Dual-Channel Interactive Graph Transformer for Traffic Classification with Message-Aware Flow Representation

Xing Qiu[1], **Guang Cheng**[1,2,3*], **Weizhou Zhu**[1], **Dandan Niu**[1], **Nan Fu**[1]
[1]Southeast University, Nanjing, China
[2]Purple Mountain Laboratories, China
[3]Zhongguancun Laboratories, China
{230218524, chengguang, 220224855, 220215222, 230229203}@seu.edu.cn

Abstract

Traffic classification is crucial for network management and security. Recently, deep learning-based methods have demonstrated good performance in traffic classification. However, they primarily capture features from raw packet bytes, overlooking the significance of inter-packet correlations within flows from a global perspective. Additionally, effectively handling both packet-length and temporal information, while extracting the structural relationships from a graph into the model, remains a challenge for enhancing the performance of traffic prediction. In this paper, we propose DigTraffic, a novel dual-channel interactive graph transformer to address these limitations. DigTraffic employs a message-level graph-structured flow representation combined with message-aware structural aggregation. To learn intrinsic flow representations, DigTraffic constructs traffic interaction graphs, by incorporating three well-designed heterogeneous types of edges to capture client-server interactions. After that, we separately encode lengthy and temporal flow sequences using a dual-channel network and fuse these modalities within a Transformer architecture. Furthermore, DigTraffic introduces a message-aware Graph Transformer that leverages both node embeddings and edge spatial relations to capture complex graph structures and rich structural information. Experimental results demonstrate that our method significantly outperforms the state-of-the-art methods on four real-world traffic datasets.

Code — https://github.com/SeuXing/DigTraffic

1 Introduction

Traffic classification has become a vital technique, allowing third-party passive observers to identify application categories from encrypted traffic. Understanding traffic content enables network operators to swiftly apply tailored network management strategies, enhancing service quality and user experience while supporting diverse business goals (Panchenko et al. 2016). Moreover, such advancements could infer more granular user privacy, including online shopping behaviors and even insights into political inclinations. However, rising usage of encrypted traffic and anonymous network technologies complicates analysis, making the accurate classification of complex traffic patterns difficult. To build a refined traffic analyzer for more precise classification, it is crucial to mine latent and robust representation within both traffic flows and message interactions.

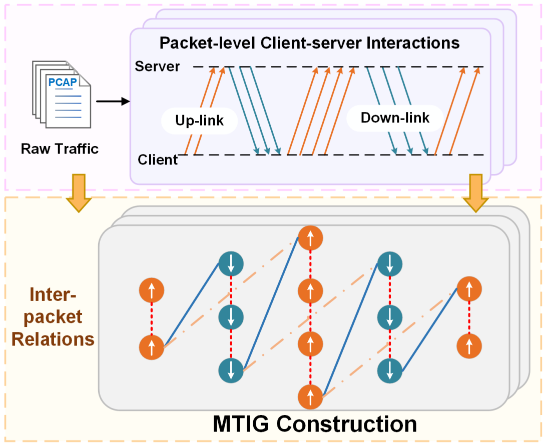

Figure 1: The construction of Message-level Traffic Interaction Graphs (MTIG)

Traditional traffic analysis methods employ hand-crafted features to train Machine Learning (ML) classifiers, requiring significant manual effort and resources to construct specific traffic features. They may also result in severe overfitting due to their heavy reliance on specific traffic features and particular communication patterns. Alternatively, Deep Learning (DL) techniques can automatically extract features from raw traffic packets, and effectively leverage underlying interactive information in message transmission (Zhao et al. 2022b). However, those DL-based methods lack the ability to represent semantic relationships between packets on a global insight, causing performance instability. Besides, simultaneously integrating lengthy and temporal features from traffic modalities to holistically guide the model poses a significant challenge. Therefore, a graph-level flow representation of transmitted packets is necessary to capture the full range of interactive messages across modalities.

Graph Neural Networks (GNNs) have achieved remarkable success in various domains, particularly in Computer

*Corresponding author.

Vision (CV) (Wang et al. 2019a) and Natural Language Processing (NLP) (Liu et al. 2020). Recent studies in traffic classification directly employ GNNs to capture the relationships within traffic flows and obtain good performance. However, the main limitation is that they learn flow representations solely by combining information from local packet neighbors, which may encounter challenges in ineffective structural connections. In addition, the issue of over-squashing causes information loss throughout GNN's iterative message-passing, which consequently degrades model performance. Therefore, it is crucial to develop a new architecture that extends beyond local neighborhood aggregation of packet nodes.

The Transformer architecture is seen as a promising solution because it can directly model distant nodes and effectively capture long-range dependencies. The key to effectively utilizing Transformers for graph data lies in the ability to encode the structural information of the graph into the model. A notable example is Graphormer (Ying et al. 2021), which adapts Transformer to effectively process graph-structured data by incorporating spatial encoding and attention mechanisms. Although Graphormer excels at graph-level representation learning tasks, using raw traffic packets directly can lead to insufficient representations of message-level traffic interactions.

To address the challenges mentioned above, we propose DigTraffic, a novel dual-channel interactive graph Transformer that learns latent representations from both multi-modal traffic nodes and edge correlations through message interactions. First, we construct Message-level Traffic Interaction Graphs (MTIG) to depict the raw traffic, as illustrated in Fig. 1. Three heterogeneous edge types are meticulously designed to model up-link and down-link message interactions. Then, we implement dual-channel encoders to extract and fuse these features from lengthy and temporal flow sequence modalities into node embeddings. Furthermore, we propose a spatial encoder by combing edge relations to guide the message-aware attention mechanism, along with a centrality encoder capturing node importance in graphs. These two strategic encoding modules enable DigTraffic to learn both edge-level and node-level flow representations. Our main contributions are summarized as follows:

- We propose a dual-channel interactive graph transformer with message-aware flow graph construction, called DigTraffic, for traffic classification. DigTraffic breaks through traditional traffic analysis methods in three aspects: graph-level traffic representation, dual-channel modal encoding, and graph transformer network.
- We design MTIG structure that fully considers traffic patterns within message interactions. MTIG leverages packet-level information as nodes, and learns inherent semantic relations in message-level interactions. We construct dual-channel modal encoders to learn both lengthy and temporal modalities.
- We introduce effective structural encoding methods guiding the message-aware attention mechanism to better model our graph-structured data. Those designed centrality and spatial encoders could help DigTraffic fully capture graph representations in node-level and message-level structural perspectives.
- We evaluate DigTraffic on four diverse real-world traffic datasets. Experimental results demonstrate that our method outperforms the state-of-the-art methods, showcasing its wide-ranging effectiveness.

2 Related Work

2.1 Traffic Analysis Methods

ML-Based Methods. As encrypted traffic has been widely used and network environments become more complicated, traffic analysis task faces significant challenges. To analyze complicated traffic, several studies employ traditional machine learning methods to design classifiers. CUMUL (Panchenko et al. 2016) is designed as a representative length feature that constructs cumulative packet length along with an SVM, for classifying website traffic. Joint statistical features are utilized to construct the model for classifying encrypted traffic (Shen et al. 2017; Liu et al. 2021). ML-based methods, though capable of analyzing complex traffic with statistical features, rely heavily on manual features and risk model over-fitting.

DL-Based Methods. Deep learning continues to flourish in traffic classification which facilitates the automatic extraction of abstract features from the raw packets, rather than relying on human-designed features. FS-Net (Liu et al. 2019) employs a Recurrent Neural Network (RNN) model to handle original flow sequences and extracts the sequential relationships. BurNet (Shen et al. 2021a) leverages a convolutional neural network (CNN) to capture spatial hierarchies in traffic sequences. The Transformer (Vaswani et al. 2017) becomes popular due to its ability using attention mechanism to capture global long-range dependencies. Recently, some works attempt to employ Transformer-based model (Zhou et al. 2024; Cui et al. 2023) to examine the generic representation of encrypted traffic. YaTC (Zhao et al. 2023, 2024) leverages the pre-trained Transformer and fine-tunes parameters for traffic identification. However, few methods focus on capturing the inter-element correlations and structural information within packets, resulting in models unable to represent the inherent traffic characteristics.

2.2 GNN-based Methods and Graph Transformer

To effectively capture local and global combinations between packets within traffic interactions, GNNs are gradually being employed in traffic classification. GraphDApp (Shen et al. 2021b) proposes an information-rich representation of encrypted flows for identifying each decentralized application (app). HyperVision (Fu, Li, and Xu 2023) represents basic GNN to design graph structural features for exploiting relationships between traffic. Some works (Zhao et al. 2022a; Fu, Cheng, and Su 2023) introduces a designed graph convolutional networks (GCN) to mine connections between flow sequences for anonymity traffic identification. However, one primary limitation of GNN models is that they derive flow representations solely from local information within packet neighbors, ignoring crucial

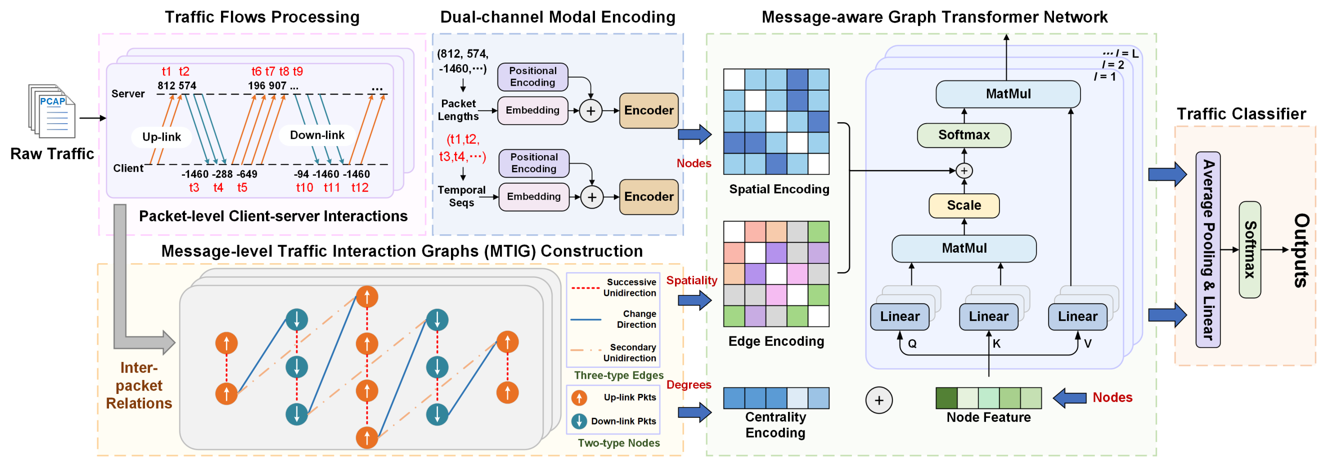

Figure 2: The framework of proposed DigTraffic.

structural aggregations. Besides, the issue of over-squashing leads to information loss and distortion, subsequently impairing model performance. To mitigate the aforementioned limitations, Graphormer (Ying et al. 2021) as a Transformer-based network for graph-level representations, comes to the forefront. This novel architecture achieves outstanding performance across a wide array of graph-based learning tasks, such as drug prediction (Su et al. 2024). In traffic classification task, it is crucial to construct traffic graphs which highlight spatial message interactions, and fully exploit features from both packet lengths and temporal sequences. Our proposed DigTraffic addresses these aspects and incorporates carefully designed centrality and edge encoding modules, enhancing Transformer attention to effectively extract information-rich structural flow representations.

3 Methodology

The architecture of DigTraffic is depicted in Fig. 2. We introduce DigTraffic in terms of graph-level traffic representation, dual-channel modal encoder structure, and message-aware graph transformer network in detail in Section 3.1, Section 3.2, and Section 3.3, respectively.

3.1 Graph-level Traffic Representation

We construct a novel traffic interaction graph structure at message level, carefully considering the whole transmission patterns of packets in client-server interactions. Unlike directly proposing graphs from a single flow or a cluster of few flows, we depict the communication of all packets between the client and server.

Client-Server Interactions. The communication process begins with the client sending an uplink request to the server for specific contents. Upon receiving the request, the server responds and sends back the required downlink messages. In the transport layer of network, we observe those messages are partitioned into one or multiple transmitted packets due to traffic transmission mechanism. We assume that each message M_a consists of those unidirectional successive packets in order $(p_1, \ldots, p_m)$, where m is the number of packets.

Traffic Flows Processing. To specifically illustrate the client-server interactions through packets and their directions, we first partition the whole sequence into individual flows, each defined as a series of packets with the same 5-tuple (source/destination IP addresses, source/destination ports, and protocol). A flow F_i with n packets is represented as a packet-level vector $F_i = (p_1, \ldots, p_n)$. Each packet p_i includes a signed length and interval, and reduces its header length (TCP header in 20 bytes and UDP header in 8 bytes). We set those uplink packet lengths as *positive* and downlink ones as *negtive*, denoted by the direction d_i. We then pad flows into identical-length packet sequences and put them into graph construction module. However, the vectors possess a constrained capacity to effectively represent client-server interactions. Motivated by the detailed insights provided by graph structures, we aim to depict inter-packet relations within flows using a graph, namely MTIG.

Message-level Traffic Interaction Graphs. Note that $G = (V, E)$ denotes a graph where $V = (v_1, v_2, \ldots, v_n)$, and n is the number of nodes. Our proposed MTIG represents packet-level entities in two directions as two types of nodes. The formula as follows:

$$V = p_{uplink} \cup p_{downlink} \tag{1}$$

More importantly, it incorporates direction-sensitive traffic transmission processes by designing three heterogeneous types of edges. In specific, here are three distinct types of message relationships between packet pairs depicted in Fig. 1, which are most worthy of our consideration:

- **(1) Successive Uni-direction Edges**. A flow typically contains those up-link and down-link blocks where a series of successive packets are transmitted along the same direction ($d_i = d_{i+1}$). In the view of traffic transmission, a block here indicates one entire message which is transmitted in the flow. We connect the two adjacent packets p_i and p_{i+1} in the same direction because they are spatially linked in one single message, showing **Intra-message** relations as follows:

$$E_1 = \{(p_i, p_{i+1}) \mid d_i = d_{i+1}, \forall i\} \tag{2}$$

- **(2) Changed Direction Edges**. Clearly, when the direction between two consecutive packets changes, it means that the current message has been completely transmitted and the subsequent message is scheduled for delivery. We connect two neighboring packets in opposite directions ($d_i \neq d_{i+1}$), indicating a change in the direction of **Inter-message** interactions. It is defined as:

$$E_2 = \{(p_i, p_{i+1}) \mid d_i \neq d_{i+1}, \forall i\} \tag{3}$$

- **(3) Secondary Uni-direction Edges**. We also consider the connections between adjacent messages in the same direction ($d_{M_a} = d_{M_c}$, while d_{M_b} is the opposite message), which shows the continuous transmission of contents by the client or server individually. Specifically, we link the first packet p_1 in one message M_a with the last packet p_m in the preceding message M_c. This type of link can help the model learn long-distance latent dependencies from **High-order Neighboring** packets, better integrating local neighbors and structural information from a global perspective. The operation is defined as:

$$E_3 = \{(p_1 \in M_a, p_m \in M_c) \mid d_{M_a} = d_{M_c}, \forall M\} \tag{4}$$

To this end, we fully learn message-level structural information from all three types of edge connections and later delicately model the graph-structured traffic data into the proposed graph Transformer.

3.2 Dual-channel Multi-modal Encoding

In the multi-modal encoding phrases shown in Fig. 2, We process packet length and timing into distinct traffic sequences, embed positional information, and use parallel Transformer encoders to extract dual-modal features, which are then fused using an MLP.

Packet Length and Temporal Sequences. Due to the subsequent use of Transformer encoding, we truncate whole flows into fixed-length segments and pad shorter sequences for uniformity. We exclude zero payload values from packet length sequences, as they represent state information rather than message transmission. Simultaneously, we convert the time series to relative times (setting the first time to zero) to simplify computations. These two input sequences are generated as follows:

$$LSeq_{\text{fixed}} = Pad(Truncate(F_i, L), L) \tag{5}$$

$$TSeq_{rel} = \{t_i - t_1 \mid t_i \in T\} \tag{6}$$

where F_i denotes the original packet length or timestamp sequence, L is the fixed sequence length. $Truncate$ function cuts the sequence to length L and Pad function extends shorter flows to L. Meanwhile, $TSeq_{rel}$ is the adjusted time sequence where each timestamp t_i is subtracted by the first timestamp t_1, converting absolute times into relative times.

Embeddings and Positional Encodings. The embedding in the Transformer converts numerical elements (packet length or temporal values) in sequences into continuous feature vectors Z. By mapping input elements to vectors in a high-dimensional space, embeddings capture semantic relationships between the elements.

Positional embeddings are used to preserve positional information within a sequence, calculated as follows:

$$PE(pos, 2i) = \sin\left(\frac{pos}{10000^{\frac{2i}{d_{\text{model}}}}}\right) \tag{7}$$

$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{\frac{2i}{d_{\text{model}}}}}\right) \tag{8}$$

Here, pos represents the position of the corresponding element within the sequence for the i-th dimension of the word embedding length spaces d_{model}. It alternates between sine and cosine functions across even and odd indices i respectively.

Attention Mechanism. The Transformer excels at handling those sequential inputs and grasping the long-distance dependencies using attention mechanism. Attention mechanism operates by generating queries Q, keys K and values V through linear transformations of the embedded features Z, using randomly initialized matrices W_q, W_k and W_v that are refined during training. We compute the dot product of Q and K, scaling it by $\sqrt{d_k}$ to moderate the magnitude. Softmax normalizes the result into a probability distribution. Finally, we multiply the result by the matrix V to achieve a weight summation. The equation is calculated as :

$$(Q, K, V) = (W_q Z, W_k Z, W_v Z) \tag{9}$$

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{10}$$

Fusing Dual Transformers as Nodes. Since dual modal embeddings of masked traffic sequences have been complicated, we leverage separate Transformer to parallel learn both lengthy and temporal semantic relationships. :

$$\text{L\_out} = TransformerEncoder_L(\text{length\_embed}, \text{mask}) \tag{11}$$

$$\text{T\_out} = TransformerEncoder_T(\text{time\_embed}, \text{mask}) \tag{12}$$

Following the separate encoder modeling, an MLP aggregates the two modalities into the node features:

$$\text{Node\_embed} = MLP(Concat[\text{L\_out}, \text{T\_out}]) \tag{13}$$

3.3 Message-aware Graph Transformer Network

The three novel graph structural encoders, namely centrality, spatial and edge encoding blocks, are meticulously designed and integrated with attention mechanisms and node features, as shown in Fig. 2. Hence, the proposed message-aware graph Transformer network surprisingly performs well on capturing complex graph-level structures from both nodes and global edge correlations.

Centrality Encoding. The attention distribution is determined by the semantic correlation between nodes, yet node centrality, a key indicator of a node's importance, is crucial for understanding the graph. For instance, the packet pairs which are selected to construct three typical types of MTIG structure hold more essential information (detailed in Section 3.1). Since such factors are valuable for Transformer network, we incorporate the position embedding specified by node degree and add it to node features as follows:

$$x_v^{(0)} = x_v + \mathbf{z}_{\deg}(v) \tag{14}$$

where $x_v^{(0)}$ is the input to the message-aware attention mechanism, $\deg(v)$ indicates the degree of v, and $\mathbf{z}()$ is an embedding specified by node degree.

Spatial Encoding. To effectively encode the structural information of a graph into the model, we propose the spatial encoding to measure the spatial relations between node pairs within the graph. Concretely, we propose a function $\phi(v_i, v_j)$ where the spatial relations are measured between each v_i and v_j connected in graph G. In this paper, we employ $\phi(v_i, v_j)$, calculated by the *Floyd* Algorithm, to be the distance of the shortest path (SPD) between those node pairs. We assign each output a learnable scalar as a bias into the self-attention detailed in Eq. 10, and the refined matrix *Attention* is following:

$$Attention_{ij} = \frac{(h_i W_Q)(h_j W_K)^T}{\sqrt{d}} + b_\phi(v_i, v_j) \tag{15}$$

where $Attention_{ij}$ indicates the site of element in matrix, $b_\phi(v_i, v_j)$ is the adjustable parameters indexed by $\phi(v_i, v_j)$.

The proposed spatial encoding enhances Transformer, allowing each node to adaptively attend to all others according to the graph's structural information.

Edge Encoding in Attention Mechanism. In our graph-level traffic representation task, three types of constructed edges mentioned in Section 3.1 all enrich the structural features. Hence, we connect edges with the attention mechanism to estimate correlations for node pairs. First, We acquire the shortest path $SP_{ij} = (e_1, e_2, \dots, e_n)$ from node v_i to v_j. Subsequently, we compute an average of the dot-products of edges along with a learnable embedding. We set the edge encoding c_{ij} as:

$$c_{ij} = \frac{1}{N} \sum_{n=1}^{N} x_{e_n} \left(w_n^E\right)^T \tag{16}$$

where x_{e_n} indicates the features of n-th edge e_n in SP_{ij}, w_n^E the indexed weights and E the edge feature.

Finally, we add this edge encoding into spatial features and modify the self-attention (Eq. 15) as follows:

$$Attention_{ij} = \frac{(h_i W_Q)(h_j W_K)^T}{\sqrt{d}} + b_\phi(v_i, v_j) + c_{ij} \tag{17}$$

4 Experiments

4.1 Experiment Settings

Datasets. Our experiments are conducted on four public and real-world encrypted traffic datasets, namely ISCX-VPN2016 (Draper-Gil et al. 2016), ISCX-Tor2016 (Lashkari et al. 2017), QUIC2019 (Rezaei and Liu 2018) and CI-CIoT2022 (Dadkhah et al. 2022), respectively.

Implementation Details. In the training stage, we set the total epoch to 300 and the batch size is 128. The basic learning rate is 3×10^{-4} along with customized Warm-up strategy. In Graph Transformer, we set 8 heads and its dimension to 256, and layers come to 3. While in dual-channel encoders, 2 heads and one layer of each is employed for lightweight and model efficiency. All evaluations are performed using Python 3.10.9 with PyTorch version 1.13.1 and executing on the PC with Intel® Core$^{\text{TM}}$ i9-14900K CPU and two NVIDIA GeForce RTX4090 GPU.

Evaluation Metrics. We classify target traffic based on the trained classifier's prediction probabilities. A flow sequence exceeding a specified threshold is deemed a True Positive (TP), while those below are False Negatives (FN). Correctly identifying an unseen flow results in True Negatives (TN), and the incorrect yields False Positives (FP). Accuracy is the ratio of correctly predicted flows to the total traffic. Recall, Precision, and F1 score are calculated:

$$\text{Recall} = \frac{TP}{TP + FN}, \quad \text{Precision} = \frac{TP}{TP + FP}, \tag{18}$$

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{19}$$

4.2 Comparison with State-of-the-Arts

To comprehensively evaluate our proposed method, we compare DigTraffic with a board range of baselines and state-of-the-art methods as follows:

- **CUMUL** (Panchenko et al. 2016) and **AppScanner** (Taylor et al. 2016) are ML-based methods using manual statistical features for traffic classification.
- **FS-Net** (Liu et al. 2019), **DeepPacket** (Lotfollahi et al. 2020), **3D-CNN** (Zhang et al. 2020), **bi-LSTM** (Chen et al. 2022), **Transformer** (Zhou et al. 2024) are DL-based methods for identifying traffic. They use raw traffic data as inputs for model training.
- **GCN-RTG** (Fu, Cheng, and Su 2023) and **GAT** (Wang et al. 2019b) are GNNs which construct traffic graphs and extract structural features from graph learning networks.

Main Results compared to SOTA. As is shown in Table 1, DigTraffic outperforms the other methods significantly across all four datasets. Our method achieves highest $Acc.$ and F_1 surpassing the second-best by 8.02% and 8.08% on ISCX-VPN2016, 12.95% and 13.02% on ISCX-Tor2016, 7.71% and 7.72% on QUIC2019, 5.06% and 5.27% on CI-CIoT2022, respectively. Concretely, those ML-based methods using manual features and traditional classifiers struggle to mine discriminative traffic features, whereas DL-based methods excel at processing sequential inputs. Although CNNs (3D-CNN) can identify local patterns through kernels, this advantage is diminished when handling the one-dimensional nature of traffic data. While RNNs (FS-Net and bi-LSTM) effectively handle sequential data, they often struggle with long-term dependencies. This limitation

Method	ISCX-VPN2016		ISCX-Tor2016		QUIC2019		CICIoT2022	
	Acc.	F_1	Acc.	F_1	Acc.	F_1	Acc.	F_1
CUMUL	76.21%	72.92%	62.07%	60.71%	70.30%	65.59%	63.82%	59.66%
AppScanner	75.82%	72.75%	60.51%	51.36%	63.13%	57.07%	69.14%	66.91%
DeepPacket	75.43%	73.73%	46.91%	45.02%	58.49%	54.96%	72.11%	69.74%
3D-CNN	76.60%	74.48%	69.82%	66.53%	65.45%	63.46%	78.58%	77.42%
FS-Net	82.06%	79.28%	70.50%	70.18%	69.95%	65.00%	81.84%	80.15%
bi-LSTM	88.92%	88.49%	73.33%	72.70%	69.63%	67.47%	84.91%	84.69%
Transformer	87.86%	87.83%	82.02%	81.52%	88.70%	88.54%	87.04%	86.95%
GCN-RTG	89.73%	89.28%	85.30%	84.90%	82.31%	81.97%	91.55%	90.94%
GAT	90.12%	89.88%	85.92%	85.33%	81.50%	81.43%	90.85%	90.41%
Ours (DigTraffic)	**98.14%**	**97.96%**	**98.57%**	**98.35%**	**96.41%**	**96.26%**	**96.91%**	**96.68%**

Table 1: Comparison of results on ISCX-VPN2016, ISCX-Tor2016, QUIC2019 and CICIoT2022 datasets

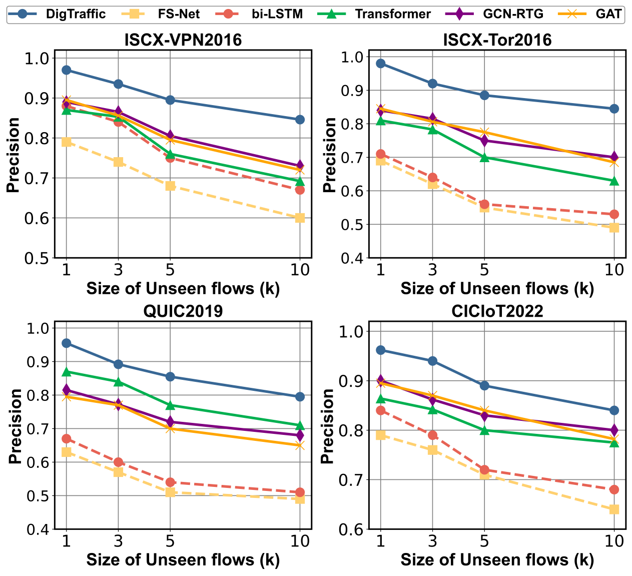

Figure 3: Comparison of the open-world scenario results on ISCX-VPN2016, ISCX-Tor2016, QUIC2019, and CICIoT2022 datasets.

becomes evident when dealing with lengthy traffic, such as QUIC flows in QUIC2019 dataset, where distant information is forgotten. In contrast, the strength of Transformer lies in its global receptive field, which captures long-range dependencies effectively. It indicates that Transformer achieves strong performance on the QUIC2019 dataset with the scale of length flows. However, it tends to fall short in integrating local latent space information, a gap that can be complemented by GNNs. Those GNNs (GCN-RTG and GAT) can address local features more effectively, complementing Transformers for tasks requiring both global and local contexts. Benefiting from well-crafted graph-level traffic representation in MTIG, dual-channel modal encoder, and message-aware graph transformer, DigTraffic delicately integrates these strengths and achieves outstanding performance across all datasets. This suggests that our method is not only well-suited for traffic classification tasks but also exhibits strong robustness.

4.3 Open-World Evaluation

We evaluate the methods in a realistic open-world setting, where models need to discern the traffic not occurred in the training data. Focusing on a binary classification task, we aim to classify each flow as seen or unseen. We compare three well-performing DL-based methods and two GNNs with our DigTraffic, using *Precision* to evaluate classification performance depicted in Fig. 3. Overall, DigTraffic accurately identifies known traffic and remains robust when handling unseen traffic in open-world settings. Additionally, GNN and Transformer performs relatively well by leveraging latent traffic features to mitigate data drift.

4.4 Computational Complexity Analysis

To analyze DigTraffic's time complexity, each Dual-channel Modal Transformer encoder has an input sequence length L and dimension d_{model}, resulting in a complexity of $O(L^2 \cdot d_{model})$ due to self-attention. It remains by using parallel Transformers. For Graph Transformer with N nodes, a fully connected graph consists of edge numbers $E = O(N^2)$, leading to a graph attention complexity of $O(N^2 \cdot d_{model})$. Since $L = N$ because spatial encoding serves as adjacent matrix, our model's total complexity is $O(N^2 \cdot d_{model})$. In practice, we enhance model's inference efficiency by 54% and 20% through lightweight parameters (detailed in Section 4.1) and parallel computation, respectively.

4.5 Ablation Study

To further assess the level of contribution where each component in DigTraffic bring to the overall performance, we conduct an ablation study using four baseline datasets. The evaluation results are illustrated in Table 2. Initially, Our method adopts average pooling rather than max pooling to enhance the capture of global relationships via a message-aware graph Transformer. After that, we comprehensively analyze all three types of edge relations in MTIG by controlling for variables related to each element. In general,

Method	ISCX-VPN2016		ISCX-Tor2016		QUIC2019		CICIoT2022	
	Acc.	F_1	Acc.	F_1	Acc.	F_1	Acc.	F_1
Ours (DigTraffic)	**98.14%**	**97.96%**	**98.57%**	**98.35%**	**96.41%**	**96.26%**	**96.91%**	**96.68%**
Ours with MP	97.50%	97.46%	97.31%	97.29%	96.95%	96.94%	96.33%	96.29%
Ours w/o 2UE	95.81%	95.77%	93.65%	93.47%	91.72%	90.88%	93.19%	93.10%
Ours w/o Het(CDE,2UE)	93.30%	93.19%	91.06%	90.94%	92.60%	92.51%	91.53%	91.36%
Ours w/o Het(SUE,2UE)	97.77%	97.72%	96.92%	96.49%	96.53%	96.48%	95.82%	95.78%
Ours w/o Het(SUE,CDE)	90.56%	89.82%	88.39%	88.20%	90.94%	90.50%	91.08%	90.58%
Ours with HoE	89.48%	88.60%	87.64%	87.33%	90.36%	89.40%	91.86%	91.01%
Ours w/o LSE	91.57%	91.71%	88.25%	87.90%	89.90%	89.66%	94.06%	93.73%
Ours w/o TSE	94.50%	94.24%	93.17%	93.00%	95.83%	95.82%	96.39%	96.37%
Ours w/o LSE & TSE	90.36%	88.52%	89.30%	89.18%	89.55%	89.46%	90.11%	89.80%
Ours w/o CE	90.29%	89.75%	88.83%	88.62%	89.50%	89.42%	91.49%	91.28%
Ours w/o SE	91.57%	91.44%	86.45%	86.35%	93.98%	93.98%	90.91%	90.88%
Ours w/o 2E	97.10%	96.89%	95.04%	94.95%	95.88%	95.87%	95.19%	94.78%
Ours w/o 2E & CE	89.74%	88.97%	87.86%	87.95%	89.13%	88.86%	89.33%	89.28%
Ours w/o SE & 2E & CE	87.90%	87.81%	84.46%	82.93%	94.04%	93.96%	88.56%	88.20%

Table 2: Ablation study of key components in DigTraffic on ISCX-VPN2016, ISCX-Tor2016, QUIC2019, CICIoT2022 datasets. The abbreviations are explained as follows, MP: max pooling, SUE / CDE / 2UE: successive uni-direction / changed direction / secondary uni-direction edges, Het(E,E): heterogeneous types of Edges, HoE: homogeneous types of edges, LSE / TSE: packet length / temporal sequence encoding, CE / SE / 2E: centrality / spatial / edge encoding.

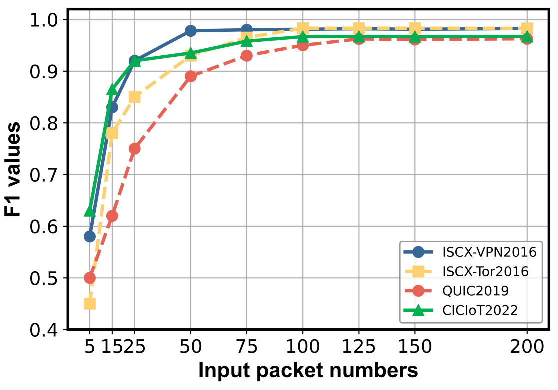

Figure 4: DigTraffic's performance with varying packet numbers across four datasets.

all three types of edges play a crucial role in representing inter-packet message interactions from different perspectives, with performance significantly decreasing when all edges are homogenized. They individually represent the intra-message, inter-message, and high-order neighboring interactions. Subsequently, the modality of packet length sequence proved more stable and informative than the temporal sequence because network transmissions may face varying delays from state fluctuations. Furthermore, whether through edge spatial connections or node distance embeddings, all three encoding blocks effectively contribute to guiding the attention mechanism within the Transformer. Therefore, omitting any of these designed modules leads to a significant decline in classification performance.

4.6 Impact of Traffic Length on DigTraffic's Performance

As illustrated in Fig. 4, The performance across all four datasets shows that the classification accuracy gradually converges as the number of input packets increases. In specific, the optimal number of input packets that DigTraffic requires to achieve the best flow classification performance depends on the traffic characteristics of the dataset being utilized. Intuitively, flows with shorter distributions require fewer packets. We find that inputting a traffic sequence of 50 packets into the model is sufficient to achieve an accuracy of over 90%. Furthermore, the results indicate that our DigTraffic is practical, since it requires only a few packets to classify traffic accurately across all datasets.

5 Conclusion

In this paper, we presented DigTraffic, a dual-channel interactive graph Transformer with message-aware flow representation for traffic classification. DigTraffic constructs traffic interaction graphs by modeling message interactions to learn intrinsic flow representation. We then employ dual-channel encoders to extract and fuse features from lengthy and temporal flow sequences into nodes. Moreover, our Graph Transformer integrates node embeddings with edge spatial relations, effectively capturing latent graph-level connections and detailed structural information. Evaluated on four real-world traffic datasets, DigTraffic consistently outperforms state-of-the-art methods, achieving superior results across all datasets.

Acknowledgments

We are grateful to anonymous reviewers for their constructive comments on this work. This work is supported by the Open Bidding for Selecting the Best Candidates of Jiangsu Province (Grant No.BE2023004-3), and the Joint Funds of the National Natural Science Foundation of China (Grant No.U22B2025).

References

Chen, Z.; Cheng, G.; Xu, Z.; Guo, S.; Zhou, Y.; and Zhao, Y. 2022. Length matters: Scalable fast encrypted internet traffic service classification based on multiple protocol data unit length sequence with composite deep learning. *Digital Communications and Networks*, 8(3): 289–302.

Cui, S.; Dong, C.; Shen, M.; Liu, Y.; Jiang, B.-S.; and Lu, Z. 2023. CBSeq: A Channel-Level Behavior Sequence for Encrypted Malware Traffic Detection. *IEEE Transactions on Information Forensics and Security*, 18: 5011–5025.

Dadkhah, S.; Mahdikhani, H.; Danso, P. K.; Zohourian, A.; Truong, K. A.; and Ghorbani, A. A. 2022. Towards the development of a realistic multidimensional IoT profiling dataset. In *2022 19th Annual International Conference on Privacy, Security & Trust (PST)*, 1–11. IEEE.

Draper-Gil, G.; Lashkari, A. H.; Mamun, M. S. I.; and Ghorbani, A. A. 2016. Characterization of encrypted and vpn traffic using time-related. In *Proceedings of the 2nd international conference on information systems security and privacy (ICISSP)*, 407–414.

Fu, C.; Li, Q.; and Xu, K. 2023. Detecting Unknown Encrypted Malicious Traffic in Real Time via Flow Interaction Graph Analysis. *ArXiv*, abs/2301.13686.

Fu, N.; Cheng, G.; and Su, X. 2023. Accurate compressed traffic detection via traffic analysis using Graph Convolutional Network based on graph structure feature. *Computer Communications*, 207: 128–139.

Lashkari, A. H.; Gil, G. D.; Mamun, M. S. I.; and Ghorbani, A. A. 2017. Characterization of tor traffic using time based features. In *International Conference on Information Systems Security and Privacy*, volume 2, 253–262. SciTePress.

Liu, C.; He, L.; Xiong, G.; Cao, Z.; and Li, Z. 2019. FS-Net: A Flow Sequence Network For Encrypted Traffic Classification. *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, 1171–1179.

Liu, C.; Xiong, G.; Gou, G.; ming Yiu, S.; Li, Z.; and Tian, Z. 2021. Classifying encrypted traffic using adaptive fingerprints with multi-level attributes. *World Wide Web*, 24: 2071 – 2097.

Liu, X.; You, X.; Zhang, X.; Wu, J.; and Lv, P. 2020. Tensor Graph Convolutional Networks for Text Classification. In *AAAI Conference on Artificial Intelligence*.

Lotfollahi, M.; Jafari Siavoshani, M.; Shirali Hossein Zade, R.; and Saberian, M. 2020. Deep packet: A novel approach for encrypted traffic classification using deep learning. *Soft Computing*, 24(3): 1999–2012.

Panchenko, A.; Lanze, F.; Pennekamp, J.; Engel, T.; Zinnen, A.; Henze, M.; and Wehrle, K. 2016. Website Fingerprinting at Internet Scale. In *Network and Distributed System Security Symposium*.

Rezaei, S.; and Liu, X. 2018. How to achieve high classification accuracy with just a few labels: A semi-supervised approach using sampled packets. *arXiv preprint arXiv:1812.09761*.

Shen, M.; Gao, Z.; Zhu, L.; and Xu, K. 2021a. Efficient Fine-Grained Website Fingerprinting via Encrypted Traffic Analysis with Deep Learning. *2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS)*, 1–10.

Shen, M.; Wei, M.; Zhu, L.; and Wang, M. 2017. Classification of Encrypted Traffic With Second-Order Markov Chains and Application Attribute Bigrams. *IEEE Transactions on Information Forensics and Security*, 12: 1830–1843.

Shen, M.; Zhang, J.; Zhu, L.; Xu, K.; and Du, X. 2021b. Accurate Decentralized Application Identification via Encrypted Traffic Analysis Using Graph Neural Networks. *IEEE Transactions on Information Forensics and Security*, 16: 2367–2380.

Su, X.; Hu, P.; You, Z.-H.; Philip, S. Y.; and Hu, L. 2024. Dual-Channel Learning Framework for Drug-Drug Interaction Prediction via Relation-Aware Heterogeneous Graph Transformer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 249–256.

Taylor, V. F.; Spolaor, R.; Conti, M.; and Martinovic, I. 2016. Appscanner: Automatic fingerprinting of smartphone apps from encrypted network traffic. In *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, 439–454. IEEE.

Vaswani, A.; Shazeer, N. M.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is All you Need. In *Neural Information Processing Systems*.

Wang, L.; Huang, Y.; Hou, Y.; Zhang, S.; and Shan, J. 2019a. Graph Attention Convolution for Point Cloud Semantic Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wang, X.; Ji, H.; Shi, C.; Wang, B.; Ye, Y.; Cui, P.; and Yu, P. S. 2019b. Heterogeneous graph attention network. In *The world wide web conference*, 2022–2032.

Ying, C.; Cai, T.; Luo, S.; Zheng, S.; Ke, G.; He, D.; Shen, Y.; and Liu, T.-Y. 2021. Do Transformers Really Perform Badly for Graph Representation? In *Advances in Neural Information Processing Systems*.

Zhang, J.; Li, F.; Ye, F.; and Wu, H. 2020. Autonomous unknown-application filtering and labeling for dl-based traffic classifier update. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*, 397–405. IEEE.

Zhao, R.; Deng, X.; Wang, Y.; Chen, L.; Liu, M.; Xue, Z.; and Wang, Y. 2022a. Flow Sequence-Based Anonymity Network Traffic Identification with Residual Graph Convolutional Networks. *2022 IEEE/ACM 30th International Symposium on Quality of Service (IWQoS)*, 1–10.

Zhao, R.; Deng, X.; Yan, Z.; Ma, J.; Xue, Z.; and Wang, Y. 2022b. MT-FlowFormer: A Semi-Supervised Flow Transformer for Encrypted Traffic Classification. *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

Zhao, R.; Zhan, M.; Deng, X.; Li, F.; Wang, Y.; Wang, Y.; Gui, G.; and Xue, Z. 2024. A Novel Self-Supervised Framework Based on Masked Autoencoder for Traffic Classification. *IEEE/ACM Transactions on Networking*, 32: 2012–2025.

Zhao, R.; Zhan, M.; Deng, X.; Wang, Y.; Wang, Y.; Gui, G.; and Xue, Z. 2023. Yet another traffic classifier: A masked autoencoder based traffic transformer with multi-level flow representation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 5420–5427.

Zhou, Q.; Wang, L.; Zhu, H.; Lu, T.; and Sheng, V. S. 2024. WF-Transformer: Learning Temporal Features for Accurate Anonymous Traffic Identification by Using Transformer Networks. *IEEE Transactions on Information Forensics and Security*, 19: 30–43.