive, true negative, false positive, and false negative samples, respectively.

C. Parameter Settings

Key hyperparameters of the model were determined through multiple adjustments to achieve the best balance between performance and efficiency. The final settings are as follows: the number of graph convolution layers is 2, with 128 hidden units per layer; the number of cross-attention heads is set to 4; the batch size is 32; training is conducted for 100 epochs with early stopping to prevent overfitting.

For each dataset, we adopt a 70/15/15 train/validation/test split at the session level using random seed 42. Early stopping is applied on the validation loss with patience 15 for USTC-TFC2016 and 10 for CIC-Bell-DNS-EXF-2021. All experiments are repeated five times with different random seeds, and we report the mean values. Baselines are re-implemented following the original descriptions.

Inference latency is measured on a single NVIDIA RTX 3090 GPU with batch size 32, averaged over 100 forward passes after 10 warm-up iterations; the reported time excludes data preprocessing and graph construction.

The optimizer used is AdamW with an initial learning rate of 0.001 and weight decay of 0.01. The learning rate scheduler employs linear warm-up for the first 5 epochs, increasing the learning rate from 0.1 times the initial value to the full value, followed by a cosine annealing phase with a period length $T_0=50$ and a minimum learning rate of 10^{-6} . After each period, the learning rate is restarted and the period is lengthened to avoid local optima.

D. Baseline Models

To comprehensively evaluate the performance of the proposed DCA-LGCN model, five advanced baseline models were selected for comparison. These models include methods based on graph convolutional networks, lightweight fast classification methods, and techniques combining generative adversarial networks for data augmentation, aiming for a comprehensive comparison across dimensions such as detection accuracy, time efficiency, and model complexity. The specific baselines are as follows:

EC-GCN Model: Employs a multi-scale graph convolutional network framework, representing encrypted traffic flows as graph structures to dynamically learn the spatio-temporal dependencies of packet length sequences, thereby improving classification robustness and accuracy.

FastTraffic Model: A lightweight method for fast classification of encrypted traffic, using N-gram feature embedding and a three-layer MLP classifier. It emphasizes small input size and computational efficiency, making it suitable for deployment on resource-constrained devices.

DE-GNN Model: Leverages temporal embedding to capture traffic sequence patterns and combines spatial graph structure modeling to capture dependencies, achieving high accuracy and generalization in fine-grained encrypted traffic classification.

GMM-cGAN Model: Integrates Gaussian Mixture Models for probabilistic label correction and Conditional Generative Adversarial Networks for targeted data augmentation.

TMC-GCN Model: Utilizes flow mapping graphs to construct traffic topology, strengthens inter-packet sequential and hop-order edge relationships, and learns spatio-temporal features through Graph Convolutional Networks to achieve fine-grained encrypted traffic classification.

V. EXPERIMENTAL RESULTS AND ANALYSIS

A. Classification Result Analysis of the DCA-LGCN Model

To evaluate the binary classification performance of the proposed DCA-LGCN model, experiments were conducted on two public network traffic datasets, and comparisons were made against the five baselines. Fig. 4 illustrates the accuracy-efficiency trade-offs of the six malicious traffic detection models on the USTC-TFC2016 and CIC-Bell-DNS-EXF-2021 datasets, respectively. The X-axis represents model accuracy (%), the Y-axis represents per-node inference latency (ms), and the bubble size represents model volume (MB). The green translucent rectangular area indicates the Pareto-optimal region: high accuracy and low latency.

As shown in the left side of Fig. 4, on the USTC-TFC2016 dataset containing mixed attack types, the DCA-LGCN model (red bubble) achieves the best accuracy-efficiency trade-off among all compared models: accuracy reaches 99.96%, inference latency is 0.26 ms, and model volume is 1.18 MB. In contrast, while EC-GCN has the smallest model volume, its latency is too high; FastTraffic shows balanced performance in latency and accuracy but has a larger model volume; GMM-cGAN has the lowest latency but significantly lower detection accuracy.

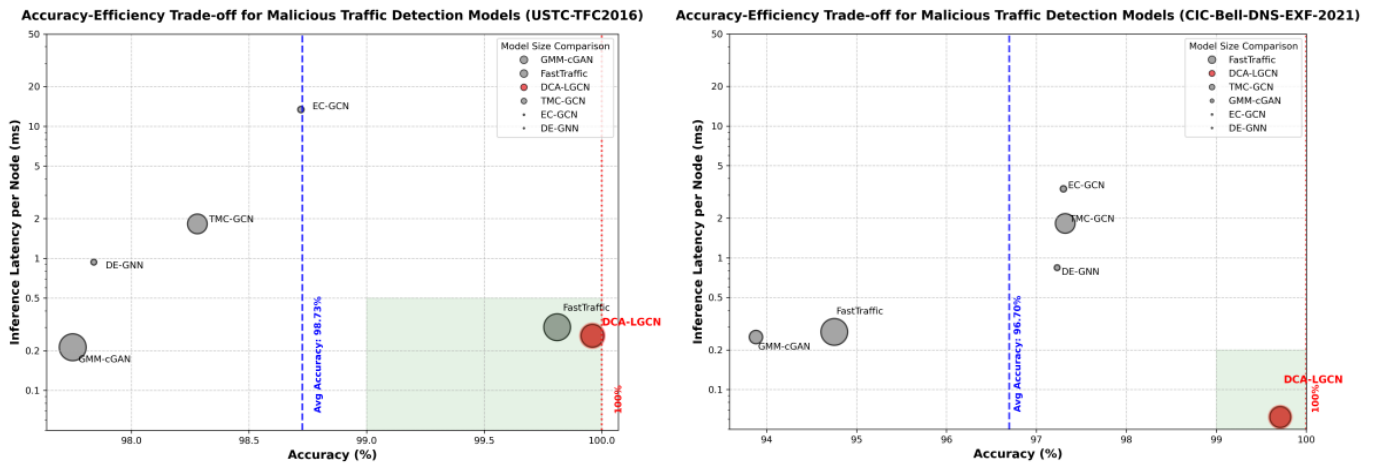


Fig. 4. Comprehensive accuracy-efficiency comparison of six models.

As shown on the right side of Fig. 4, on the CIC-Bell-DNS-EXF-2021 dataset focused on DNS covert exfiltration attacks, DCA-LGCN demonstrates more pronounced advantages: it exhibits the lowest inference latency, moderate model volume, and simultaneously maintains the highest detection accuracy. This performance improvement is particularly critical for DNS tunnel attack detection, as DNS exfiltration attacks often require millisecond-level detection response times to effectively block data leakage effectively. Other baseline models exhibit varying degrees of performance degradation on this dataset, indicating their limited adaptability to novel covert attacks.

Synthesizing the experimental results from both datasets, DCA-LGCN not only demonstrates excellent detection performance across different attack types and levels of stealth but also achieves the optimal balance across the three dimensions of latency, accuracy, and model complexity.

B. Performance Comparison of the DCA-LGCN Model

To verify whether the proposed DCA-LGCN model achieves better malicious traffic detection capability, it was compared with five methods: EC-GCN, FastTraffic, DE-GNN, GMM-cGAN, and TMC-GCN.

To visually compare the comprehensive performance of each model across the two datasets, the performance heatmap in Fig. 5 is presented. The color depth indicates the performance score. It can be observed that in the performance heatmap for the USTC-TFC2016 dataset (left) and the CIC-Bell-DNS-EXF-2021 dataset (right), the DCA-LGCN model achieves the highest numerical values (darkest color) across all evaluation metrics (Accuracy, Precision, Recall, F1-score), preliminarily validating its effectiveness.

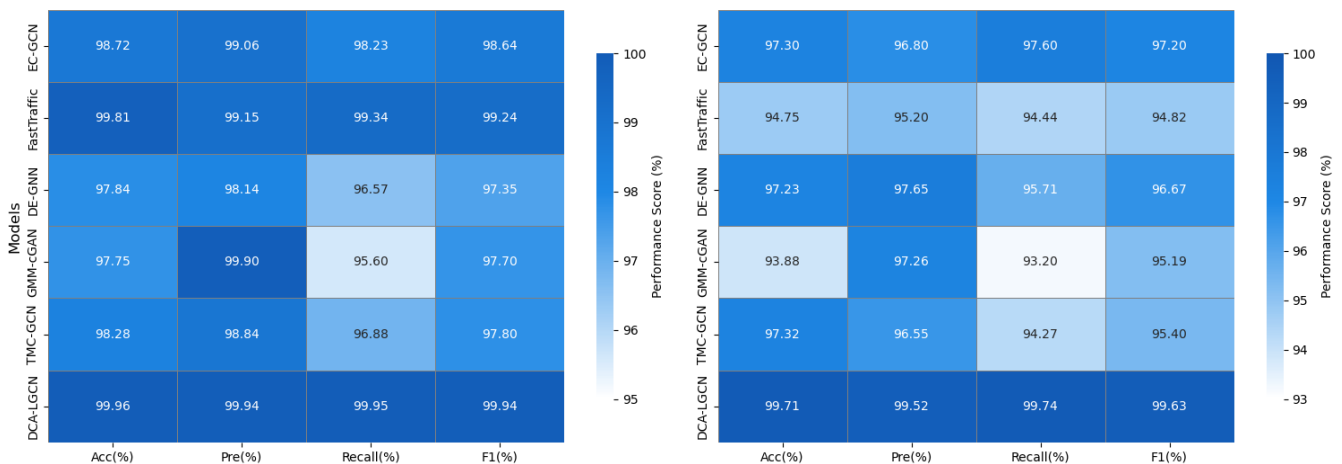


Fig. 5. Performance heatmap.

To quantify the advantage of DCA-LGCN over the baseline models, Fig. 6 further presents a performance gap heatmap. This figure calculates the difference between DCA-LGCN and each baseline model for each metric and visualizes it using a divergent color scheme: positive values are shown in green, and negative values in gray. As seen in Fig. 6, almost all cells

are green, indicating that DCA-LGCN outperforms the baseline models in the vast majority of comparisons. Especially on the CIC-Bell-DNS-EXF-2021 dataset, its advantage in Recall is most significant, which is crucial for detecting covert attacks in encrypted malicious traffic.

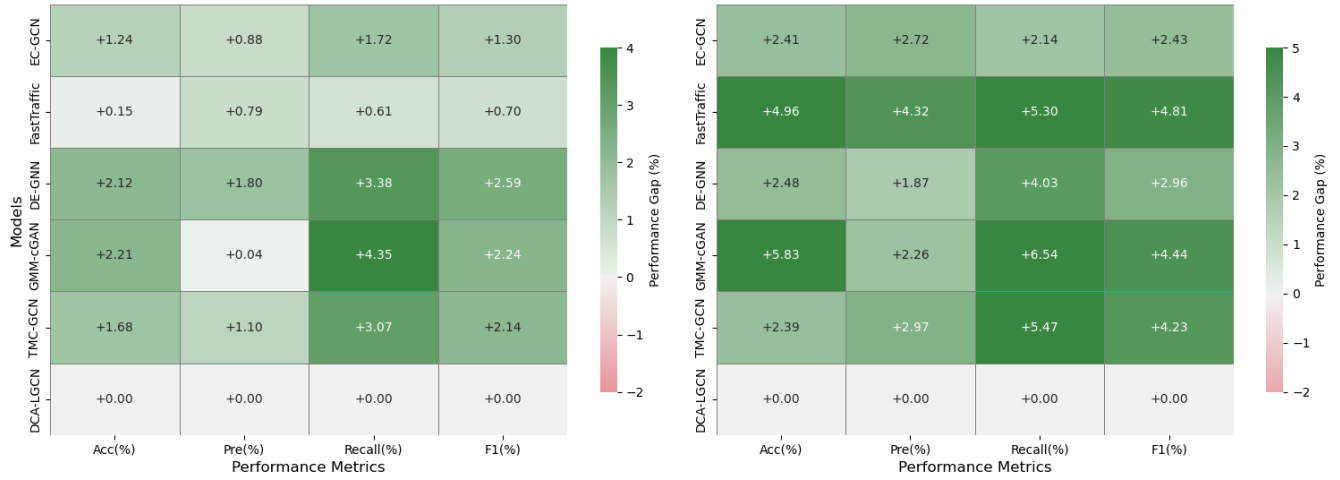


Fig. 6. Performance gap heatmap.

The numerical analysis in Table III corroborates the above observations. On the USTC-TFC2016 dataset, the average advantage of DCA-LGCN over other models ranges from 1.48% to 2.63%. On the more challenging CIC-Bell-DNS-EXF-2021 dataset, its average advantage is more pronounced, ranging

from 2.83% to 4.70%. Its FPR and FNR are consistently below 0.4% and 0.2%, respectively, across both datasets. AUC-ROC scores are 0.9994 on USTC and 0.99991 on CIC, confirming near-perfect class separability.

TABLE III. COMPARISON OF SIX MALICIOUS TRAFFIC DETECTION METHODS

Model	Dataset	Acc(%)	Pre(%)	Recall(%)	F1(%)	FPR(%)	FNR(%)
EC-GCN	USTC-TFC2016	98.72	99.06	98.23	98.64	0.48	1.77
	CIC-Bell-DNS-EXF-2021	97.30	96.80	97.60	97.20	2.15	2.47
FastTraffic	USTC-TFC2016	99.81	99.15	99.34	99.24	0.44	0.66
	CIC-Bell-DNS-EXF-2021	94.75	95.20	94.44	94.82	3.23	5.64
DE-GNN	USTC-TFC2016	97.84	98.14	96.57	97.35	0.95	3.43
	CIC-Bell-DNS-EXF-2021	97.23	97.65	95.71	96.67	1.56	4.23
GMM-cGAN	USTC-TFC2016	97.75	99.90	95.60	97.70	0.05	4.4
	CIC-Bell-DNS-EXF-2021	93.88	97.26	93.20	95.19	1.79	6.88
TMC-GCN	USTC-TFC2016	98.28	98.84	96.88	97.80	0.59	3.12
	CIC-Bell-DNS-EXF-2021	97.32	96.55	94.27	95.40	2.27	5.82
DCA-LGCN	USTC-TFC2016	99.96	99.94	99.95	99.94	0.03	0.05
	CIC-Bell-DNS-EXF-2021	99.71	99.52	99.74	99.63	0.36	0.18

C. Ablation Study

To verify the impact of each module on the proposed DCA-LGCN model, a series of ablation experiments were conducted. The results are shown in Table IV.

The GCN-Base model, which removes all attention mechanisms, shows the most significant performance drop. This indicates that simple graph convolution with feature concatenation struggles to capture the complex multi-level information interactions in malicious traffic.

The GCN-Node model retains only node-level attention. While it can focus on the intrinsic features of individual flows, its performance is limited due to the loss of inter-flow relationship information. Conversely, the GCN-Edge model models only edge-level relationships while neglecting in-depth

exploration of the node's own features, which impacts classification tasks that rely on intrinsic traffic attributes (e.g., packet length sequences).

The GCN-Node+Edge model includes both node and edge attention but uses simple feature concatenation instead of cross-attention fusion. Although its performance is better than the previous variants, it still lags behind the full model. Concatenation cannot dynamically model node-edge dependencies, highlighting the necessity of the adaptive fusion mechanism.

On the CIC-Bell-DNS-EXF-2021 dataset, removing node attention (GCN-Edge) degrades performance more (94.58%) than removing edge attention (GCN-Node, 95.36%), indicating that intra-session features are more critical for DNS exfiltration detection.

TABLE IV. IMPACT OF DIFFERENT MODULES ON THE ACCURACY OF THE DCA-LGCN MODEL

Model	USTC-TFC2016(%)	CIC-Bell-DNS-EXF-2021(%)
GCN-Base	96.48	92.74
GCN-Node	97.05	95.36
GCN-Edge	96.62	94.58
GCN-Node+Edge	97.93	96.85
DCA-LGCN	99.96	99.71

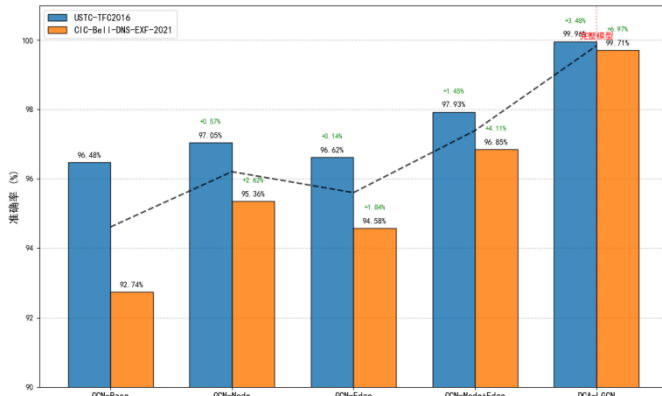


Fig. 7. Detection accuracy comparison.

As shown in Fig. 7, by comparing model variants with different module combinations, the full DCA-LGCN model achieves the highest accuracy on both the USTC-TFC2016 and

CIC-Bell-DNS-EXF-2021 datasets, significantly outperforming other simplified variants. The model performance shows an upward trend as the node-level and edge-level attention modules are gradually introduced, validating the effectiveness of the dual-level attention mechanism design. Furthermore, the model exhibits greater performance gain on the more challenging CIC dataset, indicating its strong adaptability to complex encrypted malicious traffic as well.

D. Time Efficiency of the DCA-LGCN Model

Beyond detection accuracy, time efficiency is another key metric for assessing the practicality of malicious traffic detection methods. An efficient model requires low inference latency and a compact model size to meet real-time or near-real-time detection demands in actual network environments. As shown in Table V, it can be clearly seen that the DCA-LGCN model demonstrates significant advantages in terms of time efficiency and model volume.

TABLE V. TIME COST OF SIX MALICIOUS TRAFFIC DETECTION METHODS

Model	Time Cost	USTC-TFC2016	CIC-Bell-DNS-EXF-2021
DCA-LGCN	Per-Node Inference Latency Model Size	0.26 ms 1.18mb	0.062 ms 1.0mb
EC-GCN	Per-Node Inference Latency Model Size	13.39ms 0.9mb	3.34 ms 0.9mb
FastTraffic	Per-Node Inference Latency Model Size	0.302 ms 1.64mb	0.275 ms 1.64mb
DE-GNN	Per-Node Inference Latency Model Size	0.939 ms 0.076mb	0.842 ms 0.076mb
GMM-cGAN	Per-Node Inference Latency Model Size	0.213 ms 1.656mb	0.251 ms 0.41mb
TMC-GCN	Per-Node Inference Latency Model Size	1.824 ms 0.885mb	1.824 ms 0.885mb

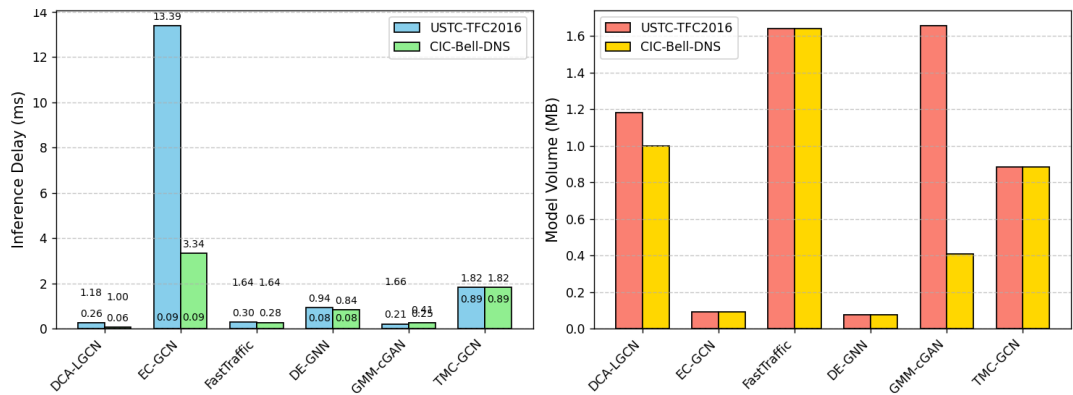


Fig. 8. Bar charts of time efficiency and model size.

Fig. 8 compares the time efficiency of each model using grouped bar charts. The left chart shows that DCA-LGCN's inference latency on both datasets is significantly lower than most baseline models, especially reaching 0.06 ms on CIC-Bell-DNS-EXF-2021, reflecting its efficient inference capability. The right chart indicates that DCA-LGCN's model size remains at a moderate level, maintaining lightweight without incurring a significant increase in volume. Overall, DCA-LGCN achieves a favorable balance between latency and size, validating the effectiveness of its lightweight design.

Compared to other models, DCA-LGCN achieves the best balance between inference latency and model volume, primarily due to its innovative design of depthwise separable graph convolution and hierarchical graph learning mechanisms. Specifically, its efficiency stems from two aspects:

- Depthwise Separable Graph Convolution decomposes the standard graph convolution operation into independent spatial and feature aggregation steps, greatly reducing computational complexity and parameter count. This allows the model to maintain strong representational capacity while achieving lightweight and fast inference.
- Hierarchical Graph Learning Mechanism, where the collaboration between node-level and edge-level GCNs allows the model to learn graph structure information from different granularities in parallel, avoiding redundant computations and further optimizing the computational efficiency of both training and inference processes.

VI. CONCLUSION

This paper proposed a Dual-Level Cross-Attention Lightweight Graph Convolutional Network model named DCA-LGCN. The model constructs an information-rich graph structure through multi-level feature engineering and innovatively introduces a dual-level attention mechanism (node-level and edge-level). The node-level attention focuses on intra-session features, while the edge-level attention captures dynamic inter-session associations. Both are adaptively fused through a cross-attention layer, effectively identifying complex dependencies within network traffic. Combined with lightweight designs such as depthwise separable graph convolution, the model significantly reduces computational and storage overhead while enhancing inference efficiency without compromising high accuracy.

Experiments on the USTC-TFC2016 and CIC-Bell-DNS-EXF-2021 datasets demonstrate that DCA-LGCN exhibits excellent comprehensive performance and time efficiency in malicious traffic detection, validating the effectiveness of its mechanisms and architecture.

Although the model performs well, the following limitations require further improvement:

- Extension from Binary to Multi-class Classification: The current work focuses on binary classification. Future work could aim for fine-grained identification of specific attack types in a multi-class scenario.

- Incorporation of Few-Shot Learning: Few-shot learning can effectively address the issue of insufficient samples for rare attack categories in malware detection[32],[33].
- Enhancing Adversarial Robustness: Improving adaptability to dynamic unknown attacks.

In summary, the DCA-LGCN model achieves a favorable balance between precision and efficiency by combining relationship modeling via graph structures with dynamic feature learning through dual-level attention. It holds positive significance for advancing lightweight, deployable malicious traffic detection technologies.

CONFLICTS OF INTEREST

The authors declare no conflicts of interest to report regarding the present study.

ACKNOWLEDGMENT

This study was supported by the National Natural Science Foundation of China (Grant No. 62562005), the University-Level General Scientific Research Project of Gansu University of Political Science and Law (Grant No. GZF2025XZD27), the Lanzhou Science and Technology Plan Project (Grant No. 2023-1-53), and the Anning County Science and Technology Plan Project (Grant No. 024-JB-5). The authors would like to express their sincere gratitude to the funding organizations for their support of this research.

REFERENCES

- [1] Zhang and Z. Sun, "Challenges and Advances in Analyzing TLS 1.3-Encrypted Traffic: a Comprehensive Survey," *Electron.*, Vol. 13, No. 20, Art. No. 4000, 2024.
- [2] Chai, X. Chen, J. Qiu, L. Du, Y. Xiao, Q. Feng, S. Ji, and Z. Tian, "MalFSCIL: a Few-Shot Class-Incremental Learning Approach for Malware Detection," *IEEE Trans. Inf. Forensics Security*, Vol. 20, Pp. 2999–3014, 2025.
- [3] CrowdStrike, Inc., "2025 Global Threat Report," CrowdStrike, Austin, TX, USA, Rep. CRWD-2025-GTR, Feb. 2025. [Online]. Available: <https://www.crowdstrike.com/wp-content/uploads/2025/02/CrowdStrikeGlobalThreatReport2025.pdf>.
- [4] Huo and F. Zhao, "Encrypted Malicious Traffic Detection Based on Stacking and Multi-feature Fusion," *Comput. Eng. (Jisuanji Gongcheng)*, Vol. 49, No. 5, Pp. 165–172, 180, 2023, Doi: 10.19678/j.issn.1000-3428.0064805.
- [5] Xu, G. Wu, W. Wang, X. Gao, a. He, and Z. Zhang, "Applying Self-supervised Learning to Network Intrusion Detection for Network Flows with Graph Neural Network," *Comput. Netw.*, Vol. 248, Art. No. 110495, 2024, Doi: 10.1016/j.comnet.2024.110495.
- [6] Chen Et Al., "a Survey on Internet Encrypted Traffic Detection, Classification and Identification," *Chin. J. Comput. (Jisuanji Xuebao)*, Vol. 46, No. 5, Pp. 1060–1085, May 2023, Doi: 10.11897/SP.J.1016.2023.01060.
- [7] Guo, W. Du, X. Yang, J. Xue, Y. Wang, W. Han, and J. Hu, "MalHAPGNN: an Enhanced Call Graph-Based Malware Detection Framework Using Hierarchical Attention Pooling Graph Neural Network," *Sensors*, Vol. 25, No. 2, Art. No. 374, 2025.
- [8] Akpaku, J. Chen, M. Ahmed, F. K. Agbenyegah, and W. L. Brown-Acquaye, "RAGN: Detecting Unknown Malicious Network Traffic Using a Robust Adaptive Graph Neural Network," *Comput. Netw.*, Vol. 262, Art. No. 111184, 2025.
- [9] AlTfaily, Z. Ghalmame, M. E. a. Brahmia, H. Hazimeh, a. Jaber, and M. Zghal, "Graph-based Federated Learning Approach for Intrusion Detection in IoT Networks," *Sci. Rep.*, Vol. 15, Art. No. 41264, 2025.

- [10] Fu, Q. Li, and K. Xu, "Detecting Unknown Encrypted Malicious Traffic in Real Time via Flow Interaction Graph Analysis," in Proc. Netw. Distrib. Syst. Security Symp. (NDSS), San Diego, CA, USA, 2023, Pp. 1–18, Doi: 10.14722/ndss.2023.23080.
- [11] Hu and L. Tang, "Method on Intrusion Detection for Industrial Internet Based on Light Gradient Boosting Machine," Chin. J. Netw. Inf. Security (Wangluo Yu Xinxi Anquan Xuebao), Vol. 9, No. 2, Pp. 46–55, 2023, Doi: 10.11959/j.issn.2096-109x.2023020.
- [12] Luo, C. Liu, G. Xiong, C. Yang, G. Gou, Y. Ren, and Z. Li, "MalRAG: a Retrieval-Augmented LLM Framework for Open-set Malicious Traffic Identification," 2025, arXiv:2511.14129. [Online]. Available: <https://arxiv.org/abs/2511.14129>
- [13] Yan, Z. Liu, Y. Xie, S. Liang, L. Liu, and K. Xu, "CertTA: Certified Robustness Made Practical for Learning-Based Traffic Analysis," in Proc. 34th USENIX Security Symp., Seattle, WA, USA, 2025, Pp. 7349–7368.
- [14] Alam, Y. Alam, S. Cui, and C. Akujuobi, "Data-Driven Network Analysis for Anomaly Traffic Detection," Sensors, Vol. 23, No. 19, Pp. 1–16, 2023.
- [15] Wang and V. L. L. Thing, "Feature Mining for Encrypted Malicious Traffic Detection with Deep Learning and Other Machine Learning Algorithms," Comput. Security, Vol. 128, Art. No. 103143, 2023.
- [16] Dash, S. Chakravarty, a. K. Rath, N. C. Giri, K. M. Aboras, and N. Gowtham, "an Optimized LSTM-based Deep Learning Model for Anomaly Network Intrusion Detection," Sci. Rep., Vol. 15, Art. No. 1554, 2025.
- [17] Kang and J. Ling, "Encrypted Traffic Classification Method Based on Densely Connected Convolutional Network," Comput. Appl. Softw. (Jisuanji Yingyong Yu Ruanjian), Vol. 41, No. 9, Pp. 363–369, 2024, Doi: 10.3969/j.issn.1000-386x.2024.09.050.
- [18] Chen, B. Qi, H. Jin, C. Wu, and L. Zhang, "Class-imbalanced Traffic Abnormal Detection Based on 1D-CNN and BiGRU," J. Comput. Appl. (Jisuanji Yingyong), Vol. 44, No. 8, Pp. 2493–2499, 2024, Doi: 10.11772/j.issn.1001-9081.2023081112.
- [19] Yang and S. Wang, "Study on Malicious Traffic Classification Algorithm Based on CNN Combined with BiGRU," Comput. Sci. (Jisuanji Kexue), Vol. 51, No. 11a, Pp. 867–875, Nov. 2024, Doi: 10.11896/j.sjcx.231100106.
- [20] Wang, Y. Zhou, H. Liu, J. Qiu, B. Fang, and Z. Tian, "ThreatInsight: Innovating Early Threat Detection through Threat-Intelligence-Driven Analysis and Attribution," IEEE Trans. Knowl. Data Eng., Vol. 36, No. 12, Pp. 9388–9402, Dec. 2024.
- [21] Ren, Y. Xiao, Y. Zhou, Z. Zhang, and Z. Tian, "CSKG4APT: a Cybersecurity Knowledge Graph for Advanced Persistent Threat Organization Attribution," IEEE Trans. Knowl. Data Eng., Vol. 35, No. 6, Pp. 5695–5709, 2023.
- [22] Zeng, D. Man, C. Xu, S. Han, H. Wang, X. Zhou, X. Li, and W. Yang, "Identification Method for Malicious Traffic in Industrial Internet under New Unknown Attack Scenarios," J. Commun., Vol. 45, No. 6, Pp. 75–83, 2024, Doi: 10.11959/j.issn.1000-436x.2024093.
- [23] Tang and B. Wang, "Study on SSL/TLS Encrypted Malicious Traffic Detection Algorithm Based on Graph Neural Networks," J. Comput. Appl., Vol. 44, No. 4, Pp. 1158–1165, 2024, Doi: 10.11772/j.issn.1001-9081.2023050566.
- [24] Liu, P. Gao, H. Zhang, K. Li, W. Yang, X. Wei, and J. Shu, "Trace2Vec: Detecting Complex Multi-step Attacks with Explainable Graph Neural Network," Pattern Recognit., Vol. 162, Art. No. 111363, 2025, Doi: 10.1016/j.patcog.2025.111363.
- [25] W. Lo, G. Kulatilleke, M. Sarhan, S. Layeghy, and M. Portmann, "XG-BoT: an Explainable Deep Graph Neural Network for Botnet Detection and Forensics," Internet Things, Vol. 22, Art. No. 100747, 2023, Doi: 10.1016/j.iot.2023.100747.
- [26] Y.-R. Tsai and Y.-H. Lee, "Edge-based graph neural networks for network intrusion detection system," in Proc. Int. Conf. Appl. Syst. Innov. (ICASI), 2025, Art. no. 2575, doi: 10.1049/icp.2025.2575.
- [27] Qian, H. Yang, R. Li, W. Chen, X. Luo, and L. Yin, "Distributed Detection of Large-scale Internet of Things Botnets Based on Graph Partitioning," Appl. Sci., Vol. 14, No. 4, Art. No. 1615, 2024, Doi: 10.3390/app14041615.
- [28] Wang, D. Yu, Y. Yang, Y. Ma, and Y. Chen, "Domain Transfer Intrusion Detection Method for Unknown Attacks on Industrial Control Systems," Comput. Sci., Vol. 51, No. 9, Pp. 1158–1165, 2024, Doi: 10.11896/j.sjcx.230800079.
- [29] K. Jonath, a. Naz, and S. Zhang, "GMM-cGAN: Mitigating Data Scarcity and Label Noise for Robust Encrypted Malicious Traffic Classification," Comput. Netw., Vol. 274, Art. No. 111823, 2026, Doi: 10.1016/j.comnet.2025.111823.
- [30] Tian, C. Luo, J. Qiu, X. Du, and M. Guizani, "a Distributed Deep Learning System for Web Attack Detection on Edge Devices," IEEE Trans. Ind. Informat., Vol. 16, No. 3, Pp. 1963–1971, 2020.
- [31] MahdaviFar, a. H. Salem, P. Victor, M. Garzon, a. H. Razavi, N. Hellberg, and a. H. Lashkari, "a Machine Learning-Based Lightweight Hybrid Data Exfiltration Detection Using DNS," in Proc. 2021 11th IEEE Int. Conf. Commun. Netw. Security (ICCNS), Weihai, China, 2021.
- [32] Chai, J. Qiu, L. Yin, L. Zhang, B. B. Gupta, and Z. Tian, "from Data and Model Levels: Improve the Performance of Few-Shot Malware Classification," IEEE Trans. Netw. Service Manag., Vol. 19, No. 4, Pp. 4248–4261, Dec. 2022, Doi: 10.1109/TNSM.2022.3200866.
- [33] Chai, L. Du, J. Qiu, L. Yin, and Z. Tian, "Dynamic Prototype Network Based on Sample Adaptation for Few-Shot Malware Detection," IEEE Trans. Knowl. Data Eng., Vol. 35, No. 5, Pp. 4754–4766, 2023.