Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.083419

ARTICLE

DVG-GNN: Dual-View Graph Representation Learning for Encrypted Traffic Classification

Guan Yang[1], Haozhen Wang[2], Yu Wang[3,*], Weiguang Liu[4] and Bo Chen[5,6]

[1]Intelligent Perception and Instrumentation College, Zhongyuan University of Technology, Zhengzhou, China
[2]School of Computer Science, Zhongyuan University of Technology, Zhengzhou, China
[3]School of Software, Henan University of Engineering, Zhengzhou, China
[4]School of Software, Zhongyuan University of Technology, Zhengzhou, China
[5]School of Mathematical Sciences, Shenzhen University, Shenzhen, China
[6]Guangdong Provincial Key Laboratory of Intelligent Information Processing, Shenzhen University, Shenzhen, China

*Corresponding Author: Yu Wang. Email: stonchor@163.com

Received: 03 April 2026; Accepted: 17 June 2026; Published: 13 August 2026

ABSTRACT: The rapid proliferation of encrypted communication technologies, such as TLS, VPNs, and Tor, has significantly limited the effectiveness of traditional traffic classification methods that rely on port numbers or deep packet inspection. While handcrafted statistical features provide partial solutions, they often lack robustness and generalization in complex traffic scenarios. Although deep learning models such as CNNs and RNNs can capture local and sequential patterns, they typically overlook higher-order structural dependencies among bytes. To address these challenges, we propose DVG-GNN, a Dual-View Graph representation learning framework for encrypted traffic classification. The framework decomposes each packet into header and payload views, leveraging distinct byte embeddings and multi-scale one-dimensional convolutions to extract fine-grained contextual features. View-specific graphs are constructed using Pointwise Mutual Information (PMI) to model byte-level relationships, and a GraphSAGE-based encoder with attention pooling is employed to learn global structural representations. A gated fusion mechanism further integrates the dual-view features to enhance discriminative capability. Extensive experiments on ISCX-VPN2016, ISCX-Tor2016, and USTC-TFC2016 demonstrate that the proposed method consistently outperforms state-of-the-art approaches across multiple evaluation metrics, validating its effectiveness and generalization ability in diverse encrypted traffic classification tasks.

KEYWORDS: Encrypted traffic classification; graph neural networks; dual-view representation learning; pointwise mutual information; traffic graph modeling

1 Introduction

With the widespread adoption of end-to-end encryption technologies, such as VPNs, Tor, and Instant Messaging (IM), network traffic visibility has significantly decreased, making traditional techniques, including port-based identification and Deep Packet Inspection (DPI), increasingly ineffective [1]. Consequently, encrypted traffic classification without payload decryption has emerged as a critical challenge in cybersecurity.

Early studies approached this problem through machine learning (ML) methods based on handcrafted statistical features extracted from traffic flows [2]. However, these methods are limited in capturing complex nonlinear patterns in encrypted traffic.

With the rapid advancement of deep learning (DL), CNN-based models have been proposed to automatically learn hierarchical representations from raw traffic data. Sequence-based models such as FS-Net further enhance temporal dependency modeling at the flow level [3]. More recently, hybrid architectures combining dual-embedding strategies and graph modeling have shown superior representation capabilities [4].

Graph Neural Networks (GNNs) have further advanced encrypted traffic classification by modeling traffic as graph-structured data and capturing non-Euclidean dependencies among packets [5]. Several graph-based frameworks have been proposed to enhance structural dependency modeling [6], particularly for encrypted malware detection tasks [7]. However, existing methods often rely on coarse-grained or heuristic graph construction strategies, which limits their ability to model fine-grained statistical dependencies between bytes or flows. For example, FlowPrint [8] mainly focuses on flow-level behavioral patterns but overlooks detailed byte-level interactions. In addition, most GNN-based methods are limited in jointly modeling heterogeneous views (e.g., header and payload), and may suffer information loss during graph construction and aggregation.

Existing GNN-based encrypted traffic classification methods mainly focus on intra-packet structural relationships while neglecting inter-packet temporal dependencies, limiting their ability to capture sequential traffic behaviors within sessions. In addition, long-session traffic is commonly handled by truncation or padding strategies, which weakens long-range temporal modeling capability. These limitations motivate the need for explicit temporal dependency modeling and more effective long-session traffic representation.

To address these issues, we propose DVG-GNN, a dual-view graph-based representation learning framework for encrypted traffic classification. DVG-GNN separates packet headers and payloads into two complementary views. PacketCNN is first employed for multi-scale feature extraction, followed by PMI-based graph construction to capture latent byte-level statistical dependencies. GraphSAGE with attention pooling is then applied for structural representation learning, while a gated fusion module adaptively integrates dual-view features.

Compared with DE-GNN and BPF-GNN, DVG-GNN introduces three major improvements: (1) PMI-based graph construction from raw byte sequences for fine-grained dependency modeling; (2) a dimension-wise gated fusion mechanism for adaptive header-payload feature integration; and (3) a lightweight BiLSTM-attention module for explicit inter-packet temporal dependency modeling. These designs enable DVG-GNN to learn more discriminative and robust traffic representations.

The main contributions are summarized as follows:

1. A dual-view encrypted traffic classification framework that jointly models header and payload semantics.
2. A unified architecture combining multi-scale convolution and PMI-based graph construction for modeling contextual and structural dependencies.
3. A GraphSAGE-based encoder with attention-based pooling and gated fusion for discriminative representation learning.
4. A lightweight BiLSTM-attention temporal enhancement module that explicitly models inter-packet temporal dependencies and enhances long-session traffic representation learning through truncation-padding and temporal encoding strategies.
5. Extensive experiments that demonstrate superior performance and robustness compared with state-of-the-art methods.

2 Task Description and Related Work

2.1 Task Description

Encrypted traffic classification aims to categorize network traffic into distinct classes (e.g., streaming, instant messaging) based on observable features, such as byte distributions and temporal sequences, without requiring decryption. This task is framed as a pattern recognition problem that takes advantage of the hierarchical structure of network traffic, which consists of bytes, packets, and flows. The fundamental unit of analysis is the network flow, defined by the five-tuple. Given a training set with N samples across C classes, a flow S is represented as a sequence of n packets:

$$S = \{P_1, P_2, \ldots, P_n\} \tag{1}$$

Each packet P_i is a sequence of L bytes:

$$P_i = \{b_{i1}, b_{i2}, \ldots, b_{iL}\} \tag{2}$$

The classification function is defined as:

$$f : S \to Y \tag{3}$$

where $Y \in \{1, 2, \ldots, C\}$ is the class label.

2.2 Related Work

Encrypted network traffic classification has been widely investigated through machine learning (ML) and deep learning (DL) techniques. Early ML-based methods primarily relied on handcrafted statistical and flow-level features, such as packet size distributions and inter-arrival times. However, these approaches suffer from limited representational capacity when handling highly obfuscated encrypted traffic. With the rapid advancement of deep learning, researchers have increasingly focused on learning discriminative representations directly from raw or minimally processed traffic data.

Recent works have explored increasingly sophisticated architectures for performance improvement. BSTFNet integrates global semantic and spatiotemporal features to effectively capture encrypted malicious traffic patterns [9]. Multi-level pre-training strategies have also been introduced to enhance hierarchical representation learning and generalization performance [10]. In addition, graph-based methods have garnered increasing attention. CGNN constructs traffic graphs to capture structural relationships among traffic entities [11]. Beyond standard graph learning, alternative approaches have been further explored. SCNNTraffic applies spiking convolutional neural networks for energy-efficient encrypted traffic classification [12]. Geometric learning-based methods model traffic data from a manifold perspective to improve feature robustness [13].

Despite these advances, existing methods still have limitations. Many approaches focus on either temporal modeling or coarse-grained structural representations, which limits their ability to model complex dependencies in encrypted traffic. Moreover, most methods do not explicitly distinguish between packet-level components such as headers and payloads, thereby restricting their ability to learn fine-grained semantic representations. These limitations motivate the need for more expressive and fine-grained modeling frameworks.

3 Methodology

This chapter outlines the DVG-GNN methodology, including dual embedding, multi-scale feature extraction, byte-level traffic graph construction, and fusion of header and payload features. A classification module and joint optimization are also introduced to enhance performance and robustness. The overall framework is illustrated in Fig. 1.

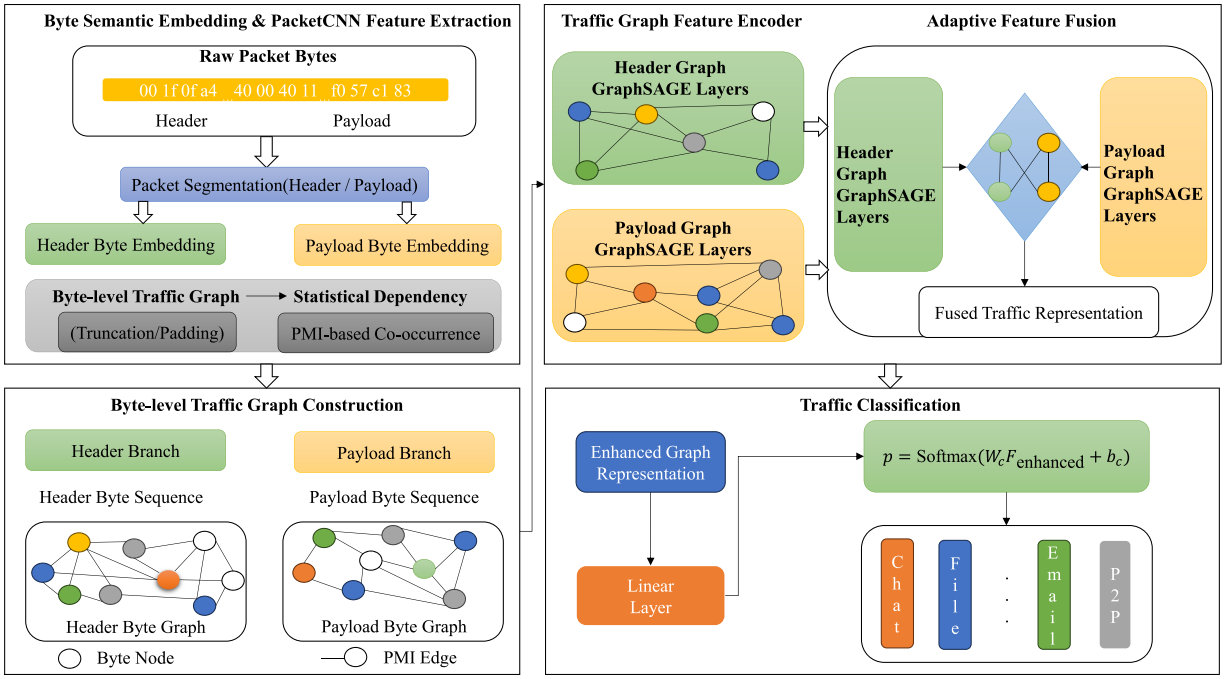

Figure 1: The overall framework of the proposed DVG-GNN model.

3.1 Dual Embedding and Multi-Scale Feature Extraction

3.1.1 Dual Embedding

Headers and payloads exhibit different statistical characteristics. Modeling them jointly may weaken discriminative capability. To address this issue, we employ a dual-embedding mechanism that learns separate semantic spaces for the two views. Denote the header and payload sequences as B_H and B_P, with lengths $N_H = 40$ and $N_P = 150$, respectively. These sequences are mapped into a shared d-dimensional embedding space through a learnable function:

$$\mathbf{E}_H, \mathbf{E}_P = \text{Embed}(B_H, B_P), \quad \mathbf{E}_H \in \mathbb{R}^{N_H \times d}, \quad \mathbf{E}_P \in \mathbb{R}^{N_P \times d} \tag{4}$$

The embedding dimension $d = 64$ is designed to capture local patterns and global dependencies. This dual embedding mechanism extracts meaningful features from both header and payload sequences while mitigating mutual interference between heterogeneous features.

Headers mainly contain protocol-structural information, such as packet length and flag patterns, whereas payloads encode richer semantic and contextual byte distributions. Due to their heterogeneous characteristics, directly embedding them into a unified space may introduce representation bias and suppress fine-grained header patterns. Therefore, the proposed dual-view embedding mechanism learns view-specific representations independently, reducing mutual interference while preserving complementary structural and semantic information for subsequent graph representation learning.

3.1.2 Multi-Scale Convolutional Feature Extraction (PacketCNN)

We propose PacketCNN, a dual-branch multi-scale 1D convolutional network for packet-level feature extraction from encrypted traffic. Raw packets are first transformed into one-hot encoded representations and then divided into header and payload views for independent feature learning.

The motivation for adopting multi-scale convolutions is that encrypted traffic exhibits dependencies at different receptive-field scales. Small kernels capture local byte dependencies and short-range protocol signatures, whereas larger kernels model broader contextual correlations. Moreover, different convolution settings are assigned to headers and payloads according to their characteristics: headers are short and highly structured, while payloads contain richer semantic information and longer contextual dependencies. This asymmetric multi-scale design enables PacketCNN to better capture heterogeneous traffic features and improve representation diversity.

Specifically, the Header Branch employs 1D convolutions with kernel sizes $K_h = \{1, 2, 3, 4\}$ (16 channels each), followed by global max pooling and a fully connected layer. The Payload Branch employs 1D convolutions with kernel sizes $K_p = \{2, 3, 4\}$ (64 channels each), followed by global max pooling. The overall architecture of PacketCNN is illustrated in Fig. 2.

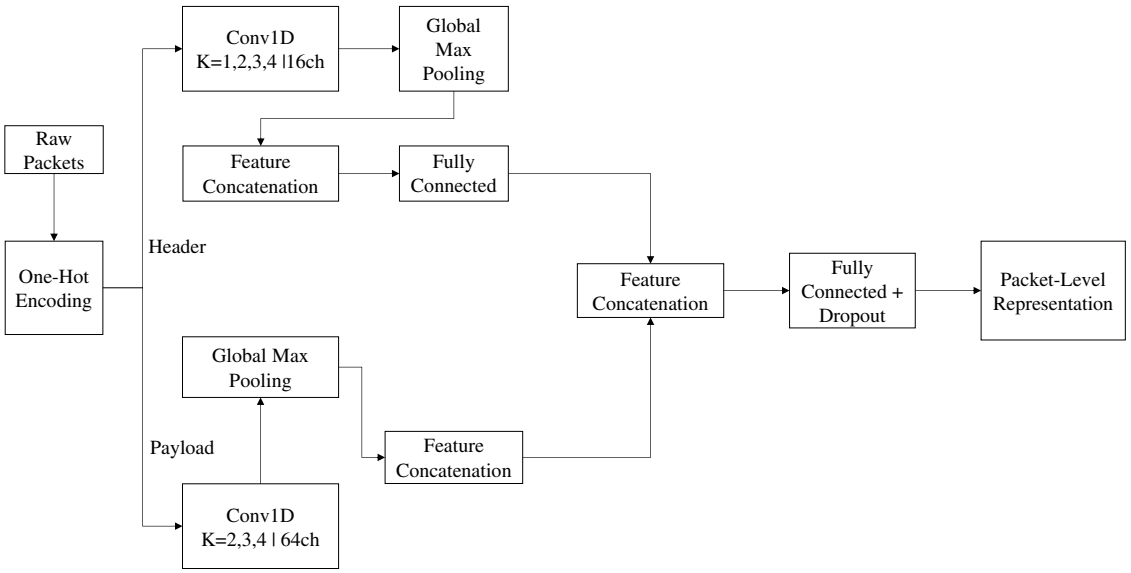

Figure 2: The architecture of PacketCNN: a dual-branch multi-scale 1D convolutional network for packet-level feature extraction.

Mathematically, the feature extraction process of the two branches is formulated as:

$$X_h = \text{GlobalPool}(\sigma(\text{Conv}(E_h))), \quad X_p = \text{GlobalPool}(\sigma(\text{Conv}(E_p))) \tag{5}$$

where σ denotes the Parametric ReLU (PReLU) activation function:

$$f(x) = \max(0, x) + \alpha \min(0, x) \tag{6}$$

The extracted header and payload features are subsequently concatenated and projected into a unified latent space:

$$P = \text{Linear}(\text{Concat}(X_h, X_p)) \tag{7}$$

The concatenation operation integrates heterogeneous features from different views and receptive fields, while the subsequent linear projection enables adaptive feature fusion in a shared latent space. This process preserves complementary structural and semantic information while reducing feature redundancy, thereby producing a compact and discriminative packet-level representation for downstream graph construction and traffic classification.

Finally, dropout regularization is applied to the final layer to improve model generalization capability.

3.2 Byte-Level Traffic Graph Construction

Each encrypted flow is represented as a weighted undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$, where $\mathcal{V}$ denotes byte-type nodes, $\mathcal{E}$ represents statistical dependencies among bytes, and $\mathbf{X}$ is the node feature matrix. To reduce computational complexity, identical byte values are aggregated into a single node, ensuring $|\mathcal{V}| \leq 256$. Each flow is truncated or padded to a fixed length to ensure input consistency.

The motivation for constructing byte-level graphs is that encrypted traffic exhibits complex non-Euclidean dependency structures that are difficult to capture using sequential representations alone. By modeling traffic as graphs, the proposed method can explicitly learn high-order relationships among bytes and local contextual interactions. Merging identical byte values into shared nodes yields a compact graph representation that reduces graph size and computational overhead while preserving essential byte co-occurrence semantics and structural dependencies.

Edges are constructed using Pointwise Mutual Information (PMI), which measures statistical dependencies between byte pairs within a sliding window of size w (stride = 1). Co-occurrence statistics are used to estimate empirical probabilities $p(i)$ and $p(i, j)$, resulting in:

$$\mathrm{PMI}(i, j) = \log\left(\frac{p(i, j)}{p(i)\, p(j)}\right) \tag{8}$$

To improve robustness against noisy co-occurrences and distributional fluctuations, a lightweight PMI sparsification strategy is adopted. Since weak byte co-occurrences are more likely to originate from random packet variations or transient traffic noise, only statistically significant dependencies are preserved.

Specifically, an adaptive threshold is defined as:

$$\tau = \mu_{\mathrm{PMI}} + \lambda \cdot \sigma_{\mathrm{PMI}} \tag{9}$$

where μ_{PMI} and σ_{PMI} denote the mean and standard deviation of PMI values estimated from the training set statistics. The coefficient $\lambda = 1.0$ is empirically selected and fixed across all datasets.

Only edges satisfying $\mathrm{PMI}(i, j) > \tau$ are retained, which effectively suppresses unstable low-strength correlations while preserving informative byte dependencies. This sparsification mechanism improves graph stability under dynamic traffic distributions and reduces sensitivity to random co-occurrence noise.

Node features $\mathbf{X}$ integrate contextual and statistical information, where PacketCNN-derived embeddings are concatenated with neighborhood-invariant statistical descriptors. All continuous features are normalized to $[0, 1]$ to improve numerical stability.

This hybrid feature design enhances node representation diversity by integrating semantic context and structural information. PacketCNN captures contextual representations from raw packets, while statistical descriptors preserve intrinsic graph properties, thereby enabling the simultaneous modeling of sequential semantics and topological structure.

The retained PMI values are directly used as edge weights without additional normalization in order to preserve relative dependency strengths among byte pairs. The construction process is linear in sequence length due to the sliding window mechanism, and graph sparsity is naturally enforced by the adaptive PMI filtering strategy.

As illustrated in Fig. 3 with key hyperparameters summarized in Table 1, the proposed approach leverages adaptive thresholding to preserve strong byte dependencies while reducing sensitivity to noise and traffic variability, thereby generating more stable graph structures for downstream learning.

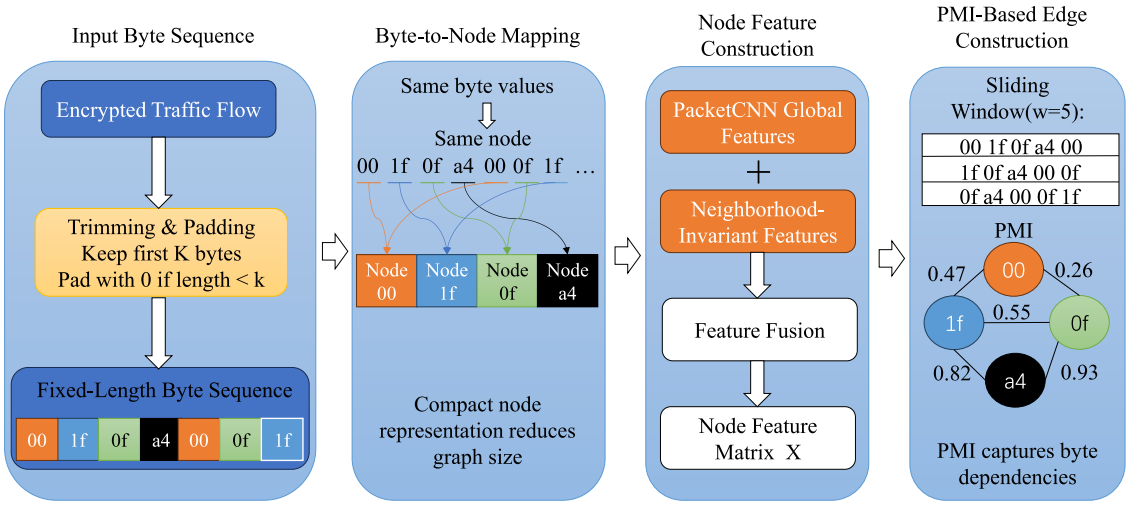

Figure 3: Byte-level traffic graph construction process.

Table 1: Key hyperparameters for PMI-based graph construction (robustness-enhanced).

Parameter	Description	Value
FLOW_PAD_TRUNC_LENGTH	Max packets per flow for fixed-length input	50
PMI_WINDOW_SIZE	Sliding window for byte co-occurrence	5
ϵ	Smoothing constant to avoid zero probability	10^{-8}
Adaptive threshold λ	Coefficient for statistical significant edge filtering	1.0

3.3 Traffic Graph Feature Encoder

The Traffic Graph Encoder learns discriminative graph-level representations from node features $\mathbf{X}$ through multi-subspace fusion, GraphSAGE aggregation, and attention-based pooling.

3.3.1 Multi-Subspace Feature Fusion

Node features are decomposed into K subspaces and mapped into a unified latent space:

$$\mathbf{H}^{(0)} = \sum_{k=1}^{K} w_k \cdot \sigma(\mathbf{W}_k \mathbf{X}_k + \mathbf{b}_k) \tag{10}$$

where $\sigma(\cdot)$ denotes PReLU.

Multi-subspace decomposition is designed to address the heterogeneous characteristics of encrypted traffic features. Directly projecting heterogeneous features into a single space may suppress complementary information. By learning subspace-specific transformations and adaptively fusing them, the encoder enhances feature representation diversity while reducing interference among heterogeneous components.

3.3.2 GraphSAGE-Based Aggregation

GraphSAGE [14] conducts neighborhood aggregation with residual connections:

$$\mathbf{H}^{(l)} = \sigma\left(\mathbf{W}^{M} \cdot \text{Concat}\left(\mathbf{H}^{(l-1)}, \text{Mean}\left(\mathbf{H}_j^{(l-1)} \mid j \in \mathcal{N}(i)\right)\right)\right) + \mathbf{H}^{(l-1)} \tag{11}$$

GraphSAGE is adopted due to its inductive learning capability and suitability for sparse traffic graphs. Mean aggregation provides stable neighborhood encoding while mitigating sensitivity to noisy neighbors. Residual connections further alleviate over-smoothing and help preserve discriminative node representations across layers.

3.3.3 Attention-Based Graph Pooling

Graph-level representation is generated through attention pooling:

$$\mathbf{h}_{\mathcal{G}} = \sum_{i \in \mathcal{V}} \alpha_i \mathbf{H}_i^{(L)}, \quad \alpha_i = \text{Softmax}\left(\text{MLP}\left(\mathbf{H}_i^{(L)}\right)\right) \tag{12}$$

Attention-based pooling addresses the limitation of uniform aggregation by assigning higher weights to more informative nodes. The MLP-based scoring function learns nonlinear importance measures, enabling the model to emphasize discriminative byte interaction patterns while suppressing less informative nodes.

This pooling is performed at the packet level, where each packet is represented as an independent graph. Thus, $\mathbf{h}_{\mathcal{G}}$ represents a packet-level embedding rather than a flow-level representation.

Packet-level graph modeling captures the hierarchical nature of encrypted traffic. Each packet encodes local structural semantics, while higher-level flow dynamics are formed through packet sequences. This design retains fine-grained structural information for subsequent temporal dependency modeling.

Each traffic flow is therefore represented as an ordered sequence of packet graphs for subsequent temporal dependency modeling.

3.4 Feature Fusion and Representation Enhancement

To integrate complementary information from dual-view graphs, we propose a gated fusion and representation enhancement framework. This module adaptively combines header and payload embeddings $\mathbf{h}_h$ and $\mathbf{h}_p$, which capture protocol-structural patterns and fine-grained behavioral cues, respectively.

The fusion design is motivated by the distinct contributions of header and payload views across traffic categories. Treating both views equally may lead to feature redundancy. Therefore, the proposed mechanism conducts adaptive cross-view feature integration with temporal enhancement to preserve complementary information and model inter-packet dependencies.

3.4.1 Gated Feature Fusion

Unlike static fusion strategies such as concatenation or averaging, we introduce a dimension-wise gating mechanism that dynamically controls the contribution of each view:

$$\boldsymbol{\alpha} = \sigma\left(\mathbf{W}_g\left[\mathbf{h}_h \| \mathbf{h}_p\right] + \mathbf{b}_g\right), \quad \hat{\mathbf{h}} = \boldsymbol{\alpha} \odot \mathbf{h}_h + (\mathbf{1} - \boldsymbol{\alpha}) \odot \mathbf{h}_p \tag{13}$$

Here, $[\cdot\|\cdot]$ represents concatenation and $\odot$ represents element-wise multiplication. The sigmoid function generates a learnable gating vector that adaptively selects informative dimensions from header and payload features.

The gating mechanism implements soft feature selection by assigning adaptive importance weights to different dimensions. Compared with static fusion methods, it reduces redundancy, preserves complementary information, and suppresses less informative components, thereby enhancing robustness and adaptability.

This design enables the fused representation $\hat{\mathbf{h}}$ to adapt to diverse structural and semantic patterns by suppressing redundant components and preserving complementary cues. The resulting representation is more robust and discriminative in downstream classification tasks. The schematic diagram of gated feature fusion is shown in Fig. 4. The gated fusion operation further enhances cross-view feature interaction modeling by introducing nonlinear dependencies between header and payload embeddings. Since gating coefficients are jointly learned from both views, the model can capture cross-view correlations that static fusion strategies fail to capture effectively.

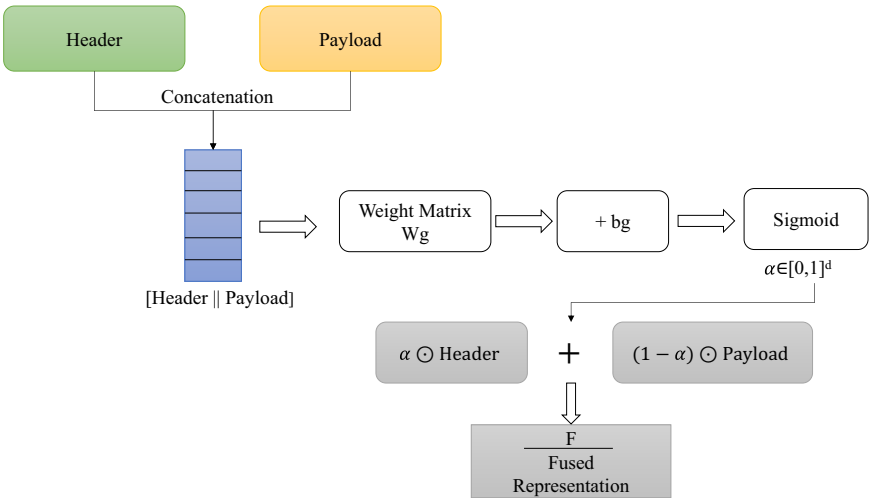

Figure 4: The architecture of gated fusion mechanism for dual-view feature integration.

3.4.2 Representation Enhancement

The fused representation $\hat{\mathbf{h}}$ is further enhanced to capture higher-order interactions and temporal dependencies.

Although gated fusion generates discriminative packet-level features, encrypted traffic flows still contain temporal dependencies that cannot be fully captured by independent packet modeling. Therefore, a temporal enhancement module is introduced to explicitly model inter-packet contextual dependencies.

Temporal Modeling Enhancement:

Each flow is represented as an ordered sequence of fused packet embeddings:

$$\mathbf{H} = \left[\hat{\mathbf{h}}_1, \hat{\mathbf{h}}_2, \ldots, \hat{\mathbf{h}}_T\right] \tag{14}$$

where $\hat{\mathbf{h}}_t \in \mathbb{R}^d$ denotes the representation of the t-th packet. Flows shorter than T are zero-padded, while longer flows are truncated. In experiments, $T = 50$ balances temporal coverage and computational efficiency.

For long-session traffic, a fixed-length sliding packet window is adopted to preserve representative temporal interaction patterns and reduce information loss caused by direct truncation.

The sequence is then modeled using a bidirectional LSTM:

$$\mathbf{H}_{\text{temp}} = \text{BiLSTM}(\mathbf{H}) \tag{15}$$

where $\mathbf{H}_{\text{temp}} \in \mathbb{R}^{T \times d'}$ denotes the temporal hidden states.

BiLSTM is employed to capture bidirectional temporal dependencies, where forward and backward hidden states jointly model long-range traffic correlations.

Lightweight Attention Enhancement

A lightweight attention mechanism is further applied:

$$\mathbf{h}_{\text{enh}} = \text{Attention}(\mathbf{H}_{\text{temp}}) \tag{16}$$

Attention weights are normalized by Softmax to adaptively aggregate informative temporal features.

The attention mechanism emphasizes informative packet subsequences while suppressing less useful temporal responses, which is particularly beneficial for long-session traffic.

The final representation $\mathbf{h}_{\text{enh}}$ integrates both semantic and temporal information for downstream traffic classification.

3.5 Classification and Joint Optimization

Based on $\mathbf{h}_{\text{enh}}$, the model predicts traffic categories using a lightweight classification head and a joint optimization strategy to address class imbalance.

3.5.1 Classification Layer

The enhanced feature is projected into the class space through a linear transformation:

$$\mathbf{z} = \mathbf{W}_c \cdot \mathbf{h}_{\text{enh}} + \mathbf{b}_c \tag{17}$$

where $\mathbf{W}_c \in \mathbb{R}^{C \times d}$ and $\mathbf{b}_c \in \mathbb{R}^C$ are learnable parameters, and C denotes the number of classes. The predicted probabilities are obtained using Softmax:

$$\hat{\mathbf{y}} = \text{Softmax}(\mathbf{z}) \tag{18}$$

3.5.2 Loss Functions

To alleviate class imbalance, a weighted focal loss is adopted:

$$\mathcal{L}_{\text{foc}} = -\sum_{c=1}^{C} w_c \cdot y_c \cdot (1 - \hat{y}_c)^{\gamma} \log(\hat{y}_c) \tag{19}$$

where w_c denotes the class weight and γ controls the contribution of hard samples.

To enhance feature separability, an inter-class separation loss is further introduced:

$$\mathcal{L}_{\text{sep}} = \sum_{c_i \neq c_j} \max\left(0, m - \|\boldsymbol{\mu}_{c_i} - \boldsymbol{\mu}_{c_j}\|_2^2\right) \tag{20}$$

where $\boldsymbol{\mu}_c$ is the batch-wise feature center of class c:

$$\boldsymbol{\mu}_c = \frac{1}{N_c} \sum_{i=1}^{N_c} \mathbf{h}_i^{(c)} \tag{21}$$

with N_c denoting the number of samples in the current mini-batch. Classes with $N_c = 0$ are excluded from computation.

The final objective is defined as:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{foc}} + \lambda \cdot \mathcal{L}_{\text{sep}} \tag{22}$$

The weighting factor λ follows a three-stage curriculum schedule: $\lambda = 0$ during the first 30% of training, linearly increases to $\lambda_{\max}$ in the next 40%, and remains fixed in the final 30%. In experiments, $\lambda_{\max} = 0.3$. This strategy prevents early over-constraint while improving inter-class discriminability and robustness to class imbalance.

4 Experiments and Analysis

4.1 Experimental Configuration

4.1.1 Datasets

We evaluate DVG-GNN on multiple widely adopted public datasets to ensure a comprehensive and fair performance evaluation. Experiments are conducted on two benchmark datasets from the Canadian Institute for Cybersecurity (CIC) [15,16]: ISCX VPN-nonVPN and ISCX Tor-nonTor. The ISCX VPN-nonVPN dataset consists of both conventional and VPN-encrypted traffic across six activity classes, while the ISCX Tor-nonTor dataset comprises Tor-anonymized and non-Tor traffic with eight classes. For ISCX-Tor, each flow is segmented into 60-s non-overlapping time windows [17] to mitigate data sparsity issues. These datasets encompass diverse encrypted communication scenarios, including VPN tunneling and Tor anonymization, and their detailed traffic class labels are summarized in Table 2.

Table 2: Traffic class labels for datasets.

Dataset	Traffic Type	Content
ISCX-VPN	VPN-Email	Email, Gmail (SMTP, POP3, IMAP)
	VPN-Chat	ICQ, AIM, Skype, Facebook, Hangouts
	VPN-Streaming	Vimeo, Youtube, Netflix, Spotify
	VPN-File transfer	Skype, FTPS, SFTP
	VPN-VoIP	Facebook, Skype, Hangouts, Voipbuster
	VPN-P2P	uTorrent, Bittorrent
ISCX-nonVPN	Email	Email, Gmail (SMTP, POP3, IMAP)
	Chat	ICQ, AIM, Skype, Facebook, Hangouts
	Streaming	Vimeo, Youtube, Netflix, Spotify
	File transfer	Skype, FTPS, SFTP

(Continued)

Table 2 (continued)

Dataset	Traffic Type	Content
	VoIP	Facebook, Skype, Hangouts, Voipbuster
	P2P	uTorrent, Bittorrent
ISCX-Tor	Tor-Browsing	Firefox, Chrome
	Tor-mail	Thunderbird (SMTP/S, POP3/SSL, IMAP/SSL)
	Tor-Chat	Facebook, Hangouts, Skype, AIM, ICQ
	Tor-Audio	Spotify
	Tor-Video	YouTube, Vimeo
	Tor-File	Skype, SFTP, FTPS
	Tor-VoIP	Facebook, Hangouts, Skype
	Tor-P2P	Bittorrent
ISCX-nonTor	Browsing	Firefox, Chrome
	Email	Thunderbird (SMTP/S, POP3/SSL, IMAP/SSL)
	Chat	Facebook, Hangouts, Skype, AIM, ICQ
	Audio	Spotify
	Video	YouTube, Vimeo
	FTP	Skype, SFTP, FTPS
	VoIP	Facebook, Hangouts, Skype
	P2P	Bittorrent

In addition, we additionally evaluate the proposed method on the USTC-TFC2016 dataset [18], a widely adopted benchmark for encrypted and malicious traffic classification. This dataset contains real-world traffic from both benign applications and multiple malware families (e.g., ransomware, botnets, and trojans), providing packet-level and flow-level data with rich behavioral diversity. Compared with the ISCX datasets, USTC-TFC2016 exhibits more heterogeneous and complex traffic patterns, particularly in malicious scenarios, making it more suitable for assessing model robustness and generalization under challenging conditions.

4.1.2 Data Preprocessing

To prepare raw PCAP data for model input, a unified preprocessing pipeline is adopted. First, SplitCap is used to divide traffic into bidirectional sessions based on the five-tuple, where sessions containing more than 10,000 packets or missing valid payloads are removed, and irrelevant fields such as Ethernet headers, IP addresses, and ports are discarded. Next, only the first N packets are retained, while packet headers and payloads are truncated or padded to 40 bytes and 150 bytes, respectively, and shorter packet sequences are padded using 0xFF. For data augmentation, Tor subset flows are segmented into non-overlapping 60-s subflows to improve training stability, where segmentation is performed after dataset partitioning to avoid data leakage. Finally, stratified sampling is applied at the session (flow) level to divide the dataset into training and testing sets with an 8:2 ratio, ensuring consistent class distributions and reliable evaluation results.

4.1.3 Implementation Details

In the experiments, the AdamW optimizer was used with an initial learning rate of 1×10^{-3} and a weight decay coefficient of 5×10^{-4}. A learning rate scheduler was applied to gradually reduce the learning rate to 1×10^{-5}. The batch size was set to 64, the warm-up ratio was 0.1, and the dropout rate was 0.2. All models were implemented using PyTorch 2.3.0+cu121 and trained on a single GPU (NVIDIA RTX 4060 Laptop GPU). Each experiment was repeated 10 times with different random seeds, and the average results were reported to ensure the reliability and reproducibility of the evaluation.

4.1.4 Evaluation Metrics

To evaluate the model's performance in encrypted traffic classification, four metrics were employed: Accuracy, Precision, Recall, and F1-score. Let TP, FP, FN, and TN denote the numbers of true positives, false positives, false negatives, and true negatives, respectively. The metrics were defined as follows:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{23}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{24}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{25}$$

$$\text{F1} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{26}$$

For multi-class classification tasks, these metrics were computed for each class individually, and macro-averaging was employed for overall evaluation to mitigate the impact of class imbalance.

4.2 Comparative Experiments

To comprehensively assess the effectiveness of the proposed DVG-GNN, we compare it with representative baseline methods spanning multiple modeling paradigms.

Sequence-based baselines include 1D-CNN [19], which effectively captures local byte-level patterns, and ET-BERT [20], which leverages pretrained representations for contextual feature learning. Transformer-based models [21,22] are additionally included in the comparison owing to their capability to model long-range dependencies via self-attention mechanisms.

Graph-based methods also include EC-GCN [23], which learns structural features through multi-scale graph convolutions; FB-GNN [24], which models flow-level interaction structures; DE-GNN [25], which enhances generalization through dual-embedding learning; GraphDApp [26], which captures behavioral relationships among traffic flows; BPF-GNN [27], which introduces a multi-granularity feature extraction mechanism for fine-grained encrypted traffic modeling; TMC-GCN [28], which explicitly models temporal dependencies and communication interactions through flow mapping graphs.

All methods are evaluated under consistent data preprocessing, dataset splits, and evaluation metrics to guarantee a fair and unbiased comparison.

4.2.1 Results on ISCX Datasets

The experimental results on ISCX-VPN, ISCX-nonVPN, ISCX-Tor, and ISCX-nonTor datasets are reported in Tables 3 and 4. Overall, the proposed DVG-GNN consistently achieves the best performance across all scenarios and metrics.

Table 3: Performance comparison on ISCX-VPN and ISCX-NonVPN datasets.

Model	ISCX-VPN				ISCX-NonVPN			
	AC	PR	RC	F1	AC	PR	RC	F1
1D-CNN	0.8426	0.8511	0.8504	0.8490	0.7824	0.7990	0.8101	0.8034
ET-BERT	0.9521	0.9452	0.9450	0.9437	0.9229	0.9127	0.9153	0.9131
Transformer	0.9193	0.9310	0.9287	0.9288	0.9015	0.8813	0.8917	0.8852
EC-GCN	0.8125	0.8346	0.8621	0.8469	0.7712	0.7857	0.7560	0.7685
FB-GNN	0.7409	0.7211	0.7132	0.7158	0.6613	0.6787	0.6726	0.6740
DE-GNN	0.9472	0.9540	0.9367	0.9437	0.9182	0.9122	0.9019	0.9058
GraphDApp	0.6967	0.5531	0.5720	0.5642	0.4421	0.3856	0.3517	0.3664
BPF-GNN	0.9520	0.9455	0.9491	0.9468	0.9148	0.9223	0.9136	0.9167
TMC-GCN	0.9314	0.9357	0.9203	0.9270	0.9022	0.9034	0.9019	0.9021
Ours	0.9884	0.9875	0.9867	0.9855	0.9243	0.9358	0.9367	0.9349

Table 4: Performance comparison on ISCX-Tor and ISCX-NonTor datasets.

Model	ISCX-Tor				ISCX-NonTor			
	AC	PR	RC	F1	AC	PR	RC	F1
1D-CNN	0.7034	0.5525	0.5237	0.5368	0.8051	0.8219	0.7837	0.8016
ET-BERT	0.9621	0.9330	0.9515	0.9411	0.9124	0.8430	0.8374	0.8392
Transformer	0.9392	0.9489	0.9577	0.9534	0.9082	0.9007	0.8914	0.8944
EC-GCN	0.6926	0.6267	0.5623	0.5910	0.8782	0.7865	0.7325	0.7567
FB-GNN	0.6599	0.4483	0.4301	0.4377	0.8169	0.5790	0.5473	0.5591
DE-GNN	0.9722	0.9829	0.9820	0.9811	0.9320	0.8914	0.8723	0.8799
GraphDApp	0.6848	0.5516	0.4879	0.5178	0.7671	0.6428	0.6254	0.6301
BPF-GNN	0.9559	0.9620	0.9686	0.9651	0.9217	0.8826	0.8540	0.8671
TMC-GCN	0.9635	0.9714	0.9725	0.9708	0.9297	0.8883	0.8697	0.8780
Ours	0.9896	0.9901	0.9925	0.9895	0.9718	0.9247	0.8813	0.9003

On the VPN/NonVPN classification task, DVG-GNN achieves an Accuracy of 0.9884 and an F1-score of 0.9855 on ISCX-VPN, and 0.9243 and 0.9349 on ISCX-nonVPN, respectively, achieving the best overall performance among the compared methods. Compared with sequence-based models such as ET-BERT and Transformer-based architectures, it demonstrates notable performance improvements, suggesting that purely sequential modeling may be insufficient for capturing complex encrypted traffic patterns.

Furthermore, DVG-GNN outperforms strong graph-based baselines, including DE-GNN, BPF-GNN, and TMC-GCN, indicating the effectiveness of the proposed dual-view graph representation in learning more discriminative traffic structures.

Similar trends are observed on the ISCX-Tor and ISCX-nonTor datasets. DVG-GNN achieves the highest Accuracy (0.9896) and F1-score (0.9895) on ISCX-Tor, and 0.9718 and 0.9003 on ISCX-nonTor, respectively. These results further demonstrate the robustness and generalization capability of the proposed model across different encrypted traffic classification scenarios.

4.2.2 Results on USTC-TFC2016 Dataset

The results on the USTC-TFC2016 dataset, summarized in Table 5, further validate the superiority of the proposed method. DVG-GNN achieves the best overall performance, with an Accuracy of 0.9969, Precision of 0.9957, Recall of 0.9963, and F1-score of 0.9950.

Table 5: Performance comparison on USTC-TFC2016 dataset.

Model	USTC-TFC2016			
	AC	**PR**	**RC**	**F1**
1D-CNN	0.9358	0.9394	0.9285	0.9340
ET-BERT	0.9466	0.9580	0.9629	0.9593
Transformer	0.9470	0.9553	0.9605	0.9569
EC-GCN	0.8264	0.8471	0.8368	0.8416
FB-GNN	0.9328	0.9437	0.9550	0.9484
DE-GNN	0.9831	0.9728	0.9757	0.9735
GraphDApp	0.9140	0.9253	0.9181	0.9196
BPF-GNN	0.9747	0.9636	0.9550	0.9586
TMC-GCN	0.9752	0.9640	0.9433	0.9546
Ours	0.9969	0.9957	0.9963	0.9950

Compared with sequence-based methods such as 1D-CNN, ET-BERT, and Transformer-based models, DVG-GNN consistently achieves superior performance across multiple evaluation metrics, demonstrating the advantage of incorporating structural information beyond purely sequential modeling. In particular, the improvements over ET-BERT and Transformer-based architectures in Accuracy and F1-score suggest that sequence-only approaches may be insufficient for capturing complex encrypted traffic patterns.

Compared with graph-based baselines, including EC-GCN, FB-GNN, GraphDApp, DE-GNN, BPF-GNN, and TMC-GCN, DVG-GNN further achieves superior classification performance. These results validate the effectiveness of the proposed dual-view graph modeling and adaptive fusion mechanism in jointly capturing structural dependencies and contextual semantics.

Overall, experimental results on the ISCX and USTC-TFC2016 datasets demonstrate the robustness and strong generalization capability of DVG-GNN for encrypted traffic classification.

4.3 Ablation Study

The ablation study evaluates the contribution of each component in DVG-GNN. Removing the payload branch (-PL) or header branch (-HD) analyzes the effect of dual-view information. Excluding the dual embedding module (-DE) evaluates semantic space separation, while removing PacketCNN (-PC), replacing PMI-based edges with temporal adjacency (-TE), and replacing gated fusion with concatenation (-CF) assess packet-level feature extraction, statistical dependency modeling, and adaptive feature fusion, respectively.

In addition, a temporal modeling ablation without the BiLSTM-attention module (-TP) is conducted to evaluate explicit inter-packet dependency learning. Several combined ablation settings are further introduced to analyze interactions among convolutional extraction, graph construction, and dual-view fusion.

The results in Table 6 show that the full model achieves the best performance on both ISCX-VPN and ISCX-Tor. Removing either the payload or header branch leads to noticeable degradation, confirming the

complementarity of dual-view representations. Performance further decreases under (-DE), indicating that separate semantic embedding alleviates feature interference.

Table 6: Ablation study results on ISCX-VPN and ISCX-Tor datasets.

Model	ISCX-VPN				ISCX-Tor			
	AC	PR	RC	F1	AC	PR	RC	F1
w/o PL	0.9152	0.9263	0.9187	0.9219	0.9744	0.9640	0.9745	0.9683
w/o HD	0.8324	0.8211	0.8263	0.8224	0.9465	0.9380	0.9462	0.9415
w/o DE	0.9346	0.9477	0.9275	0.9367	0.9718	0.9594	0.9747	0.9659
w/o PC	0.8327	0.8388	0.8429	0.8395	0.8892	0.9037	0.9163	0.9084
w/o TE	0.8873	0.8794	0.9032	0.8908	0.9473	0.9361	0.9384	0.9356
w/o CF	0.9424	0.9577	0.9526	0.9537	0.9750	0.9820	0.9837	0.9795
w/o TP	0.9297	0.9125	0.9345	0.9226	0.9477	0.9513	0.9630	0.9551
w/o TE + PC	0.7815	0.7946	0.8012	0.7968	0.8614	0.8745	0.8823	0.8771
w/o TP + CF	0.8463	0.8528	0.8617	0.8561	0.9186	0.9247	0.9314	0.9270
w/o PC + CF	0.8048	0.8121	0.8013	0.8040	0.8725	0.8939	0.9018	0.8962
ALL	0.9884	0.9875	0.9867	0.9855	0.9896	0.9901	0.9925	0.9895

Among all variants, removing PacketCNN (-PC) causes the largest performance drop, highlighting the importance of packet-level contextual extraction. Replacing PMI-based edges with temporal adjacency (-TE) also reduces performance, verifying the effectiveness of statistical dependency modeling, while the degradation under (-CF) demonstrates the benefit of adaptive gated fusion.

Removing the temporal enhancement module (-TP) consistently reduces performance, confirming the importance of explicit temporal dependency modeling. Furthermore, combined ablation experiments lead to greater performance degradation than individual variants, demonstrating that PacketCNN, PMI-based graph construction, and dual-view fusion work in concert rather than in isolation. This mutual reinforcement ensures that the full model learns a more robust representation, where the removal of any single component undermines the contributions of the others.

Overall, the results demonstrate that all components positively contribute to the performance of DVG-GNN.

4.4 Confusion Matrix Analysis

To further evaluate class-level performance, the normalized confusion matrix on the ISCX-Tor dataset is shown in Fig. 5.

The confusion matrix exhibits strong diagonal dominance, indicating that most samples are correctly classified and demonstrating the effectiveness of the proposed model. Misclassifications are sparse and mainly occur between classes with similar traffic patterns.

Specifically, limited confusion appears in the Tor-File class, where several samples are misclassified as Tor-mail due to similar low-volume bursty transmissions. Minor errors are also observed between Tor-VoIP, Tor-P2P, and email-related traffic because of similar short signaling or handshake behaviors. In contrast, Tor-Audio and Tor-Video are clearly distinguished, demonstrating the capability of DVG-GNN to capture discriminative structural and contextual features.

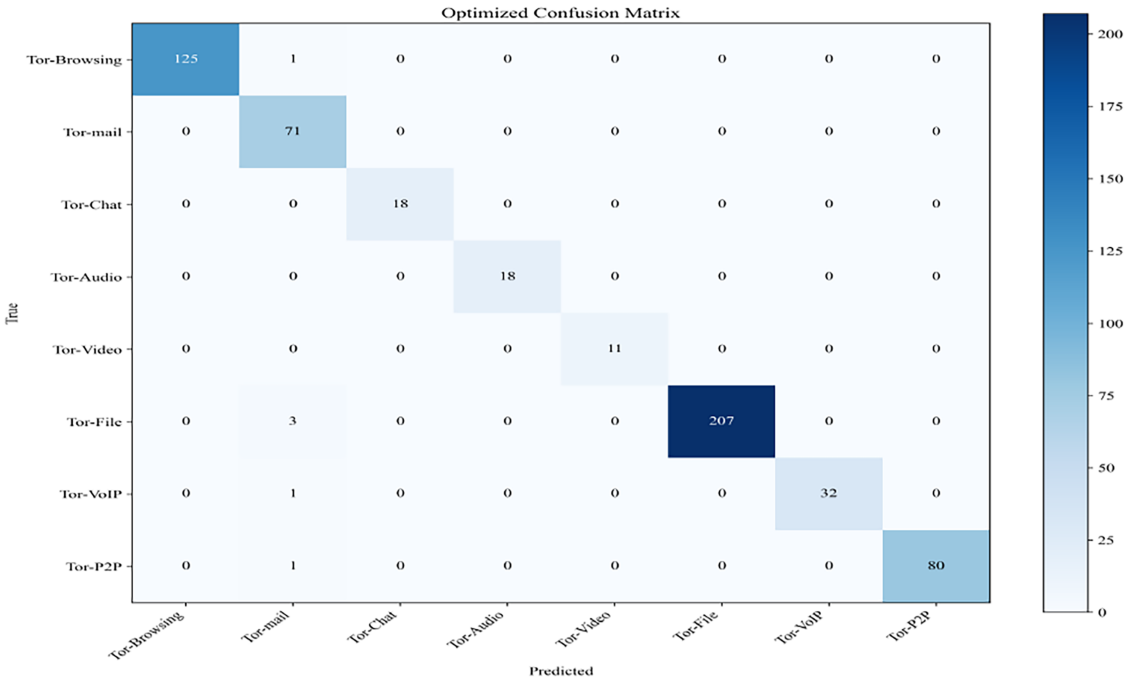

Figure 5: Confusion matrix of the proposed method on the ISCX-Tor dataset.

Overall, the sparse and low-magnitude misclassifications further verify the robustness of the proposed framework for encrypted traffic classification.

4.5 Model Sensitivity Analysis

A sensitivity analysis is conducted on three key hyperparameters, as shown in Fig. 6. Model performance improves as the embedding dimension increases to 64, while larger dimensions introduce feature redundancy and overfitting. Similarly, increasing the packet number from 10 to 50 and the byte length from 50 to 150 enhances performance by capturing richer byte-level interactions. However, further increases lead to performance saturation, higher computational cost, and noise accumulation.

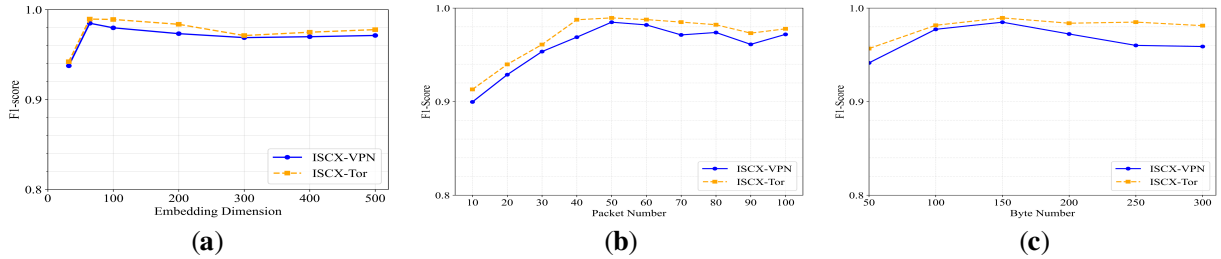

Figure 6: Sensitivity analysis of key hyperparameters: (**a**) Embedding dimension, (**b**) Packet number, (**c**) Byte number.

The performance degradation mainly stems from excessive zero-padding and increased graph sparsity. Longer byte sequences introduce large amounts of non-informative padded tokens, weakening meaningful statistical dependencies. Meanwhile, in PMI-based graph construction, low-frequency and padded tokens produce sparse and less informative adjacency structures, reducing the effectiveness of neighborhood aggregation in the GraphSAGE encoder.

Therefore, the embedding dimension, packet number, and byte length are set to 64, 50, and 150, respectively.

4.6 Impact of GNN Encoders and GraphSAGE Depth Analysis

To evaluate different graph encoders and model depths, experiments are conducted on multiple GNN architectures and GraphSAGE configurations. Fig. 7 shows the comparison of representative GNN encoders, while Fig. 8 presents the impact of GraphSAGE depth on classification performance.

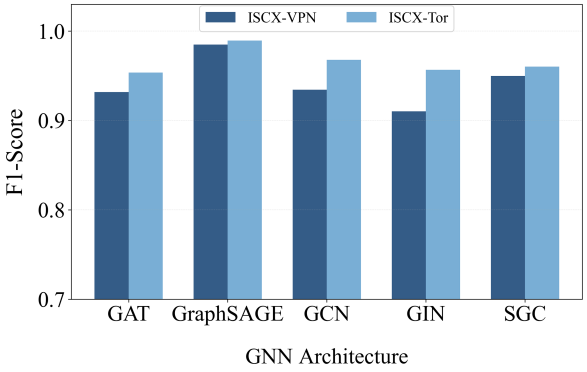

Figure 7: Performance comparison of different GNN encoders.

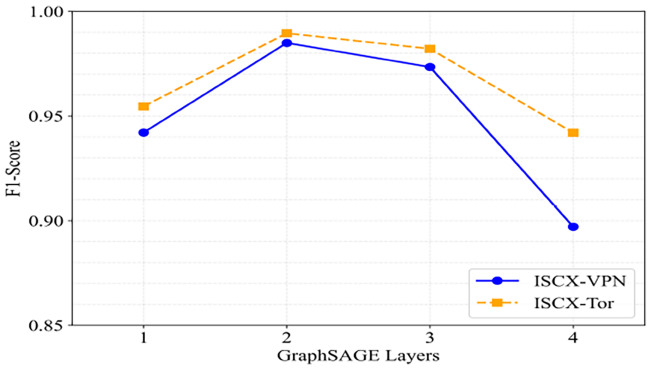

Figure 8: Layer-wise performance of GraphSAGE.

GraphSAGE consistently achieves the best or near-best F1-scores on both ISCX-VPN and ISCX-Tor, demonstrating strong and stable neighborhood aggregation. In contrast, GAT is sensitive to noisy edges, resulting in unstable performance, while GIN shows dataset-dependent behavior with better results on Tor traffic. Although GCN and SGC offer higher computational efficiency, their limited expressiveness restricts performance. Overall, these results validate GraphSAGE as the backbone encoder in DVG-GNN.

A two-layer GraphSAGE consistently achieves the best performance on both datasets. Shallow models fail to capture sufficient higher-order neighborhood information, while deeper models suffer from over-smoothing, noise propagation, and optimization difficulties such as gradient vanishing. These issues are more pronounced in heterogeneous encrypted traffic graphs. Therefore, a two-layer configuration provides a good balance between representation capacity and training stability.

4.7 Computational Cost Comparison

To evaluate computational efficiency, DVG-GNN is compared with ET-BERT, DE-GNN, and BPF-GNN on the ISCX-VPN dataset under identical settings using an NVIDIA RTX 4060 GPU with a batch size of 64.

In addition to FLOPs and parameter size, we further report training time per epoch, GPU memory consumption, and average inference latency per sample for a comprehensive efficiency evaluation. Training time is measured under identical preprocessing and batch settings, while inference latency is averaged over 10 runs after model warm-up on a single GPU.

As shown in Table 7, DVG-GNN achieves the lowest FLOPs and parameter size among all methods. ET-BERT has the highest computational cost, training time, and memory usage due to its large Transformer architecture, whereas DE-GNN and BPF-GNN reduce overhead through lightweight graph-based designs.

Table 7: Computational cost comparison with state-of-the-art models.

Model	FLOPS (M)	Parameters (M)	Train (s/epoch)	Memory (GB)	Latency (ms/Sample)
ET-BERT	1.5e+4	8.8e+1	392.2	7.1	12.6
DE-GNN	4.6e+2	7.1e−1	28.7	2.1	3.8
BPF-GNN	5.3e+2	1.7e+0	35.2	2.8	4.6
DVG-GNN	4.2e+2	5.7e−1	33.5	2.3	5.1

Notably, DVG-GNN requires only 4.2×10^2 MFLOPs and 0.57 M parameters while maintaining strong representation capability. Although a lightweight BiLSTM-attention temporal module is introduced, its additional overhead remains limited.

Compared with ET-BERT, DVG-GNN significantly reduces training time, memory consumption, and inference latency while achieving better classification performance. Compared with DE-GNN, it incurs slightly higher training cost and memory usage due to temporal modeling and dual-view fusion, but the overall overhead remains moderate considering performance gains.

Overall, DVG-GNN achieves a favorable trade-off between computational efficiency and representation capability, making it suitable for resource-constrained encrypted traffic classification.

4.8 Model Interpretability Analysis

To better understand the decision-making process of DVG-GNN, we conduct an interpretability analysis from the perspectives of attention distribution, graph structure, and dual-view fusion.

The attention-based graph pooling mechanism identifies discriminative byte regions that contribute most to traffic classification. For VPN traffic, the model focuses on protocol-related header bytes, whereas for Tor traffic it emphasizes payload dependency patterns, indicating that DVG-GNN captures meaningful semantic information rather than noise.

The PMI-based graph construction preserves statistical dependencies among packet bytes. Header graphs retain protocol-sensitive structural information, while payload graphs capture richer semantic correlations among byte sequences. The gated fusion module adaptively integrates these two complementary views, enabling joint exploitation of structural and semantic information.

To quantitatively evaluate the concentration of the learned attention distribution, we introduce attention entropy computed from the normalized attention coefficients of the graph pooling layer, as defined below:

$$H = -\sum_{i} \alpha_i \log \alpha_i \tag{27}$$

where H denotes the attention entropy and α_i represents the normalized attention weight of node i. Lower entropy indicates that the model concentrates on a smaller set of informative nodes. All interpretability metrics are evaluated on the test set without retraining. For node occlusion analysis, selected nodes are removed during inference with model parameters fixed. For PMI edge removal, high-weight edges are removed before inference and the model is re-evaluated.

The interpretability results are summarized in Tables 8–11. Table 8 reports representative samples from the ISCX-VPN and ISCX-Tor test sets, showing that lower attention entropy generally corresponds to higher prediction confidence, indicating that DVG-GNN focuses on discriminative graph regions during decision making. To further quantify this relationship, the Pearson correlation coefficient over all test samples is -0.91 ($p < 0.01$), demonstrating a strong and statistically significant negative correlation between attention concentration and prediction confidence. This suggests that more concentrated attention leads to more reliable predictions.

Table 8: Attention entropy analysis on different traffic categories.

Traffic Category	Attention Entropy	Prediction Confidence (%)
VPN-Chat	2.11	97.93
VPN-Email	1.94	98.46
VPN-Streaming	2.24	97.71
Tor-Chat	1.87	98.88
Tor-File	1.73	99.21
Tor-Video	1.82	98.90

Table 9: Average attention allocation across dual views.

Traffic Type	Header Attention (%)	Payload Attention (%)
VPN Traffic	61.8	38.2
Tor Traffic	34.5	65.5

Table 10: Node occlusion attribution analysis.

Setting	Ac	F1
Original Graph	0.9884	0.9855
Random Node Removal (5%)	0.9824	0.9816
High-Attention Node Removal (5%)	0.9747	0.9735
Random Node Removal (10%)	0.9769	0.9753
High-Attention Node Removal (10%)	0.9628	0.9611

Table 11: Impact of removing high-weight PMI edges.

Removed Edge Ratio	Ac	F1
0%	0.9884	0.9855
10%	0.9794	0.9762
20%	0.9712	0.9704
30%	0.9581	0.9639
40%	0.9336	0.9477

The reported ratios are obtained by averaging the gating coefficients generated by the fusion module over all test samples. Table 9 shows that the model primarily attends to header structures for VPN traffic and payload dependency patterns for Tor traffic, validating the complementarity of the dual-view design.

Nodes are ranked according to the attention coefficients produced by the graph pooling layer, and the top-k nodes are removed. As shown in Table 10, removing high-attention nodes results in substantially larger performance degradation than random removal, indicating that the attention mechanism successfully identifies task-critical nodes. The results are obtained on the ISCX-VPN dataset.

Edges are ranked by PMI weights and removed in descending order. Table 11 shows a gradual decline in classification performance as high-weight PMI edges are removed, highlighting the importance of structural dependencies captured by PMI-based graph construction.

Overall, these results provide quantitative evidence for the interpretability of DVG-GNN, demonstrating that attention modeling, structural learning, and dual-view fusion jointly enhance classification reliability in encrypted traffic analysis.

4.9 Security Robustness Evaluation Against Obfuscation and Adversarial Perturbation

To evaluate the robustness of the proposed framework in realistic encrypted traffic environments, experiments are conducted under three settings: traffic obfuscation, adversarial perturbation, and cross-scenario evaluation. All robustness experiments are performed on the ISCX-VPN dataset.

For adversarial evaluation, FGSM attacks are applied to normalized packet byte features with a perturbation strength of $\epsilon = 0.03$, affecting at most 5% of byte positions while preserving traffic semantics. FGSM perturbations are introduced only during testing. For obfuscation evaluation, payload padding, packet fragmentation, temporal reordering, and random byte masking are employed to simulate common traffic transformation techniques. Cross-scenario evaluation follows the train-test split described in Section 4.1, where application categories in the test set are excluded from the training distribution.

Each experiment is repeated 10 times with different random seeds, and the mean F1-score and standard deviation are reported. Since DE-GNN is the strongest graph-based baseline, paired t-tests are conducted between DVG-GNN and DE-GNN at a 95% confidence level ($p < 0.05$).

As shown in Table 12, all methods suffer performance degradation under obfuscation, adversarial perturbation, and cross-scenario settings; however, DVG-GNN consistently achieves the best performance.

The small standard deviations demonstrate stable performance across repeated runs, while paired t-tests confirm that the improvements over DE-GNN are statistically significant ($p < 0.05$). DVG-GNN achieves F1-scores of 0.9184 and 0.8732 under obfuscation and adversarial perturbation, surpassing DE-GNN by 1.69 and 2.06 percentage points, respectively. Similar gains are observed in the cross-scenario evaluation, indicating strong robustness and generalization under traffic transformations and distribution shifts.

Table 12: Robustness comparison under obfuscation, adversarial perturbation, and cross-scenario evaluation (F1-score ± Std). *Indicates statistically significant improvement over DE-GNN ($p < 0.05$).

Model	Normal	Obfuscation	Adversarial	Cross-Scenario
1D-CNN	0.8490 ± 0.0071	0.7914 ± 0.0086	0.7218 ± 0.0108	0.7486 ± 0.0095
DE-GNN	0.9437 ± 0.0042	0.9015 ± 0.0054	0.8526 ± 0.0067	0.8813 ± 0.0061
BPF-GNN	0.9468 ± 0.0045	0.8942 ± 0.0058	0.8417 ± 0.0072	0.8734 ± 0.0065
DVG-GNN	0.9855 ± 0.0031	0.9184* ± 0.0044	0.8732* ± 0.0052	0.8945* ± 0.0048

Overall, the results demonstrate that DVG-GNN maintains reliable performance under obfuscation, adversarial perturbations, and unseen traffic distributions, highlighting its practical applicability to real-world encrypted traffic classification.

5 Limitations and Future Work

Despite the strong performance of DVG-GNN, several limitations remain. First, the current graph construction strategy mainly focuses on local statistical dependencies and may not fully capture long-range interactions in complex encrypted traffic. Future work will explore more adaptive graph construction mechanisms to enhance long-range dependency modeling.

Second, the experiments are primarily conducted on public benchmark datasets, lacking validation on large-scale real-world encrypted traffic scenarios. Future research will further evaluate DVG-GNN in real network environments to verify its scalability, robustness, and practical applicability.

6 Conclusion

This paper proposes DVG-GNN, a graph neural network-based framework for encrypted traffic classification. By jointly modeling header and payload information through a dual-view design, DVG-GNN enhances both structural and semantic representations. The framework combines multi-scale convolution and PMI-based graph construction to capture contextual and structural dependencies, while a GraphSAGE encoder with attention pooling and gated fusion learns discriminative graph-level features.

Extensive experiments on ISCX-VPN, ISCX-Tor, and USTC-TFC2016 demonstrate that DVG-GNN consistently achieves superior classification performance and strong generalization capability across diverse encrypted traffic scenarios. Future work will focus on dynamic graph construction, multi-flow modeling, and large-scale real-world deployment to further improve scalability and adaptability.

Acknowledgement: Not applicable.

Funding Statement: This work was supported by the National Natural Science Foundation of China (Grant No. 62573298), the Guangdong Provincial Key Laboratory (Grant No. 2023B1212060076), the Henan Province Science and Technology Tackling Key Problems Plan Project (Grant No. 252102210173 and 252103810209), the Key Research Projects of Higher Education Institutions in Henan Province (Grant No. 25A520051, 24A520011, and 24A520008), the Key Research and Development Program of Henan Province (Grant No. 261111211200), the Natural Science Foundation of Henan Province (Grant No. 252300421507), the Henan Provincial Higher Education Teaching Reform Research and Practice Project (Grant No. 2026SJGLX192), and the Research Project on Education and Teaching Reform of Henan University of Engineering (Grant No. 2024JYYB020).

Author Contributions: Guan Yang was responsible for conceptualization and overall study design. Haozhen Wang designed the core methodology and implemented the corresponding code. Guan Yang and Haozhen Wang jointly

conducted the investigation and data curation.Yu Wang contributed to validation and provided critical resources. Weiguang Liu was responsible for project administration and funding acquisition. Bo Chen supervised the project and also contributed to funding acquisition. Guan Yang and Haozhen Wang drafted the original manuscript. Yu Wang, Weiguang Liu, and Bo Chen reviewed and edited the manuscript. Haozhen Wang created the visualizations. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The data that support the findings of this study are openly available in the public repositories ISCXVPN2016, ISCXTOR2016, and USTC-TFC2016, with the download URLs: https://www.unb.ca/cic/datasets/vpn.html (ISCXVPN2016), https://www.unb.ca/cic/datasets/tor.html (ISCXTOR2016), and https://github.com/yungshenglu/USTC-TFC2016 (USTC-TFC2016).

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Alwhbi IA, Zou CC, Alharbi RN. Encrypted network traffic analysis and classification utilizing machine learning. Sensors. 2024;24(11):3509. doi:10.3390/s24113509.
2. Alshammari R, Zincir-Heywood AN. Machine learning based encrypted traffic classification: identifying SSH and skype. In: Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications; 2009 Jul 8–10; Ottawa, ON, Canada. p. 1–8.
3. Liu C, He L, Xiong G, Cao Z, Li Z. FS-net: a flow sequence network for encrypted traffic classification. In: Proceedings of the IEEE INFOCOM 2019—IEEE Conference on Computer Communications; 2019 Apr 29–May 2; Paris, France. New York, NY, USA: IEEE; 2019. p. 1171–9.
4. Liu Z, Wei Q, Song Q, Duan C. Fine-grained encrypted traffic classification using dual embedding and graph neural networks. Electronics. 2025;14(4):778. doi:10.3390/electronics14040778.
5. Chen Z, Wei X, Wang Y. Encrypted traffic classification encoder based on lightweight graph representation. Sci Rep. 2025;15:28564.
6. Okonkwo Z, Foo E, Hou Z, Li Q, Jadidi Z. A graph representation framework for encrypted network traffic classification. Comput Secur. 2025;148(4):104134. doi:10.1016/j.cose.2024.104134.
7. Liu J, Zeng Y, Shi J, Yang Y, RuiWang, He L. MalDetect: a structure of encrypted malware traffic detection. Comput Mater Continua. 2019;60(2):721–39. doi:10.32604/cmc.2019.05610.
8. van Ede T, Bortolameotti R, Continella A, Ren J, Dubois DJ, Lindorfer M, et al. Flowprint: semi-supervised mobile-app fingerprinting on encrypted network traffic. In: Proceedings of the Network and Distributed System Security Symposium (NDSS); 2020 Feb 23–26; San Diego, CA, USA. 27 p.
9. Huang H, Zhang X, Lu Y, Li Z, Zhou S. BSTFNet: an encrypted malicious traffic classification method integrating global semantic and spatiotemporal features. Comput Mater Continua. 2024;78(3):3929–51. doi:10.32604/cmc.2024.047918.
10. Park JT, Choi YS, Cho BS, Kim SH, Kim MS. Multi-level pre-training for encrypted network traffic classification. IEEE Access. 2025;13(8):68643–59. doi:10.1109/access.2025.3559068.
11. Pang B, Fu Y, Ren S, Wang Y, Liao Q, Jia Y. CGNN: traffic classification with graph neural network. arXiv:2110.09726. 2021.
12. Zeng Q, Qu D, Zhang H, Chen Y, Zhang W. SCNNTraffic: a lightweight and energy-efficient traffic classification method based on spiking convolution neural networks. Peer-to-Peer Netw Appl. 2026;19(2):52. doi:10.1007/s12083-026-02212-y.
13. Huoh TL, Luo Y, Zhang T. Encrypted network traffic classification using a geometric learning model. In: Proceedings of the 2021 IFIP/IEEE International Symposium on Integrated Network Management (IM); 2021 May 17–21; Bordeaux, France. p. 376–83.
14. Hamilton W, Ying Z, Leskovec J. Inductive representation learning on large graphs. Adv Neural Inf Process Syst. 2017;30:1025–35.

15. Gil GD, Lashkari AH, Mamun MSI, Ghorbani AA. Characterization of encrypted and VPN traffic using time-related features. In: Proceedings of the 2nd International Conference on Information Systems Security and Privacy; 2016 Feb 19–21; Rome, Italy. p. 407–14.
16. Lashkari AH, Gil GD, Mamun MSI, Ghorbani AA. Characterization of tor traffic using time based features. In: Proceedings of the International Conference on Information Systems Security and Privacy; 2017 Feb 19–21; Porto, Portugal. p. 253–62.
17. Shapira T, Shavitt Y. FlowPic: a generic representation for encrypted traffic classification and applications identification. IEEE Trans Netw Serv Manage. 2021;18(2):1218–32. doi:10.1109/tnsm.2021.3071441.
18. Wang W, Zhu M, Zeng X, Ye X, Sheng Y. Malware traffic classification using convolutional neural network for representation learning. In: Proceedings of the 2017 International Conference on Information Networking (ICOIN); 2017 Jan 11–13; Da Nang, Vietnam. p. 712–7.
19. Wang W, Zhu M, Wang J, Zeng X, Yang Z. End-to-end encrypted traffic classification with one-dimensional convolution neural networks. In: Proceedings of the 2017 IEEE International Conference on Intelligence and Security Informatics (ISI); 2017 Jul 22–24; Beijing, China. p. 43–8.
20. Lin X, Xiong G, Gou G, Li Z, Shi J, Yu J. ET-BERT: a contextualized datagram representation with pre-training transformers for encrypted traffic classification. In: Proceedings of the ACM Web Conference 2022; 2022 Apr 25–29; Lyon France. p. 633–42.
21. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. Attention is all you need. Adv Neural Inf Process Syst. 2017;30:6000–10. doi:10.65215/2q58a426.
22. He HY, Yang Z, Chen XN. PERT: payload encoding representation from transformer for encrypted traffic classification. In: Proceedings of the 2020 ITU Kaleidoscope: Industry-Driven Digital Transformation (ITU K); 2020 Dec 7–11; Ha Noi, Vietnam. p. 1–8.
23. Diao Z, Xie G, Wang X, Ren R, Meng X, Zhang G, et al. EC-GCN: a encrypted traffic classification framework based on multi-scale graph convolution networks. Comput Netw. 2023;224:109614. doi:10.1016/j.comnet.2023.109614.
24. Huoh TL, Luo Y, Li P, Zhang T. Flow-based encrypted network traffic classification with graph neural networks. IEEE Trans Netw Serv Manage. 2023;20(2):1224–37. doi:10.1109/tnsm.2022.3227500.
25. Han X, Xu G, Zhang M, Yang Z, Yu Z, Huang W, et al. DE-GNN: dual embedding with graph neural network for fine-grained encrypted traffic classification. Comput Netw. 2024;245(5):110372. doi:10.1016/j.comnet.2024.110372.
26. Shen M, Zhang J, Zhu L, Xu K, Du X. Accurate decentralized application identification via encrypted traffic analysis using graph neural networks. IEEE Trans Inform Forensic Secur. 2021;16:2367–80. doi:10.1109/tifs.2021.3050608.
27. Li G, Gao Y, Ren J, Chen S. BPF-GNN: a multi-granularity feature extraction model using graph neural networks for encrypted traffic classification. IEEE Trans Netw Serv Manage. 2026;23(2):3105–17. doi:10.1109/tnsm.2026.3671203.
28. Liu B, Chen X, Yuan Q, Li D, Gu C. TMC-GCN: encrypted traffic mapping classification method based on graph convolutional networks. Comput Mater Continua. 2025;82(2):3179–201. doi:10.32604/cmc.2024.059688.