EdgeFlow: Statistical Feature Modulation for Efficient Encrypted Traffic Classification

Xiuwei Zhou
College Of Computer Science
Sichuan Normal University
Chengdu, China
1131839024xiu@gmail.com

Min Li*
College Of Internet
Sichuan Normal University
Chengdu, China
Limin@sicnu.edu.cn

Yuanfang Lu
College Of Computer Science
Sichuan Normal University
Chengdu, China
luyuanfang9@gmail.com

Chen Huang
College Of Computer Science
Sichuan Normal University
Chengdu, China
huangchen27217@gmail.com

Yaolin Sun
College Of Computer Science
Sichuan Normal University
Chengdu, China
linstar.yao@gmail.com

***Abstract*—Encrypted traffic classification is fundamental to network management and security, yet presents an inherent challenge: traffic flows comprise heterogeneous feature modalities with divergent statistical properties. Raw packet bytes exhibit high-dimensional, high-entropy representations encoding application-level patterns, whereas metadata constitute compact dense vectors capturing network-level characteristics. Conventional fusion strategies such as simple concatenation may lead to suboptimal representation learning due to dimensional and distributional discrepancies. We present EdgeFlow, a hierarchical Transformer architecture designed to address this multimodal fusion challenge through Statistical Feature Modulation (Stats-Mod). Rather than directly combining heterogeneous features in a shared space, Stats-Mod employs global statistical context to dynamically calibrate the distribution of local byte-level representations via learned affine transformations. This distribution-level conditioning mechanism facilitates cross-modal information integration while maintaining computational efficiency. We further incorporate Token Compression via vertical pooling to reduce sequence length, thereby improving inference throughput. Experimental evaluation across four benchmark datasets demonstrates that EdgeFlow achieves competitive classification performance while reducing computational overhead—up to $5\times$ higher inference throughput and $1.5\times$ lower GPU memory consumption compared to models of comparable scale.**

***Index Terms*—Encrypted Traffic Classification, Multimodal Fusion, Statistical Feature Modulation, Hierarchical Transformer, Edge Deployment**

I. Introduction

The widespread adoption of TLS 1.3 and end-to-end encryption has substantially limited the effectiveness of traditional Deep Packet Inspection (DPI) approaches [1], [2]. Consequently, encrypted traffic classification increasingly relies on encrypted byte patterns and side-channel metadata (e.g., packet sizes, inter-arrival times, direction), motivating classifiers that balance accuracy and computational efficiency for applications such as QoS provisioning and intrusion detection [3], [4].

*Corresponding author: Min Li

Encrypted traffic flows inherently contain two heterogeneous feature modalities: (1) Raw Bytes—high-dimensional sequences with high entropy encoding application-level patterns [5], [6]; and (2) Metadata Statistics—low-dimensional, dense vectors capturing network interaction patterns [7], [8]. Direct concatenation of these modalities causes dimensional disparity (low-dimensional metadata may be marginalized by high-dimensional byte embeddings) and distribution collision (mixing features with different statistical properties may degrade representations) [9], [10]. Existing methods face an accuracy–efficiency–fusion dilemma: sequence-based pre-trained models [5], [11] achieve strong accuracy but suffer from quadratic complexity; vision-based models [12], [13] offer efficient parallelism but may not fully exploit metadata; multimodal approaches [9] attempt fusion but rely on feature-level mixing without explicit distribution calibration.

We hypothesize that metadata statistics can more effectively serve as conditioning signals for byte-level features rather than being treated as competing input modalities. Drawing on conditional normalization techniques from the computer vision literature [14]–[16], we propose Stats-Mod: a mechanism that uses global statistics to generate affine transformation parameters for calibrating the distribution of local semantic features. Based on this insight, we present EdgeFlow, a hierarchical Transformer architecture that integrates Stats-Mod for distribution-level multimodal fusion and employs Token Compression via vertical pooling to reduce sequence length and improve inference throughput.

Our main contributions are summarized as follows:

- We formulate encrypted traffic classification as a multimodal fusion problem, explicitly addressing the dimensional and distributional discrepancies between raw packet bytes and metadata statistics.
- We propose Stats-Mod, a distribution-level conditioning mechanism that calibrates byte-level feature representations using metadata-derived affine parameters, enabling

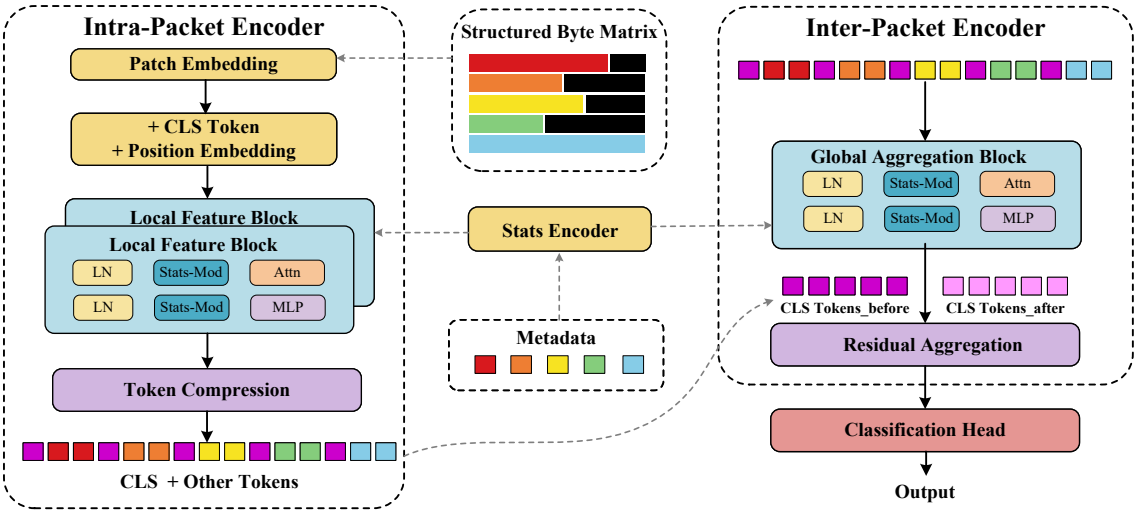

Fig. 1. The overall architecture of EdgeFlow.

effective cross-modal information integration without feature space expansion.

- We present EdgeFlow, a lightweight hierarchical Transformer that combines Stats-Mod with Token Compression, achieving competitive or improved accuracy across diverse classification tasks.
- Comprehensive experiments on four benchmark datasets demonstrate that EdgeFlow achieves competitive accuracy with up to $5\times$ higher inference throughput and $1.5\times$ lower GPU memory consumption compared to models of comparable scale, and substantially greater efficiency gains over larger pre-trained models.

The source code is publicly available at https://github.com/G034/edgeflow.

II. Related Work

Statistical Fingerprinting. Early approaches relied on hand-crafted features. Moore et al. [17] pioneered Bayesian classification using flow statistics, while subsequent works like AppScanner [7] and FlowPrint [8] utilized packet size distributions and temporal correlations for fingerprinting. IoT Sentinel [18] extended this to device identification. Although computationally efficient, these methods rely heavily on domain expertise [19], are vulnerable to traffic shaping [20], and overlook semantic information encoded in encrypted payloads [21].

Deep Learning for Traffic Classification. Deep learning automates feature extraction. CNNs [13], [22], [23] and RNNs [24] have shown promise but suffer from limited receptive fields or sequential bottlenecks [2]. Recently, pre-trained Transformers like ET-BERT [5], [6] and TrafficFormer [11] established state-of-the-art accuracy. Vision-based methods like YaTC [12] and FlowMAE [25] leverage masked autoencoding. Emerging research also explores large language models (LLMs) [26] and zero-day detection [27], [28], though robustness remains a critical challenge [4], [29].

Multimodal Traffic Analysis. Multimodal approaches combine diverse features. DISTILLER [9] and Lin et al. [10] jointly process payloads and statistics via separate branches, while TFE-GNN [30] utilizes traffic graphs for enhanced representation. However, these methods typically rely on simple concatenation or decision-level fusion, ignoring the significant distributional discrepancies between high-dimensional bytes and dense metadata.

Conditional Normalization. Techniques like AdaIN [14] and FiLM [15] enable distribution-level modulation by generating affine parameters from one modality to calibrate another. We adapt this principle to traffic classification, employing metadata statistics to dynamically modulate byte-level feature distributions, achieving effective cross-modal conditioning without feature space expansion.

TABLE I
Model Hyperparameters.

Input		Architecture	
Packets per flow P	5	Embedding dim D	128
Byte matrix size	30×32	Local Feature Blocks	2
Packet block size	6×32	Global Agg. Blocks	2
Patch size	2×2	Attention heads	4
Packet metadata dim	5	MLP ratio	2

III. Methodology

We propose EdgeFlow, a novel framework designed for efficient encrypted traffic classification. Formally, we define the classification task as modeling a flow composed of two distinct modalities: high-dimensional Raw Bytes and low-dimensional Metadata. To effectively integrate these modali-

ties, we introduce Statistical Feature Modulation, a mechanism devised for distribution-level fusion.

A. Problem Setup and Overview

Let a flow be denoted as $F = \{p_i\}_{i=1}^{P}$. Each packet p_i is characterized by: (i) Raw Bytes, represented as a structured byte matrix X_i that encodes high-dimensional semantic features; and (ii) Metadata, represented as a statistical vector s_i that captures low-dimensional global context. Table I summarizes the model configuration.

The EdgeFlow architecture comprises four integrated stages (Fig. 1):

1) Feature Extraction: The byte matrix is partitioned into patches, while metadata statistics are concurrently encoded via a Stats-Encoder.
2) Intra-Packet Encoding: Patch tokens are processed using Transformer blocks augmented with Stats-Mod for distribution-level conditioning.
3) Token Compression: A vertical pooling mechanism reduces the sequence length of patch tokens prior to flow-level aggregation.
4) Inter-Packet Aggregation: Cross-packet dependencies are modeled to derive the final classification decision.

B. Heterogeneous Data Representation

Each packet is represented in two complementary forms to preserve byte-level semantics while avoiding distortion from embedding statistics as pixel values.

1) Raw Bytes: Structured Byte Matrix: We formulate the raw byte representation as a grayscale matrix wherein each packet is mapped to a fixed-size block (Table I). Our analysis prioritizes the first 5 packets, as they encapsulate the critical connection establishment phase—including the TCP three-way handshake, TLS handshake, and initial application data exchange [7]. This phase preserves rich protocol fingerprints before encryption fully masks application semantics. For each packet, we extract 192 bytes (64 header + 128 payload) to encompass the complete IP/TCP headers and a sufficient portion of the encrypted payload, thereby balancing information retention with computational efficiency. Within each block, we explicitly delineate:

- Header region: Captures protocol fingerprints derived from IP and transport headers.
- Payload region: Contains high-entropy encrypted content that encodes application-level patterns.

Zero-padding is applied to handle missing bytes. This structured layout facilitates specific representation learning for headers and payloads [12], [25].

2) Metadata Statistical Feature Vector: For each packet, we extract five discriminative statistics guided by RFC specifications [31], which identify header fields with high discriminative power: (1) direction encodes client/server roles; (2) inter-arrival time (log-normalized) captures temporal patterns; (3) TTL reflects network topology; (4) IP length and (5) payload length characterize packet size distributions. These features are derived from standard protocol fields and demonstrate strong discriminative power with minimal redundancy [7], [8].

TABLE II
SUMMARY OF BENCHMARK DATASETS.

Dataset	Task	Flows	Classes
CICIOT2022	IoT Device Identification	2,126	9
ISCXVPN2016	Service Type Classification	1,317	6
USTC-TFC2016	Malware Detection	8,348	14
CSTNET-TLS1.3	Website Fingerprinting	46,372	120

TABLE III
TRAINING HYPERPARAMETERS.

Batch size	64	Label smoothing ε	0.1
Learning rate	10^{-3}	Optimizer	AdamW
Weight decay	10^{-2}	Scheduler	Cosine
Warmup epochs	5	Total epochs	100

C. Model Architecture

EdgeFlow employs a hierarchical Transformer architecture comprising an Intra-Packet Encoder for local pattern extraction and an Inter-Packet Encoder for flow-level modeling.

1) Intra-Packet Encoder with Stats-Mod: As shown in Fig. 1, packets are partitioned into patches and processed by stacked Local Feature Blocks containing Layer Normalization, Stats-Mod, Multi-Head Self-Attention (MSA), and MLP layers. Stats-Mod calibrates internal byte representations using external metadata statistics (e.g., timing, direction) as conditioning signals. A lightweight Stats-Encoder maps the metadata vector s to affine parameters (γ, β) to modulate the intermediate feature map Z:

$$\text{Stats-Mod}(Z, s) = \gamma(s) \cdot \frac{Z - \mu(Z)}{\sigma(Z) + \delta} + \beta(s), \tag{1}$$

where $\mu(Z)$ and $\sigma(Z)$ are instance statistics. This mechanism effectively integrates multimodal information at the distribution level without expanding the feature space.

2) Token Compression: To optimize efficiency, we introduce a Vertical Pooling operator to reduce sequence length. For a packet token grid $T^{(i)}$, the compressed representation $\bar{T}^{(i)}$ is computed by aggregating k vertically adjacent patches:

$$\bar{T}^{(i)}_{r,c} = \frac{1}{k} \sum_{j=1}^{k} T^{(i)}_{(r-1)k+j,c}, \tag{2}$$

With $k = 3$, this reduces the sequence length by $3\times$ while preserving the temporal structure encoded in the horizontal dimension.

3) Inter-Packet Encoder: The compressed tokens from all P packets are concatenated into a global sequence $\mathbf{Z}^{(0)}$ augmented with packet-level positional embeddings. This sequence is processed by Global Aggregation Blocks, which also utilize Stats-Mod conditioned on global flow statistics to model cross-packet dependencies. To synthesize the final flow representation $\mathbf{h}_{final}$, we employ Residual Aggregation,

TABLE IV
COMPARISON RESULTS ON CICIOT2022, ISCXVPN2016, USTC-TFC2016, AND CSTNET-TLS1.3.

Method	Params (M)	CICIOT2022		ISCXVPN2016		USTC-TFC2016		CSTNET-TLS1.3	
		Acc	F1	Acc	F1	Acc	F1	Acc	F1
AppScanner [7]	–	76.52%	76.81%	76.93%	74.85%	80.41%	78.53%	70.78%	68.42%
FlowPrint [8]	–	56.26%	55.14%	33.76%	17.09%	89.16%	89.82%	27.25%	22.56%
ET-BERT [5]	136.4	87.52%	86.62%	86.21%	85.80%	94.34%	93.12%	81.20%	78.84%
YaTC [12]	1.9	92.33%	90.88%	86.31%	88.15%	96.44%	95.12%	84.07%	81.33%
TFE-GNN [30]	44.3	91.25%	91.32%	87.18%	89.25%	95.56%	96.45%	79.85%	76.92%
TrafficFormer [11]	109.1	91.69%	90.44%	92.07%	91.39%	97.32%	97.85%	83.97%	83.14%
EdgeFlow (Ours)	0.7	92.49%	92.80%	90.39%	89.03%	97.84%	97.90%	84.57%	82.26%

fusing CLS tokens from the input ($\mathcal{C}_{pre}$) and output ($\mathcal{C}_{post}$) of the global encoder:

$$\mathbf{h}_{final} = \frac{1}{2P} \sum_{i=1}^{P} \left(c_{pre}^{(i)} + c_{post}^{(i)} \right). \tag{3}$$

This mechanism acts as a semantic skip connection, preserving strong local signals while integrating global context for the final classification.

D. Training Strategy

EdgeFlow is trained end-to-end using label-smoothed cross-entropy ($\varepsilon = 0.1$). Further training details are provided in Section IV.

E. Complexity Analysis

Time Complexity. Standard self-attention incurs quadratic $\mathcal{O}(L^2)$ cost. EdgeFlow reduces this via token compression (factor $k = 3$) to $\approx \mathcal{O}((L/k)^2)$, providing a $9\times$ reduction in attention computation. Furthermore, our hierarchical design avoids the expensive global attention across all packets, ensuring scalability.

Parameter Efficiency. EdgeFlow requires only 0.7M parameters, approximately $2.7\times$ fewer than the lightweight YaTC [12] (1.9M) and significantly smaller than large pre-trained models (e.g., ET-BERT's 136M). The Stats-Mod components introduce negligible overhead ($\sim$2K parameters), enabling efficient multimodal fusion without heavy parameterization.

IV. EXPERIMENTS

We evaluate EdgeFlow on four diverse benchmarks spanning IoT identification, VPN service classification, malware detection, and encrypted website fingerprinting.

A. Experimental Setup

Datasets. We utilize four diverse benchmarks summarized in Table II: CICIOT2022 [32] (IoT identification), ISCXVPN2016 [20] (VPN services), USTC-TFC2016 [33] (malware detection), and CSTNET-TLS1.3 [5] (TLS 1.3 website fingerprinting).

Preprocessing. We extract flows using bidirectional 5-tuples, filtering distinct non-traffic protocols (e.g., DNS, ICMP) and empty payloads. Classes are balanced via downsampling, and datasets are split 8:1:1 for training, validation, and testing.

Baselines. We compare against statistical approaches AppScanner [7] and FlowPrint [8]; pre-trained Transformers ET-BERT [5] and TrafficFormer [11]; and specialized deep learning models YaTC [12] and TFE-GNN [30].

Implementation. EdgeFlow is implemented in PyTorch 2.1 using an NVIDIA RTX 3070Ti. We use official codebases for baselines and fine-tune pre-trained models. Training hyperparameters are listed in Table III.

B. Main Results

Table IV compares EdgeFlow against state-of-the-art methods across four benchmark datasets.

As shown in Table IV, EdgeFlow demonstrates competitive or improved performance across the four benchmark datasets. On USTC-TFC2016 and CICIOT2022, EdgeFlow achieves the best results among all compared methods. For CSTNET-TLS1.3 with 120 fine-grained classes, EdgeFlow obtains the highest accuracy. On ISCXVPN2016, EdgeFlow exhibits lower performance than TrafficFormer, which may be attributed to two factors: (1) the relatively small dataset size (1,317 flows) limits training of lightweight models, and (2) VPN traffic exhibits more homogeneous byte patterns that benefit from the richer representations of large pre-trained models. Notably, EdgeFlow achieves these competitive results with substantially improved computational efficiency, as detailed in Section IV-C.

C. Efficiency Analysis

Beyond classification accuracy, we examine inference efficiency (Figures 2 and 3).

Throughput. EdgeFlow exhibits substantially higher inference throughput compared to baseline methods across varying batch sizes. Compared to YaTC (1.9M parameters), EdgeFlow achieves approximately $5\times$ higher throughput at batch size 64. Compared to larger pre-trained models (ET-BERT, TrafficFormer), EdgeFlow achieves up to $10\times$ higher throughput, attributed to its compact architecture and Token Compression.

Memory. EdgeFlow demonstrates approximately $1.5\times$ lower GPU memory consumption compared to YaTC, and up to $7\times$ lower compared to larger pre-trained approaches, facilitating deployment in resource-constrained environments.

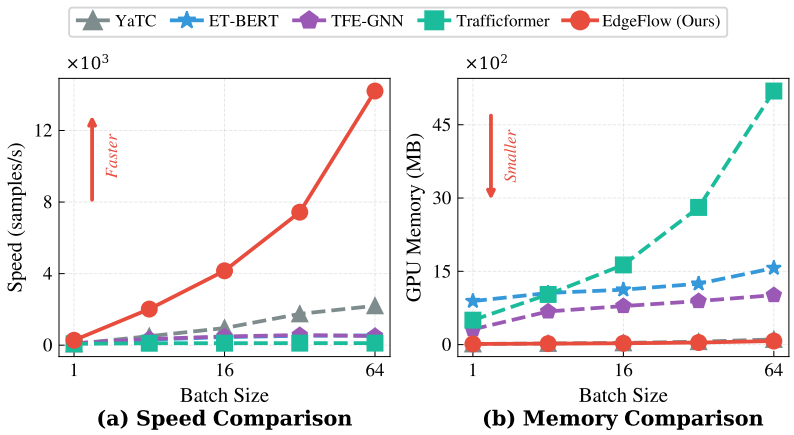

Fig. 2. Inference throughput (samples/second) and GPU memory consumption across different batch sizes.

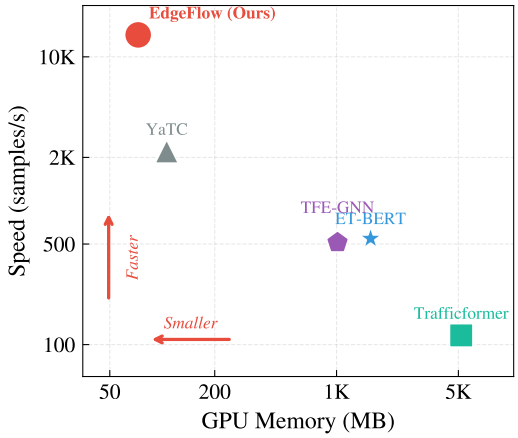

Fig. 3. Throughput-memory trade-off for different models at varying batch sizes.

Scalability. As batch size increases, EdgeFlow's throughput scales favorably while larger Transformers plateau due to memory constraints.

D. Ablation Study

We conduct comprehensive ablation studies to validate each component of EdgeFlow (Table V).

Stats-Mod vs. Concatenation. Removing Stats-Mod degrades performance, particularly on ISCXVPN2016 and CICIOT2022. Simple concatenation yields intermediate results, suggesting distribution-level modulation offers advantages over feature-level mixing [14], [15].

Token Compression. Removing compression changes accuracy in a dataset-dependent manner. On CSTNET-TLS1.3, the uncompressed variant achieves slightly higher F1 (82.73% vs. 82.26%), suggesting that for pure TLS 1.3 encrypted traffic with fine-grained classes, preserving full spatial resolution provides marginal benefits. However, this comes at the cost of increased inference latency, and compression remains beneficial for efficiency-focused deployments [34].

Hierarchical Design. Flattening packets reduces performance on CSTNET-TLS1.3, suggesting the Intra→Inter-Packet hierarchy better captures flow structure.

Input Modalities. Removing metadata degrades performance, particularly on CSTNET-TLS1.3, supporting multi-modal fusion for robustness [9], [10].

V. Conclusion

We proposed EdgeFlow, a hierarchical Transformer that addresses multimodal traffic fusion via Statistical Feature Modulation (Stats-Mod). By using metadata to dynamically calibrate byte-level representations, EdgeFlow facilitates effective cross-modal integration. Combined with token compression, it achieves competitive accuracy across four benchmarks while delivering up to $5\times$ higher throughput and $1.5\times$ lower memory usage than models of comparable scale.

Limitations and Future Work. EdgeFlow's reliance on the first five packets may limit robustness against early-stage obfuscation or packet loss. Future work will explore online streaming adaptations, adversarial robustness, and extending distribution-level conditioning to wider network tasks.

TABLE V
Ablation Study of EdgeFlow on All Datasets.

Variant	CICIOT2022		ISCXVPN2016		USTC-TFC2016		CSTNET-TLS1.3	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1
EdgeFlow (Full)	92.49%	92.80%	90.39%	89.03%	97.84%	97.90%	84.57%	82.26%
w/ Concat Metadata[a]	90.14%	90.36%	90.15%	87.03%	96.65%	96.76%	82.41%	81.25%
w/o Stats-Mod	85.92%	85.92%	79.55%	77.46%	97.72%	97.77%	82.15%	79.57%
w/o Token Compression[b]	90.14%	89.80%	88.64%	88.10%	97.25%	97.37%	84.19%	82.73%
w/o Hierarchical	91.08%	91.14%	87.12%	86.72%	97.84%	97.88%	81.42%	78.90%
w/o Metadata	89.67%	89.51%	76.52%	76.53%	97.60%	97.66%	81.82%	79.08%

[a]Replace Stats-Mod with simple concatenation fusion.
[b]Remove token compression, keep all 48 patch tokens.

REFERENCES

[1] F. Pacheco, E. Exposito, M. Gineste, C. Baudoin, and J. Aguilar, "Deep learning for encrypted traffic classification: An overview," *IEEE Communications Magazine*, vol. 57, no. 5, pp. 76–81, 2019.

[2] M. Shen, J. Zhang, L. Zhu, K. Xu, and X. Du, "Machine learning-powered encrypted network traffic analysis: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 1, pp. 791–824, 2023.

[3] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapè, "Mobile encrypted traffic classification using deep learning: Experimental evaluation, lessons learned, and challenges," *IEEE Transactions on Network and Service Management*, vol. 16, no. 2, pp. 445–458, 2019.

[4] D. Arp, E. Quiring, F. Pendlebury, A. Warnecke, F. Pierazzi, C. Wressnegger, L. Cavallaro, and K. Rieck, "Dos and don'ts of machine learning in computer security," in *Proceedings of the 31st USENIX Security Symposium*, 2022, pp. 3971–3988.

[5] X. Lin, G. Xiong, G. Gou, Z. Li, J. Shi, and J. Yu, "Et-bert: A contextualized datagram representation with pre-training transformers for encrypted traffic classification," in *Proceedings of the ACM Web Conference*, 2022, pp. 633–642.

[6] H. Y. He, Z. G. Yang, and X. N. Chen, "Pert: Payload encoding representation from transformer for encrypted traffic classification," in *ITU Kaleidoscope: Industry-Driven Digital Transformation*, 2020, pp. 1–8.

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

[8] T. Van Ede, R. Bortolameotti, A. Continella, J. Ren, D. J. Dubois, M. Lindorfer, D. Choffnes, M. Van Steen, and A. Peter, "Flowprint: Semi-supervised mobile-app fingerprinting on encrypted network traffic," in *Network and Distributed System Security Symposium*, 2020.

[9] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapè, "Distiller: Encrypted traffic classification via multimodal multitask deep learning," *Journal of Network and Computer Applications*, vol. 183, p. 102985, 2021.

[10] P. Lin, K. Ye, Y. Hu, Y. Lin, and C.-Z. Xu, "A novel multimodal deep learning framework for encrypted traffic classification," *IEEE/ACM Transactions on Networking*, vol. 31, no. 3, pp. 1369–1384, 2023.

[11] G. Zhou, X. Guo, Z. Liu, T. Li, Q. Li, and K. Xu, "Trafficformer: An efficient pre-trained model for traffic data," in *Proceedings of the IEEE Symposium on Security and Privacy*, 2025.

[12] R. Zhao, M. Zhan, X. Deng, Y. Wang, Y. Wang, G. Gui, and Z. Xue, "Yet another traffic classifier: A masked autoencoder based traffic transformer with multi-level flow representation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 4, 2023, pp. 5420–5427.

[13] T. Shapira and Y. Shavitt, "Flowpic: Encrypted internet traffic classification is as easy as image recognition," in *IEEE INFOCOM Workshops*, 2019, pp. 680–687.

[14] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1501–1510.

[15] E. Perez, F. Strub, H. de Vries, V. Dumoulin, and A. Courville, "Film: Visual reasoning with a general conditioning layer," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[16] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[17] A. W. Moore and D. Zuev, "Internet traffic classification using bayesian analysis techniques," in *ACM SIGMETRICS*, 2005, pp. 50–60.

[18] M. Miettinen, S. Marchal, I. Vittori, F. Papadopoulos, N. Asokan, and S. Tarkoma, "Iot sentinel: Automated device-type identification for security enforcement in iot," in *IEEE International Conference on Distributed Computing Systems*, 2017, pp. 2177–2184.

[19] M. Finsterbusch, C. Richter, E. Rocha, J.-A. Miller, and H.-J. Muller, "A survey of payload-based traffic classification approaches," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 2, pp. 1135–1156, 2014.

[20] G. Draper-Gil, A. H. Lashkari, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of encrypted and vpn traffic using time-related features," in *International Conference on Information Systems Security and Privacy*, 2016.

[21] B. Anderson, S. Paul, and D. McGrew, "Deciphering malware's use of tls (without decrypting it)," *Journal of Computer Virology and Hacking Techniques*, vol. 14, no. 3, pp. 195–211, 2018.

[22] M. Lotfollahi, M. Jafari Siavoshani, R. Shirali Hossein Zade, and M. Saberian, "Deep packet: A novel approach for encrypted traffic classification using deep learning," *Soft Computing*, vol. 24, pp. 1999–2012, 2020.

[23] W. Wang, M. Zhu, X. Zeng, X. Ye, and Y. Sheng, "End-to-end encrypted traffic classification with one-dimensional convolution neural networks," in *IEEE International Conference on Intelligence and Security Informatics*, 2017, pp. 43–48.

[24] C. Liu, L. He, G. Xiong, Z. Cao, and Z. Li, "Fs-net: A flow sequence network for encrypted traffic classification," in *IEEE INFOCOM*, 2019, pp. 1171–1179.

[25] Z. Hang, Y. Lu, Y. Wang, and Y. Xie, "Flowmae: Leveraging masked autoencoder for accurate, efficient and robust malicious traffic classification," in *Proceedings of the 26th International Symposium on Research in Attacks, Intrusions and Defenses*, 2023.

[26] T. Cui, X. Lin, S. Li, M. Chen, Q. Yin, Q. Li, and K. Xu, "Trafficllm: Enhancing large language models for network traffic analysis with generic traffic representation," 2025.

[27] J. F. Cevallos, A. Rizzardi, S. Sicari, and A. Coen-Porisini, "Hero: From high-dimensional network traffic to zero-day attack detection," *Computer Networks*, vol. 243, p. 111264, 2025.

[28] F. Deldar and M. Abadi, "Deep learning for zero-day malware detection and classification: A survey," *ACM Computing Surveys*, vol. 56, pp. 1–37, 2023.

[29] Y. Zhao, G. Dettori, M. Boffa, L. Vassio, and M. Mellia, "The sweet danger of sugar: Debunking representation learning for encrypted traffic classification," in *Proceedings of the ACM SIGCOMM Conference*, 2025.

[30] H. Zhang, L. Yu, X. Xiao, Q. Li, F. Mercaldo, X. Luo, and Q. Liu, "Tfe-gnn: A temporal fusion encoder using graph neural networks for fine-grained encrypted traffic classification," in *Proceedings of the ACM Web Conference*, 2023.

[31] N. Wickramasinghe, A. Shaghaghi, E. Ferrari, and S. Jha, "Less is more: Simplifying network traffic classification leveraging rfcs," in *Companion Proceedings of the ACM Web Conference*, 2025.

[32] S. Dadkhah, H. Mahdikhani, P. Danso, and A. A. Ghorbani, "Towards the development of a realistic multidimensional iot profiling dataset," in *Proceedings of the 19th Annual International Conference on Privacy, Security and Trust*, 2022.

[33] W. Wang, M. Zhu, J. Wang, X. Zeng, and Z. Yang, "Malware traffic classification using convolutional neural network for representation learning," in *International Conference on Information Networking*, 2017, pp. 712–717.

[34] Y. Rao, W. Zhao, B. Liu, J. Lu, J. Zhou, and C.-J. Hsieh, "Dynamicvit: Efficient vision transformers with dynamic token sparsification," in *Advances in Neural Information Processing Systems*, vol. 34, 2021, pp. 13 937–13 949.