TrafficGraphMAE: Self-Supervised Masked Graph Autoencoders for Encrypted Traffic Classification

Zhixin Meng[1,2], Mengyan Liu[1,2 *], Junzheng Shi[1,2], Gang Xiong[1,2], Zhen Li[1,2], Gaopeng Gou[1,2]
[1]Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2]School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
{mengzhixin, liumengyan, shijunzheng, xionggang, lizhen, gougaopeng}@iie.ac.cn

***Abstract*—Encrypted traffic classification plays an essential role in network management and security. However, the widespread adoption of payload encryption poses significant challenges for traffic analysis. Existing approaches either model traffic as sequential data, ignoring structural relationships among packets, or construct graph-based representations that model temporal and structural information separately, limiting their representational capacity. In this paper, we propose TrafficGraphMAE, a self-supervised masked graph autoencoder for encrypted traffic classification. We first introduce the Temporal-Traffic Interaction Graph (TTIG), a novel graph construction method that jointly models packet-level temporal dependencies and burst-level structural interactions within a unified framework. Based on TTIG, TrafficGraphMAE employs a masked graph autoencoder with multi-task learning objectives, including node feature reconstruction and structural relationship prediction, to learn expressive traffic representations in a self-supervised manner. Experiments on four public datasets demonstrate that TrafficGraphMAE outperforms multiple existing methods.**

***Index Terms*—Encrypted Traffic Classification, Masked Graph Autoencoders, Self-Supervised Learning**

I. Introduction

Identifying network traffic types is a fundamental and indispensable capability in network traffic analysis. Accurate traffic classification is of great importance for various applications, including Quality of Service (QoS) control, malware detection, and intrusion detection [1]. Recently, to protect user privacy and anonymity, various encryption techniques have been extensively deployed in network communications [2]. Although encryption effectively protects sensitive information, it simultaneously obscures packet contents, thereby substantially increasing the difficulty of traffic analysis. Moreover, privacy-enhancing technologies such as VPNs and Tor are increasingly leveraged to bypass Internet censorship mechanisms [3], further exacerbating the complexity of encrypted traffic classification. Consequently, extracting discriminative information from encrypted traffic to enable accurate classification has become a crucial yet highly challenging task.

Early approaches for encrypted traffic classification primarily relied on rule-based techniques such as Deep Packet Inspection (DPI) [4] and port-based identification [5], which are largely ineffective in modern network environments due to payload encryption and dynamic port usage. With the advancement of machine learning and deep learning, encrypted traffic classification has shifted to data-driven modeling based on traffic statistics or raw traffic sequences. Traditional machine learning methods typically extract handcrafted features and employ classical classifiers for traffic identification [6], [7]. Nevertheless, these methods are highly dependent on domain expertise, which limits their scalability and generalizability. In contrast, deep learning approaches, benefiting from end-to-end representation learning, have gradually become dominant. Many studies leverage Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) to capture spatial and temporal patterns within traffic data [8]–[10]. However, such approaches implicitly assume that traffic data are located in Euclidean space, making them inherently inadequate for modeling the complex non-Euclidean dependencies inherent in network traffic.

Graph Neural Networks (GNNs) have recently emerged as a promising paradigm for traffic classification due to their ability to explicitly model inter-packet relationships. Existing graph-based approaches typically construct packet-level graphs and apply GNNs for representation learning [11]–[16]. However, two key limitations remain in current methods. (1) Separate modeling of temporal and structural information. Existing approaches either construct temporal graphs that only capture sequential dependencies or structural graphs that focus on packet-level interactions while ignoring temporal information [12], [13]. This separation results in incomplete representations that fail to adequately characterize traffic behaviors. (2) Single-objective self-supervised learning. Recent works have introduced masked autoencoders for traffic representation learning [18], [19]. However, they typically rely on a single reconstruction objective, which provides limited supervision signals for capturing complex structural patterns.

To address these limitations, we propose TrafficGraphMAE, a self-supervised masked graph autoencoder based on a novel graph construction method called the Temporal-Traffic Interaction Graph (TTIG). Our core observation is that encrypted traffic exhibits an inherent hierarchical structure. At the packet level, consecutive packets form temporal sequences, while at the burst level, direction-consistent packet groups reflect application-layer request-response behaviors. By jointly modeling both levels within a unified graph, TTIG provides a more expressive representation. Built upon TTIG, TrafficGraphMAE employs a masked graph autoencoder with multi-task learning

* Corresponding author

objectives that jointly optimizes node feature reconstruction and structural relationship prediction to learn both content-aware representations and topological dependencies.

The main contributions of this paper are as follows.

- We propose the TTIG, a novel graph construction method that jointly captures packet-level temporal dependencies and burst-level structural interactions within flows.
- We introduce TrafficGraphMAE, which learns traffic representations by jointly optimizing node feature reconstruction and structural relationship prediction.
- Extensive experiments on four public datasets demonstrate that the proposed method consistently outperforms multiple existing approaches.

II. Related Work

A. Encrypted Traffic Classification Methods

Statistical Feature–Based Methods. These methods predominantly leverage handcrafted statistical features, employing classical machine learning models for traffic classification. Taylor et al. [6] introduced AppScanner, a classification model that uses the random forest to distinguish mobile apps. Ede et al. [7] proposed FlowPrint, which combines destination-based clustering, browser isolation, and pattern recognition to perform application fingerprinting. However, these models typically necessitate manual feature selection and face challenges due to feature instability.

Deep Learning–Based Methods. These methods adopt Convolutional Neural Networks (CNNs) or Recurrent Neural Networks (RNNs) to model encrypted traffic data. Sirinam et al. [9] proposed DF model, which adopts a multi-layer CNN architecture to incorporate the spatial correlations of encrypted traffic. Liu et al. [10] introduced FS-Net, which takes the packet length sequences as input, and employs bidirectional GRU for feature encoding. However, such methods cannot model the non-Euclidean structures within network flows.

Graph-Based Methods. Graph-based approaches construct packet-level or flow-level graphs to model the inter-packet relationships. Shen et al. [12] proposed GraphDApp, which constructs the Traffic Interaction Graph based on the packet length and direction, and turned decentralized applications identification into a graph classification problem. Huoh et al. [13] proposed a graph construction method, which directly creates edges based on the chronological relationship of packets within a flow and leverages the geometric learning model to learn the node, edge and global features. Zhang et al. [14] proposed the TFE-GNN that processes packet headers and payloads separately, and then concatenates their embeddings for enhanced classification. Nevertheless, existing methods typically model temporal dependencies or structural relationships separately, neglecting their joint modeling.

B. Self-Supervised Learning and Masked Graph Models

Self-supervised learning aims to learn general representations from unlabeled data through carefully designed pretext tasks. Masked learning, which involves reconstructing randomly masked inputs to exploit contextual dependencies and improve generalization, has emerged as a prominent paradigm. In the graph learning domain, GraphMAE [17] applies feature masking and reconstruction to learn graph representations and achieves strong performance on downstream tasks. Recently, masked self-supervised learning has also been introduced to network traffic analysis. Qing et al. [18] used masked autoencoders to reconstruct packet length sequences for anomaly detection, while Zhao et al. [19] employed masking-based self-supervised pretraining to improve generalization in traffic classification. However, most existing approaches rely on a single reconstruction objective, which is insufficient to capture the deeper structural relationships inherent in traffic data.

III. Methodology

A. Framework Overview

As illustrated in Fig. 1, the training of TrafficGraphMAE consists of three stages: Temporal-Traffic Interaction Graph Construction, Self-Supervised Pretraining, and Supervised Fine-tuning.

First, raw traffic flows are modeled as TTIGs that capture packet-level temporal dependencies and burst-level structural interactions. Next, in the self-supervised pretraining stage, a proportion of the node features in each TTIG are masked and fed into the encoder to learn effective representations. Based on the representations, the decoder performs two reconstruction tasks: Node Feature Reconstruction and Structural Relationship Prediction. Finally, in the supervised fine-tuning stage, a co-teaching strategy is applied to further improve robustness under complex scenarios by training two parallel classifiers simultaneously.

B. Temporal-Traffic Interaction Graph Construction

Encrypted traffic inherently exhibits both packet-level temporal dependencies, which capture fine-grained transmission dynamics, and burst-level structural interactions, which reflect application-layer request–response behaviors. However, existing methods often model these two aspects separately. To address this limitation and jointly model temporal dependencies and structural interactions, we propose the Temporal-Traffic Interaction Graph (TTIG), a unified and expressive graph representation for encrypted traffic analysis.

Given a flow $F = \{p_i\}_{i=1}^{n}$, each packet p_i is modeled as a node in the graph. The set of nodes is defined as $V = \{v_i \mid p_i \in F\}$. The initial feature of each node v_i is defined as $x_i = d_i \cdot len_i$, where $d_i \in \{+1, -1\}$ denotes the direction of packet transmission ($+1$ for client-to-server and -1 for server-to-client), and len_i represents the packet length. Collectively, all node features form the feature matrix $X = \{x_i\}_{i=1}^{n}$. To further characterize packet behaviors within each flow, the concept of burst, a sequence of consecutive packets transmitted in the same direction, is introduced. The set of bursts is denoted as $\mathcal{B} = \{B_1, B_2, \ldots, B_K\}$.

To enhance the ability of TTIG to capture packet-level temporal dependencies and burst-level structural interactions within network flows simultaneously, two types of edges are introduced during graph construction, as defined below.

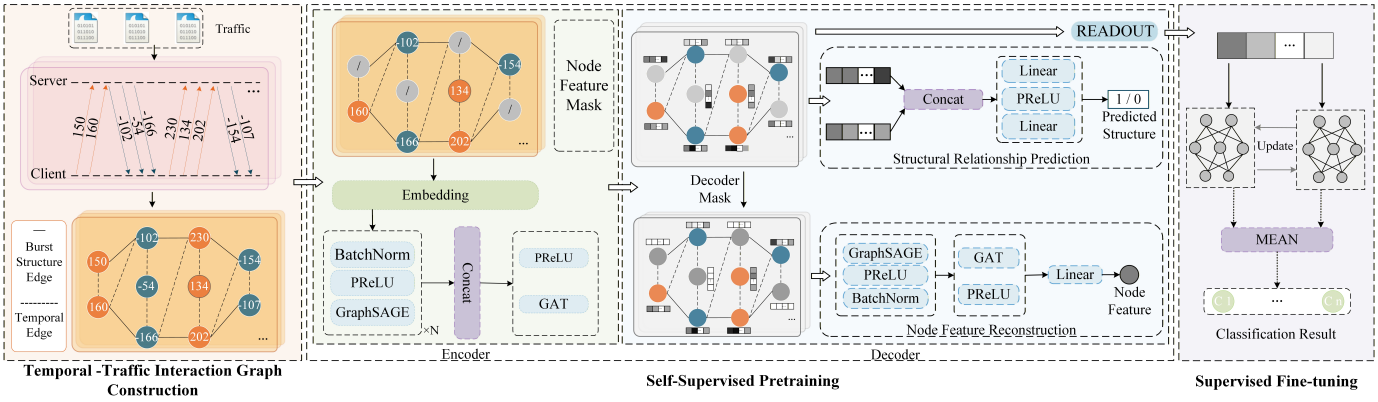

Figure 1 The framework of the proposed TrafficGraphMAE.

Algorithm 1 TTIG Construction

Input: Flow $F = \{p_i\}_{i=1}^{n}$ with direction d_i and length len_i
Output: Graph $G = (V, E_t \cup E_b, X)$
for $i = 1$ to n **do**
  $V \leftarrow V \cup \{v_i\}$, $X \leftarrow X \cup \{x_i\}$, $x_i \leftarrow d_i \cdot len_i$
  if $i < n$ **then**
    $E_t \leftarrow E_t \cup \{(v_i, v_{i+1})\}$
  end if
end for
Partition packets into bursts $\mathcal{B}$
for $k = 1$ to $|\mathcal{B}| - 1$ **do**
  $E_b \leftarrow E_b \cup \{(v_{s_k}, v_{s_{k+1}}), (v_{e_k}, v_{e_{k+1}})\}$
end for
return $G = (V, E_t \cup E_b, X)$

Temporal edges: $E_t = \{(v_i, v_{i+1})\}$, which capture the temporal sequential dependency between consecutive packets and retain the intrinsic order of packet transmission.

Burst structural edges: $E_b = \{(v_{s_k}, v_{s_{k+1}}), (v_{e_k}, v_{e_{k+1}}) \mid k = 1, \ldots, K-1\}$ where v_{s_k} and v_{e_k} denote the first and last packets of burst B_k, respectively. This type of edge can capture structural interactions between adjacent bursts and reflect application-level round-trip communication patterns. Furthermore, it is noteworthy that such edges exhibit robust stability even in the context of encrypted traffic scenarios.

Based on the above definitions, the TTIG can be constructed as $G = (V, E_t \cup E_b, X)$. By integrating temporal information and burst-level structural information, the TTIG provides more expressive representations than traditional sequence-based modeling approaches. The detailed pseudocode for TTIG construction is presented in Algorithm 1.

C. Self-Supervised Pretraining

The core objective of self-supervised pretraining is to learn effective traffic representations by reconstructing masked node features based on the topological structure of the TTIG. To this end, we propose TrafficGraphMAE, which adopts a masked graph autoencoder and leverages feature reconstruction as the primary training signal. In contrast to contrastive learning paradigms, masked autoencoders eliminate the necessity for the construction of positive-negative sample pairs or data augmentation, making them particularly suitable for encrypted traffic analysis, where effective semantic augmentation strategies are often unavailable.

The overall framework of TrafficGraphMAE consists of three components: Mask Mechanism, Encoder, and Decoder. By masking node features and imposing reconstruction-based objectives, the model is encouraged to learn structural dependencies and local semantic correlations within the TTIG, thereby generating more discriminative representations.

Mask Mechanism. Given a masking ratio $r \in (0, 1)$, the nodes randomly sampled from V with the ratio r form the masked node set M, and the remaining nodes constitute the unmasked set U. For nodes in the set M, their initial features are masked according to (1). This mask mechanism forces the model to rely on neighborhood features and graph topology to reconstruct missing information, thereby enhancing the encoder's ability to capture contextual structural patterns.

$$\tilde{x}_i = \begin{cases} \text{mask}_{encoder}, & i \in M \\ x_i, & i \notin M \end{cases} \tag{1}$$

Encoder. The encoder is designed to capture both local neighborhood information and global dependencies simultaneously. It is composed of multiple GraphSAGE [20] layers, followed by a single GAT [21] layer. Formally, in the l-th GraphSAGE layer, node representations are updated according to (2).

$$\begin{aligned} m_i^{(l)} &= \text{AGG}\big(\{h_j^{(l-1)} \mid j \in \mathcal{N}(i)\}\big) \\ h_i^{(l)} &= \sigma\big(W^{(l)}[h_i^{(l-1)} \,\|\, m_i^{(l)}]\big) \end{aligned} \tag{2}$$

where the aggregation function AGG employs mean aggregation to ensure the stability of the local structure model. After L GraphSAGE layers, the outputs from all layers are concatenated to form multi-scale node representations, $\tilde{h}_i = [h_i^{(1)} || h_i^{(2)} || \ldots || h_i^{(L)}]$. A single GAT layer is then applied to assign adaptive attention weights to neighboring nodes, enabling the modeling of non-uniform dependencies.

$$h_i^{enc} = \text{GAT}\big(\tilde{h}_i,\ \{\tilde{h}_j \mid j \in \mathcal{N}(i)\}\big) \tag{3}$$

where $\mathcal{N}(i)$ denotes the set of neighboring nodes of node v_i. The node-level representations generated by the encoder are denoted as $H^{enc} = \{h_i^{enc}\}$. These node-level representations are subsequently aggregated via a readout operation to obtain graph-level representations, $h_G = \text{ReadOut}(\{h_i^{\text{enc}} \mid v_i \in V\})$, which serve as input for downstream tasks.

Decoder and Self-Supervised Objectives. The decoder is designed to support multi-task self-supervised pretraining and consists of two functional modules: Node Feature Reconstruction and Structural Relationship Prediction. Consequently, the self-supervised objective of TrafficGraphMAE comprises three complementary tasks: masked node feature reconstruction, unmasked node feature reconstruction, and structural relationship prediction.

(1) **Node Feature Reconstruction**. Node feature reconstruction serves as the primary supervision signal during pretraining. To prevent information leakage, the representations corresponding to masked nodes are re-masked before being fed into the decoder according to (4).

$$g_i^{(0)} = \begin{cases} \text{mask}_{decoder}, & i \in M \\ h_i^{enc}, & i \notin M \end{cases} \tag{4}$$

Specifically, for unmasked nodes, the decoder inputs are directly taken from the encoder outputs; for masked nodes, the corresponding representations are masked again, ensuring that reconstruction relies solely on message passing rather than direct feature memorization.

The decoder adopts a stack of GraphSAGE, GAT, and MLP layers similar to the encoder. It takes $g_i^{(0)}$ as input and outputs reconstructed node features. Based on this design, two reconstruction tasks are jointly optimized.

The **masked node feature reconstruction** task aims to reconstruct the original features (i.e., packet length) of masked nodes, and the reconstruction error is measured using a cross-entropy loss. This task encourages the model to learn the neighborhood context within flows.

Additionally, the **unmasked node feature reconstruction** is introduced as an auxiliary task, also supervised by cross-entropy loss. This task is particularly critical at high mask ratios, as it alleviates gradient instability and prevents the model from over-relying on masked-node reconstruction signals.

(2) **Structural Relationship Prediction**. To explicitly capture topological dependencies within the TTIG, the decoder further performs a structural relationship prediction task. This task focuses exclusively on unmasked node pairs. Positive and negative samples are drawn from $S \subseteq U \times U$. Specifically, an MLP takes the concatenated encoded representations of each node pair (i.e., $e_{ij} = [h_i^{enc} || h_j^{enc}]$) as input and outputs the predicted edge existence probability $\hat{A}_{ij}$, indicating whether an edge exists between nodes v_i and v_j. This objective guides the model to capture burst-level interaction patterns between clients and servers, thereby enriching the learned semantic representations.

Overall Pretraining Objective. The final self-supervised pretraining objective is formulated as a weighted combination of the three aforementioned tasks. Specifically, the overall objective is defined as

$$\mathcal{L}_{pre} = \alpha \mathcal{L}_{mask} + \beta \mathcal{L}_{unmask} + \gamma \mathcal{L}_{struct} \tag{5}$$

where $\mathcal{L}_{mask}$, $\mathcal{L}_{unmask}$, and $\mathcal{L}_{struct}$ denote the masked node feature reconstruction loss, the unmasked node feature reconstruction loss, and the structural relationship prediction loss, respectively. The hyperparameter α, β, and γ balance the relative contributions of the different tasks during training, enabling the model to jointly learn content-aware representations and topological structural dependencies.

D. Supervised Fine-tuning

After self-supervised pretraining, the encoder parameters are frozen and used solely as a feature extractor to generate graph-level representations h_G. To further improve robustness and generalization under complex traffic scenarios, a co-teaching strategy is adopted during supervised fine-tuning. Specifically, two classifiers C_1 and C_2 with identical architectures are trained in parallel. For each batch, their predictions are generated as follows:

$$\hat{y}_1 = C_1(h_G), \quad \hat{y}_2 = C_2(h_G) \tag{6}$$

where $\hat{y}_1$ and $\hat{y}_2$ denote the predicted class distributions produced by C_1 and C_2, respectively. Each classifier then computes the cross-entropy loss for all samples in the batch and selects a subset of samples with small losses, denoted as

$$\mathcal{S}_1 = \text{SmallLoss}(C_1), \quad \mathcal{S}_2 = \text{SmallLoss}(C_2) \tag{7}$$

Subsequently, a cross-update mechanism is applied: classifier C_1 updates its parameters using the sample set $\mathcal{S}_2$, while classifier C_2 is updated using $\mathcal{S}_1$. This collaborative filtering and updating strategy effectively reduces the influence of noisy samples and enhances classification performance.

IV. Experiments

A. Experimental Setup

Datasets. In order to comprehensively evaluate the effectiveness of TrafficGraphMAE, we adopt multiple public datasets, including ISCX VPN-nonVPN [22] and ISCX Tor-nonTor [23].

The ISCX VPN-nonVPN dataset contains two subsets: the ISCX-VPN dataset and the ISCX-nonVPN dataset. The ISCX-VPN dataset is collected through VPNs, which are commonly used to access restricted services and are difficult to identify due to traffic obfuscation technology. In contrast, the ISCX-nonVPN dataset is transmitted without VPNs and exhibits more conventional communication patterns. The detailed label distribution is reported in Table II.

Similarly, the ISCX Tor-nonTor dataset contains two subsets: the ISCX-Tor dataset and the ISCX-nonTor dataset. The ISCX-Tor dataset is collected over the onion router (Tor), which is inherently difficult to trace, whereas the ISCX-nonTor dataset contains regular traffic and is not collected over Tor. The corresponding label distribution is reported in Table II.

Table I Performance Comparison on ISCX Datasets

Model	ISCX-VPN				ISCX-nonVPN				ISCX-Tor				ISCX-nonTor			
	Acc	Pre	Re	F1	Acc	Pre	Re	F1	Acc	Pre	Re	F1	Acc	Pre	Re	F1
AppScanner	0.8889	0.8679	0.8815	0.8722	0.7399	0.7426	0.7351	0.7357	0.7543	0.6629	0.6042	0.6163	0.9153	0.8435	0.8140	0.8273
FlowPrint	0.8795	0.8965	0.8935	0.8883	0.6436	0.6126	0.5898	0.5952	0.2400	0.0300	0.1250	0.0484	0.8885	0.7472	0.7536	0.7485
DF	0.8133	0.8072	0.8270	0.8112	0.6561	0.6890	0.6071	0.6321	0.6514	0.4803	0.4767	0.4719	0.8568	0.8003	0.7415	0.7590
FS-Net	0.9298	0.9263	0.9211	0.9234	0.7388	0.7504	0.7613	0.7486	0.8685	0.7542	0.7689	0.7512	0.9278	0.8368	0.8254	0.8285
GraphDApp	0.6491	0.5668	0.6103	0.5740	0.4495	0.4230	0.3647	0.3614	0.4286	0.2557	0.2509	0.2281	0.6936	0.5447	0.5398	0.5352
FB-GNN	0.8431	0.8293	0.8141	0.8184	0.7618	0.7641	0.7667	0.7637	0.4286	0.2381	0.3136	0.2581	0.8420	0.5983	0.6842	0.6250
TFE-GNN	0.9267	0.9145	0.9065	0.9094	0.8399	0.8235	0.8011	0.8111	**0.9886**	**0.9792**	**0.9939**	**0.9855**	0.9450	0.8171	0.7450	0.7690
TrafficGraphMAE	**0.9432**	**0.9428**	**0.9417**	**0.9413**	**0.9136**	**0.9085**	**0.9107**	**0.9084**	0.9722	0.9667	0.9665	0.9663	**0.9509**	**0.8870**	**0.8483**	**0.8656**

Table II Traffic Types in ISCX Datasets

Dataset	Traffic Type
ISCX-VPN	Chat, Email, File, P2P, Streaming, VoIP
ISCX-nonVPN	Chat, Email, File, Streaming, Video, VoIP
ISCX-Tor	Audio, Browsing, Chat, File, Mail, P2P, Video, VoIP
ISCX-nonTor	Audio, Browsing, Chat, Email, FTP, P2P, Video, VoIP

Evaluation Metrics and Baselines. To provide a fair comparison, we use four evaluation metrics, i.e., Overall Accuracy (Acc), Precision (Pre), Recall (Re), and Macro F1-score to evaluate TrafficGraphMAE with the following baselines, including **Statistical Feature–Based Methods** (i.e., AppScanner [6] and FlowPrint [7]), **Deep Learning–Based Methods** (i.e., DF [9] and FS-Net [10]), and **Graph-Based Methods** (i.e., GraphDApp [12], FB-GNN [13], and TFE-GNN [14]).

Implementation Details. In the self-supervised pretraining stage, the node masking ratio r is set to 0.5, and the loss weights α, β, and γ are set to 0.6, 0.2, and 0.2, respectively. For each dataset, we applied stratified sampling to split the data into training and testing sets with a ratio of 8:2. Each experiment is independently repeated five times, and the average results are reported.

B. Comparative Experiments

Comparative results on the ISCX VPN-nonVPN and ISCX Tor-nonTor datasets are reported in Table I. The following observations can be drawn. (1) TrafficGraphMAE achieves the best performance on three of the four datasets. Specifically, on the ISCX-VPN, ISCX-nonVPN, and ISCX-nonTor datasets, TrafficGraphMAE obtains the highest scores across all evaluation metrics. Notably, on the ISCX-nonVPN dataset, TrafficGraphMAE achieves an F1-score of 0.9084, outperforming TFE-GNN by 9.7%. (2) On the ISCX-Tor dataset, TFE-GNN slightly outperforms TrafficGraphMAE (F1: 0.9855 vs. 0.9663). This can be attributed to the utilization of the packet header and payload features in TFE-GNN, which capture Tor-specific protocol patterns. In contrast, TrafficGraphMAE relies solely on packet length and direction information without requiring payload inspection. (3) TrafficGraphMAE demonstrates stronger generalization across different scenarios. It achieves the highest average F1-score (0.9204) across all datasets with lower variance, while TFE-GNN exhibits performance degradation on the ISCX-nonVPN dataset (F1: 0.8111). (4) Compared to existing graph-based approaches, including GraphDApp and FB-GNN, TrafficGraphMAE achieves substantial performance improvements, validating the effectiveness of the proposed TTIG construction and multi-task self-supervised pretraining.

C. Ablation Study

The ablation study is conducted on the ISCX VPN-nonVPN dataset to evaluate the contribution of each component in TrafficGraphMAE. The evaluated variants include removing node mask (w/o node mask), removing decoder mask (w/o decoder mask), removing the unmasked node feature reconstruction task (w/o $\mathcal{L}_{\mathrm{unmask}}$), removing the structural relationship prediction task (w/o $\mathcal{L}_{\mathrm{struct}}$), and a baseline model retaining only the masked node reconstruction task (w/ $\mathcal{L}_{\mathrm{mask}}$). The results are reported in Table III.

Table III Ablation Study on ISCX-VPN and ISCX-nonVPN

Method	Acc	Pre	Re	F1
w/o node mask	0.9121 | 0.8739	0.9109 | 0.8623	0.9120 | 0.8644	0.9094 | 0.8611
w/o decoder mask	0.9147 | 0.8824	0.9200 | 0.8694	0.9136 | 0.8697	0.9147 | 0.8679
w/o L_{unmask}	0.8941 | 0.8896	0.8957 | 0.8803	0.8864 | 0.8801	0.8901 | 0.8786
w/o L_{struct}	0.9121 | 0.8860	0.9132 | 0.8756	0.9083 | 0.8744	0.9095 | 0.8728
w/ L_{mask}	0.8915 | 0.8932	0.8886 | 0.8806	0.8840 | 0.8875	0.8851 | 0.8826
TrafficGraphMAE	**0.9432 | 0.9136**	**0.9428 | 0.9085**	**0.9417 | 0.9107**	**0.9413 | 0.9084**

The following observations can be obtained. (1) Node Mask plays a core component of self-supervised learning. After integrating the Node Mask mechanism, the model's F1-score improves by approximately 3.2% and 5.2%, respectively, which indicates that Node Mask Mechanism facilitates the model in learning deeper relational patterns within traffic graphs. (2) Decoder Mask effectively mitigates information leakage. With the introduction of Decoder Mask, the model's F1-score improves by approximately 2.8% and 4.5%, respectively, suggesting that re-masking node representations prevents trivial reconstruction and encourages structural learning. (3) Effect of auxiliary tasks. On one hand, adding the unmasked node feature reconstruction task leads to F1-score improvements of approximately 5.4% and 3.3%, demonstrating that this task provides supplementary supervision signals and improves the stability of model training. On the other hand, incorporating the structural relationship prediction task further increases the F1-score by approximately 3.4% and 3.9%, which verifies that this task strengthens the model's ability to capture traffic topological structures. (4) Multi-task joint training plays an important role. When only the masked node reconstruction task is retained, the F1-score decreases to 0.8851 and 0.8826,

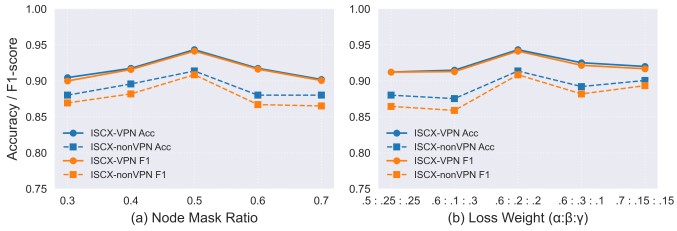

Figure 2 Parameter Analysis on ISCX-VPN and ISCX-nonVPN.

which indicates that the single reconstruction objective is insufficient to fully capture structural characteristics, whereas multi-task joint training provides complementary supervision information.

D. Parameter Analysis

In this section, we further investigate the impact of parameters in self-supervised pretraining on model performance. Two parameters-the node mask ratio and the multi-task loss weights-are selected for analysis, and experiments are conducted on the ISCX VPN-nonVPN dataset. The results are shown in Fig. 2. (1) Effect of node mask ratio. When the mask ratio is low, the model relies on explicit information for feature reconstruction but struggles to learn deep behavioral features. In contrast, a high mask ratio leads to insufficient effective information, which causes the training to become unstable. (2) Effect of multi-task loss weights. A rational allocation of task weights is crucial for learning optimal representations. A high weight for the structural relationship prediction task may cause the model to ignore local behavioral features, whereas overemphasizing the unmasked node feature reconstruction weakens its ability to model structural dependencies. The model achieves the best performance under a balanced weight configuration, demonstrating the effectiveness and stability of the proposed joint optimization strategy.

V. Conclusion

In this paper, we propose TrafficGraphMAE, a self-supervised masked graph autoencoder for encrypted traffic classification. Specifically, we introduce the Temporal-Traffic Interaction Graph (TTIG), a novel graph construction approach that jointly models packet-level temporal dependencies and burst-level structural interactions within a unified framework. Based on TTIG, TrafficGraphMAE adopts multi-task learning objectives, including node feature reconstruction and structural relationship prediction, to learn effective traffic representations without labeled data. Experimental results on four public datasets demonstrate that the proposed approach consistently outperforms multiple existing methods. Despite these results, the current framework is limited to static graph construction within individual flows and does not explicitly model dynamic or cross-flow interactions. Extending TrafficGraphMAE to incorporate such interactions is a promising direction for future research.

References

[1] W. Dong, J. Yu, X. Lin, *et al.*, Deep learning and pre-training technology for encrypted traffic classification: A comprehensive review, *Neurocomputing*, vol. 617, Art. no. 128444, 2025.

[2] A. H. Abdi, L. Audah, A. Salh, *et al.*, Security control and data planes of SDN: A comprehensive review of traditional, AI, and MTD approaches to security solutions, *IEEE Access*, vol. 12, pp. 69941–69980, 2024.

[3] E. Ramadhani, Anonymity communication VPN and Tor: A comparative study, in *J. Phys.: Conf. Ser.*, vol. 983, IOP Publishing, 2018, Art. no. 012060.

[4] J. Su, S. Chen, B. Han, C. Xu, and X. Wang, A 60 Gbps DPI prototype based on memory-centric FPGA, in *Proc. ACM SIGCOMM*, 2016, pp. 627–628.

[5] T. Nelms, R. Perdisci, and M. Ahamad, ExecScent: Mining for new C&C domains in live networks with adaptive control protocol templates, in *Proc. 22nd USENIX Security Symp. (USENIX Security)*, 2013, pp. 589–604.

[6] V. F. Taylor, R. Spolaor, M. Conti, *et al.*, AppScanner: Automatic fingerprinting of smartphone apps from encrypted network traffic, in *Proc. IEEE Eur. Symp. Security and Privacy (EuroS&P)*, 2016, pp. 439–454.

[7] T. Van Ede, R. Bortolameotti, A. Continella, *et al.*, FlowPrint: Semi-supervised mobile-app fingerprinting on encrypted network traffic, in *Proc. Network and Distributed System Security Symp. (NDSS)*, 2020.

[8] E. Papadogiannaki and S. Ioannidis, A survey on encrypted network traffic analysis applications, techniques, and countermeasures, *ACM Comput. Surv.*, vol. 54, no. 6, pp. 1–35, 2021.

[9] P. Sirinam, M. Imani, M. Juarez, *et al.*, Deep fingerprinting: Undermining website fingerprinting defenses with deep learning, in *Proc. ACM SIGSAC Conf. Computer and Communications Security (CCS)*, 2018, pp. 1928–1943.

[10] C. Liu, L. He, G. Xiong, *et al.*, FS-Net: A flow sequence network for encrypted traffic classification, in *Proc. IEEE INFOCOM*, 2019, pp. 1171–1179.

[11] Y. Han and H. Dai, Research on network traffic classification based on graph neural network, *IAENG Int. J. Comput. Sci.*, vol. 51, no. 12, 2024.

[12] M. Shen, J. Zhang, L. Zhu, *et al.*, Accurate decentralized application identification via encrypted traffic analysis using graph neural networks, *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 2367–2380, 2021.

[13] T. L. Huoh, Y. Luo, P. Li, *et al.*, Flow-based encrypted network traffic classification with graph neural networks, *IEEE Trans. Netw. Service Manag.*, vol. 20, no. 2, pp. 1224–1237, 2022.

[14] H. Zhang, L. Yu, X. Xiao, *et al.*, TFE-GNN: A temporal fusion encoder using graph neural networks for fine-grained encrypted traffic classification, in *Proc. ACM Web Conf.*, 2023, pp. 2066–2075.

[15] Z. W. Chen, X. X. Wei, and Y. S. Wang, Encrypted traffic classification encoder based on lightweight graph representation, *Sci. Rep.*, vol. 15, no. 1, Art. no. 28564, 2025.

[16] H. Zhang, H. Yue, X. Xiao, *et al.*, Revolutionizing encrypted traffic classification with MH-Net: A multi-view heterogeneous graph model, in *Proc. AAAI Conf. Artificial Intelligence*, vol. 39, no. 1, 2025, pp. 1048–1056.

[17] Z. Hou, X. Liu, Y. Cen, *et al.*, GraphMAE: Self-supervised masked graph autoencoders, in *Proc. ACM SIGKDD*, 2022, pp. 594–604.

[18] Y. Qing, Q. Yin, X. Deng, *et al.*, Low-quality training data only? A robust framework for detecting encrypted malicious network traffic, in *Proc. Network and Distributed System Security Symposium (NDSS)*, 2024.

[19] R. Zhao, M. Zhan, X. Deng, *et al.*, A novel self-supervised framework based on masked autoencoder for traffic classification, *IEEE/ACM Trans. Netw.*, vol. 32, no. 3, pp. 2012–2025, 2024.

[20] W. Hamilton, Z. Ying, and J. Leskovec, Inductive representation learning on large graphs, in *Adv. Neural Inf. Process. Syst.*, vol. 30, 2017.

[21] P. Veličković, G. Cucurull, A. Casanova, *et al.*, Graph attention networks, in *Proc. Int. Conf. Learning Representations (ICLR)*, 2018.

[22] G. Draper-Gil, A. H. Lashkari, M. Mamun, and A. A. Ghorbani, Characterization of encrypted and VPN traffic using time-related features, in *Proc. Int. Conf. Information Systems Security and Privacy*, 2016, pp. 407–414.

[23] A. H. Lashkari, G. Draper-Gil, M. S. I. Mamun, and A. A. Ghorbani, Characterization of Tor traffic using time-based features, in *Proc. Int. Conf. Information Systems Security and Privacy*, 2017, pp. 253–262.