ENACT: LLMs for Encrypted Network Traffic Classification

Xiaoyu Jiang[†], Chuanting Zhang[†], Wei Wang[†], Fangjun Li[‡], Liang Zhang[⋆], Basem Shihada[⋆], and Jingping Qiao[§]

[†]Shandong University, Jinan 250101, China
[‡]University of Manchester, Oxford Rd, Manchester M13 9PL, UK
[⋆]King Abdullah University of Science and Technology (KAUST), Thuwal 23955, Makkah Province, Saudi Arabia
[§]Shandong Normal University, Jinan 250358, China

***Abstract*—Accurate network traffic classification can empower future networks to enforce fine-grained quality of service (QoS) policies and enhance security monitoring. This paper proposes ENACT, a large language model framework to solve the encrypted network traffic classification problem. By prompt engineering and customized traffic representation, underlying features are well captured, leveraging a pre-trained LLM without fine-tuning. Experimental results demonstrate that classification performance in terms of accuracy and precision can be largely improved compared with the state-of-the-art models on two real-world benchmark datasets. The ablation study reveals the essential traffic representation structure for LLM and discrepancies among LLMs applied to cross-domain traffic classification tasks.**

***Index Terms*—Encrypted traffic classification, Large language model, Multi-head attention, Network security.**

I. Introduction

Network traffic classification [1], [2], which aims to classify flows generated by diverse applications and communication activities, is fundamental to network management and security. Accurate classification enables the detection of malicious traffic (e.g., attacks, malware) before it reaches clients, improving resource allocation for quality of service [3].

However, the increasing adoption of encryption techniques (e.g., TLS 1.3, VPN, Tor) and the diversity of emerging protocols significantly degrade the performance of traditional payload-based and statistical methods [4]. To address these challenges, deep learning has been widely explored for encrypted traffic classification, achieving superior feature extraction ability compared to conventional machine learning approaches. For example, CNN-based and sequence modeling methods have been successfully applied for end-to-end flow analysis [5], [6], though they are typically trained from scratch without pretraining or transfer learning. Moreover, recent studies demonstrate that pretrained and finetuned models can further enhance generalization and robustness [7]. In the network traffic classification domain, a representative work is EAPT [8], which employs adversarial pre-trained transformers for encrypted traffic classification, demonstrating that pretraining on large-scale data and finetuning improve performance in various encryption scenarios.

Despite these advances, existing deep learning models still face difficulties in capturing long-range dependencies across packets and often fail to generalize to unseen applications or protocols. This limitation is critical in practice, as encrypted traffic continuously evolves with the emergence of new services and protocol structures. Recently, large language models (LLMs) such as GPT [7] have demonstrated remarkable capabilities in contextual sequence modeling and knowledge transfer across domains [9]. Leveraging such pretrained knowledge provides a promising way to classify future traffic that follows previously unseen protocol patterns, without relying solely on large unlabeled datasets for pretraining. Furthermore, LLMs also offer the flexibility to incorporate heterogeneous auxiliary information, such as link type, in the form of prompt-based instructions. Inspired by this, Time-LLM [10] reformulates time-series prediction as language modeling, highlighting the potential of LLMs in cross-domain tasks [11].

Motivated by these observations, we propose ENACT, a large language model framework designed to solve the encrypted network traffic classification problem at the flow level, incorporating heterogeneous features to represent complex traffic patterns effectively. With temporal encoding, flow-aware tokenization, and prompt-based auxiliary information, ENACT leverages the pretrained knowledge of LLMs, without flow-specified pretraining to capture long-range dependencies and adapt to emerging protocols. Experimental evaluations demonstrate that ENACT achieves superior generalization and robustness compared to existing deep learning approaches.

II. Problem Formulation and Our Proposed ENACT Framework

A. Problem Formulation

Let $\mathcal{F} = \{\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_N\}$ denote a dataset of N network flows, where each flow $\mathbf{f}_i$ consists of P packets:

$$\mathbf{f}_i = \{\mathbf{p}_1^{(i)}, \mathbf{p}_2^{(i)}, \ldots, \mathbf{p}_P^{(i)}\}. \tag{1}$$

This work was supported in part by NSFC under Grant No. 62401338, in part by the Shandong Province Excellent Youth Science Fund Project (Overseas) under Grant No. 2024HWYQ-028.

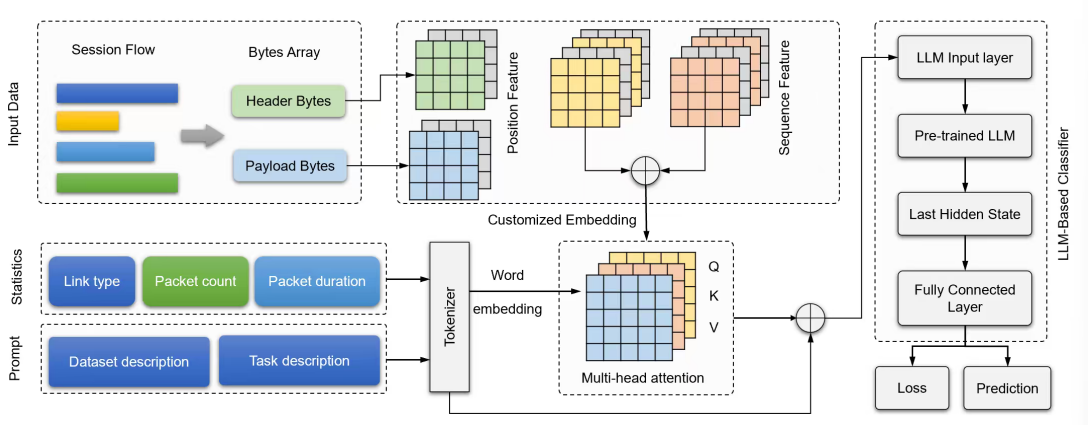

Fig. 1. Our proposed ENACT framework for encrypted network traffic classification.

Each packet $\mathbf{p}_j^{(i)}$ is represented as a concatenation of header and payload byte sequences:

$$\mathbf{p}_j^{(i)} = [\mathbf{h}_j^{(i)} \| \mathbf{l}_j^{(i)}], \tag{2}$$

where $\mathbf{h}_j^{(i)} \in \{0, 1, \ldots, 255\}^{T_h}$ represents the header of length T_h bytes, and $\mathbf{l}_j^{(i)} \in \{0, 1, \ldots, 255\}^{T_l}$ represents the payload of length T_l bytes.

The traffic classification problem is to learn a mapping function $\Phi : \mathcal{F} \to \mathcal{Y}$, where $\mathcal{Y} = \{y_1, y_2, \ldots, y_C\}$ is the set of C traffic categories. Our objective is to minimize the cross-entropy loss:

$$\mathcal{L} = -\frac{1}{N} \sum_{i=1}^{N} \sum_{c=1}^{C} y_{i,c} \log \hat{y}_{i,c}, \tag{3}$$

where $y_{i,c}$ is the ground truth label and $\hat{y}_{i,c}$ is the predicted probability for class c.

B. Proposed ENACT Framework

To solve the above formulated problem, we propose ENACT, an LLM framework for encrypted traffic classification, as shown in Fig. 1. It consists of five principal components: (1) custom embedding layer $\mathcal{E}$, (2) reprogramming layer $\mathcal{R}$, (3) frozen LLM backbone $\mathcal{M}$, (4) prompt construction $\mathcal{P}$, and (5) classification head $\mathcal{C}$. The complete forward pass can be expressed as:

$$\hat{\mathbf{y}} = \mathcal{C}(\mathcal{M}(\mathcal{P}(\mathcal{R}(\mathcal{E}(\mathbf{f}))))). \tag{4}$$

1) Data Preprocessing: Raw packet captures undergo a multi-stage preprocessing pipeline. First, continuous PCAP files are segmented into individual flows using 5-tuple identification (source IP, destination IP, source port, destination port, protocol). Each packet's network layer is extracted and converted to a hexadecimal representation.

For a packet $\mathbf{p}$, the header extraction function ξ_h and payload extraction function ξ_l are defined as:

$$\mathbf{h} = \xi_h(\mathbf{p}) = \text{hex}(\mathbf{p}_{IP})[: T_h], \tag{5}$$

$$\mathbf{l} = \xi_l(\mathbf{p}) = \text{hex}(\mathbf{p}_{Raw})[: T_l], \tag{6}$$

where $\mathbf{p}_{IP}$ denotes the IP layer and $\mathbf{p}_{Raw}$ denotes the raw payload. Sequences shorter than the fixed lengths are zero-padded.

2) Custom Embedding Layer: The embedding layer transforms raw byte sequences into dense vector representations. Given a flow with P packets, we separately embed headers and payloads, then combine them with positional and packet-level encodings.

Header and Payload Embedding: The header embedding function $\mathcal{E}_h$ maps byte-valued headers to d-dimensional vectors through a learnable lookup table:

$$\mathbf{E}_h = \mathcal{E}_h(\mathbf{H}) = \text{LN}(\mathbf{W}_h[\mathbf{H}] + \mathbf{b}_h), \tag{7}$$

where $\mathbf{H} \in \mathbb{Z}_{[0,255]}^{B \times P \times T_h}$ is the batch of header sequences, $\mathbf{W}_h \in \mathbb{R}^{256 \times d}$ is the embedding matrix, LN denotes layer normalization, and the result $\mathbf{E}_h \in \mathbb{R}^{B \times P \times T_h \times d}$.

Unlike headers, payloads exhibit local patterns that benefit from convolutional feature extraction. The payload embedding applies 1D convolution followed by activation:

$$\mathbf{E}_l = \text{LN}(\text{GELU}(\mathbf{W}_l * \mathbf{L} + \mathbf{b}_l)), \tag{8}$$

where $\mathbf{L} \in \mathbb{R}^{B \times P \times T_l}$ is the payload tensor, $\mathbf{W}_l \in \mathbb{R}^{d \times 1 \times 3}$ is the convolutional kernel with size 3, and $*$ denotes the convolution operation. The result $\mathbf{E}_l \in \mathbb{R}^{B \times P \times T_l \times d}$.

Positional Encoding: To inject positional information, we employ sinusoidal encodings:

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d}}\right), \tag{9}$$

$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d}}\right), \tag{10}$$

where pos is the position and i is the dimension index. The positional encoding matrix $\mathbf{PE} \in \mathbb{R}^{T_{max} \times d}$ is added to both header and payload embeddings.

Packet-level Encoding: To distinguish different packets within a flow, we introduce packet-level embeddings:

$$\mathbf{E}_p = \tanh(\mathbf{W}_p[\mathbf{idx}]), \tag{11}$$

where $\mathbf{idx} \in \{0, 1, \ldots, P-1\}^{B\times P}$ are packet indices, $\mathbf{W}_p \in \mathbb{R}^{P\times d}$ is initialized with large-scale values (uniform distribution $\mathcal{U}(-10, 10)$), and $\mathbf{E}_p \in \mathbb{R}^{B\times P\times d}$.

The final embeddings combine all encoding types:

$$\mathbf{E}_h \leftarrow \mathbf{E}_h + \mathbf{PE}[:T_h] + \mathbf{E}_p^{\text{broad}}, \tag{12}$$

$$\mathbf{E}_l \leftarrow \mathbf{E}_l + \mathbf{PE}[:T_l] + \mathbf{E}_p^{\text{broad}}, \tag{13}$$

where $\mathbf{E}_p^{\text{broad}}$ denotes broadcasting the packet embedding to match sequence dimensions.

C. Reprogramming Layer

The reprogramming layer bridges the modality gap between traffic embeddings (dimension $d = 768$) and LLM token space (dimension d_{llm}). We formulate this as a cross-attention mechanism where traffic embeddings serve as queries and LLM vocabulary embeddings serve as keys and values.

Vocabulary Mapping: To reduce computational cost, we first project the full vocabulary $\mathbf{V} \in \mathbb{R}^{|V|\times d_{llm}}$ to a reduced representation:

$$\tilde{\mathbf{V}} = \mathbf{W}_m \mathbf{V}^\top \in \mathbb{R}^{K\times d_{llm}}, \tag{14}$$

where $\mathbf{W}_m \in \mathbb{R}^{K\times|V|}$ is a learnable projection matrix with $K = 1024 \ll |V|$.

Multi-head Cross-Attention: The reprogramming operation $\mathcal{R}$ employs multi-head attention with H heads:

$$\mathcal{R}(\mathbf{E}, \tilde{\mathbf{V}}) = \mathbf{W}_o \text{Concat}(\text{head}_1, \ldots, \text{head}_H) \tag{15}$$

where head_h is computed as $\text{Attention}(\mathbf{Q}_h, \mathbf{K}_h, \mathbf{V}_h)$. The query, key, and value projections are defined as:

$$\mathbf{Q}_h = \mathbf{W}_h^Q \mathbf{E} \in \mathbb{R}^{B\times L\times d_k}, \tag{16}$$

$$\mathbf{K}_h = \mathbf{W}_h^K \tilde{\mathbf{V}} \in \mathbb{R}^{K\times d_k}, \tag{17}$$

$$\mathbf{V}_h = \mathbf{W}_h^V \tilde{\mathbf{V}} \in \mathbb{R}^{K\times d_k}, \tag{18}$$

where $\mathbf{E} \in \mathbb{R}^{B\times L\times d}$ is the flattened traffic embedding with $L = P(T_h+T_l)$, and $d_k = d_{llm}/H$ is the dimension per head. The attention mechanism is computed as:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V}. \tag{19}$$

The output projection combines all heads:

$$\mathbf{W}_o \in \mathbb{R}^{d_{llm}\times(H\cdot d_k)}, \quad \mathcal{R}(\mathbf{E}) \in \mathbb{R}^{B\times L\times d_{llm}}. \tag{20}$$

D. Prompt Construction

To provide task context to the LLM, we construct domain-specific prompts. Let $\mathcal{T}$ denote the tokenizer associated with the LLM. The prompt for flow $\mathbf{f}_i$ with metadata can be described as $\mathbf{s}_i$, including dataset description, task description, packet count, link, and duration. The prompt embedding is obtained via:

$$\mathbf{E}_s = \mathcal{M}_{emb}(\mathcal{T}(\mathbf{s})) \in \mathbb{R}^{B\times L_s\times d_{llm}}, \tag{21}$$

where $\mathcal{M}_{emb}$ is the LLM's embedding layer and L_s is the prompt length. Similarly, an objective statement $\mathbf{o}$ = "predict the flow category" is embedded as:

$$\mathbf{E}_o = \mathcal{M}_{emb}(\mathcal{T}(\mathbf{o})) \in \mathbb{R}^{B\times L_o\times d_{llm}}. \tag{22}$$

TABLE I
FIXED-NUMBER PACKET SEGMENTATION FOR THE ISCX-TOR 2016 DATASET

Category	Packets per Segment	Total Packets
Browsing	60	637,721
Chat	100	21,018
Audio	100	425,936
Video	200	2,418,102
File Transfer	300	4,006,834
Mail	300	392,413
P2P	150	5,726,345
VoIP	100	1,550,086

E. Sequence Construction

The complete input sequence interleaves prompt, traffic embeddings, and objective:

$$\mathbf{X} = [\mathbf{E}_s \| \mathbf{E}_h^{(1)} \| \mathbf{E}_l^{(1)} \| \cdots \| \mathbf{E}_h^{(P)} \| \mathbf{E}_l^{(P)} \| \mathbf{E}_o], \tag{23}$$

where $\|$ denotes concatenation along the sequence dimension, yielding $\mathbf{X} \in \mathbb{R}^{B\times L_{total}\times d_{llm}}$ with total length:

$$L_{total} = L_s + P(T_h + T_l + 2) + L_o. \tag{24}$$

The additional 2 accounts for separator tokens inserted between headers and payloads.

F. LLM Processing and Classification

The frozen LLM processes the input sequence:

$$\mathbf{Z} = \mathcal{M}(\mathbf{X}) \in \mathbb{R}^{B\times L_{total}\times d_{llm}}, \tag{25}$$

where $\mathcal{M}$ consists of $N_L = 6$ transformer layers. We extract the last token's representation as the flow-level embedding:

$$\mathbf{z}_{cls} = \mathbf{Z}[:, -1, :] \in \mathbb{R}^{B\times d_{llm}}. \tag{26}$$

The classification head applies a linear projection:

$$\hat{\mathbf{y}} = \text{softmax}(\mathbf{W}_c \mathbf{z}_{cls} + \mathbf{b}_c), \tag{27}$$

where $\mathbf{W}_c \in \mathbb{R}^{C\times d_{llm}}$ and $\hat{\mathbf{y}} \in \mathbb{R}^{B\times C}$ represents class probabilities.

G. Computational Complexity

The complexity of ENACT is dominated by the frozen LLM backbone, scaling as $\mathcal{O}(N_L L_{total}^2 d_{llm})$. Reprogramming adds another $\mathcal{O}(BLKd_k)$ computation, while embedding contributes negligible overhead.

III. Experimental Results and Analysis

A. Dataset and Preprocessing

We evaluate ENACT on two benchmark datasets: USTC-TFC 2016 [12], comprising 20 TLS-encrypted benign and malware classes, and ISCX-Tor 2016 [13], featuring 8 Tor-encapsulated categories. Raw traces are filtered to retain IP-layer packets and grouped into session flows via the standard 5-tuple.

For USTC-TFC, we downsample to 5,000 flows per category for binary and 20-class classification. For ISCX-Tor, flows are segmented into shorter units to overcome anonymity-related reconstruction issues, with 1,800 segments sampled per

TABLE II
CLASSIFICATION PERFORMANCE COMPARISON

USTC-TFC2016 (20-class)				
Model	Acc.	Prec.	Rec.	F1
AppScaner	58.22	70.22	46.38	49.41
KNN	92.25	92.46	92.22	92.13
1D-CNN	98.13	98.32	98.19	98.13
2D-CNN	97.77	97.95	97.83	97.75
LSTM	97.25	97.22	97.22	97.21
ENACT	**98.50**	**98.49**	**98.51**	**98.49**
Tor Dataset (8-class)				
Model	Acc.	Prec.	Rec.	F1
AppScaner	90.56	88.85	88.36	88.56
KNN	60.50	65.72	57.40	58.10
1D-CNN	97.32	97.42	97.13	97.22
2D-CNN	97.89	97.52	97.57	97.54
LSTM	88.73	90.70	86.68	87.79
ENACT	**99.59**	**99.66**	**99.67**	**99.66**

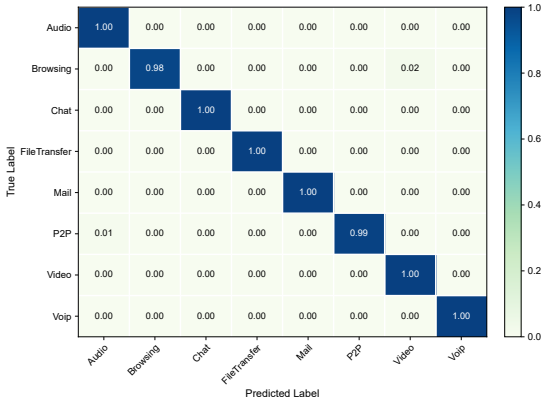

Fig. 2. Confusion matrix results obtained on the ISCX-Tor dataset.

class. Each instance is represented by its first $M = 5$ packets, which are processed via truncation and padding to ensure a fixed length compatible with our customized embedding. Additionally, we extract three flow-level statistical features: total packet count, link type, and flow duration. Finally, both datasets are randomly partitioned into training, validation, and test sets using an 8:1:1 ratio.

B. Hyper-parameters and Configurations

In our implementation, we utilize the DeepSeek-R1-Distill-Qwen-7B model [14] as the pre-trained backbone, which dictates a hidden dimension $d_{llm} = 3584$. The customized embedding dimension D is set to 768, and the reprogramming layer employs $H = 8$ attention heads with a key dimension $D_{\text{keys}} = 32$. To prevent overfitting, a dropout rate of 0.1 is applied throughout the architecture. During the training phase, we use a batch size $B = 4$ and employ the Adam optimizer [15] alongside a OneCycle learning rate scheduler [16] with an initial learning rate of 1×10^{-4}. To ensure numerical stability and accuracy, all experiments are conducted using full-precision training without the use of automatic mixed precision.

We evaluate classification performance using four standard metrics, i.e., *Accuracy*, *Precision*, *Recall*, and *F1-score*. Accuracy measures the proportion of correctly classified samples, while Precision and Recall quantify the correctness and completeness of positive predictions, respectively. Several representative baseline models are considered for comparison, including AppScaner [17], KNN [4], 1D-CNN, 2D-CNN [18], and LSTM [19].

C. Model Performance

Table II compares ENACT with classical machine-learning and deep-learning baselines on two benchmark datasets. On USTC-TFC2016 (20 classes), ENACT achieves the best overall performance, with an accuracy of 98.50% and an F1-score of 98.49%, demonstrating a consistent yet moderate improvement over strong CNN-based methods. This indicates that ENACT provides a more balanced precision–recall trade-off while preserving the advantages of convolutional feature extraction.

On the more challenging Tor dataset (8 classes), ENACT demonstrates a substantially larger performance gain, achieving 99.59% accuracy and 99.66% F1-score, outperforming the best CNN baseline by approximately 1.7% in accuracy and 2.2% in F1-score. These results suggest that ENACT is more effective at capturing global flow-level patterns and long-range dependencies under strong encryption and traffic obfuscation, leading to improved robustness and generalization.

Fig. 2 illustrates the confusion matrix of ENACT on the ISCX-Tor dataset. As shown in the figure, ENACT achieves near-perfect classification performance across all eight traffic categories, with the majority of samples correctly classified along the diagonal. In particular, six traffic classes (Audio, Chat, FileTransfer, Mail, Video, and VoIP) are classified with perfect accuracy, while the remaining two classes (Browsing and P2P) exhibit only marginal confusion, with classification accuracies of approximately 98% and 99%, respectively. This result demonstrates that ENACT effectively captures discriminative flow-level patterns under Tor encryption and remains robust in highly anonymized and obfuscated traffic environments.

D. Impact of Key Components

To investigate the impact of different pretrained LLM backbones and key architectural components, we conduct ablation studies on the 20-class classification task of the USTC-TFC 2016 dataset. All model variants are trained under identical settings, including the optimizer, learning rate, and batch size, for a total of three epochs. Performance is evaluated on the test set using Accuracy, Precision, Recall, and F1-score.

Effect of Different LLM Backbones: We evaluate the impact of different pretrained LLM backbones in the ENACT framework by replacing only the backbone while keeping all other components unchanged. As shown in Table III, DeepSeek-based models consistently outperform other LLMs, with DeepSeek-1.5B achieving the best overall performance

TABLE III
IMPACT OF LLM BACKBONE CHOICE

LLM	d_{llm}	Acc.	Prec.	Rec.	F1
DeepSeek-7B	3584	97.11	97.12	96.99	97.01
DeepSeek-1.5B	1536	**97.31**	97.47	97.29	97.37
DeepSeek-Llama-8B	4096	97.30	97.32	97.27	97.32
LLaMA-7B	4096	95.25	95.43	95.17	95.27
Longformer	768	43.64	45.08	43.51	39.28

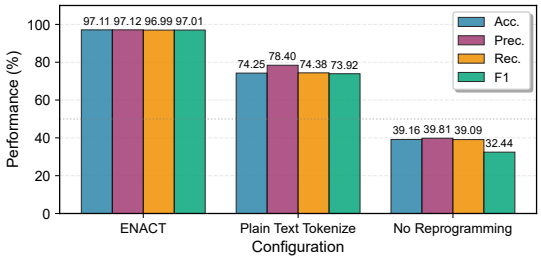

Fig. 3. Ablation study on removing reprogramming layer.

despite its smaller hidden dimension. In contrast, general-purpose models such as LLaMA-7B and Longformer exhibit noticeably lower accuracy and F1-scores, indicating that backbone pretraining characteristics play a critical role in capturing encrypted traffic patterns. These results suggest that compact yet well-aligned LLMs can be more effective within ENACT under the same training budget.

Effect of Model Architecture: We further analyze the contribution of key architectural components, with a focus on the customized embedding and reprogramming modules. All experiments use DeepSeek-R1-Distill-Qwen-7B as the backbone LLM. Specifically, we compare the full ENACT model with two variants: (i) a tokenizer-only model that directly applies the LLM tokenizer to raw flow bytes, and (ii) a model without the reprogramming module, where customized embeddings are not aligned to the LLM representation space. As shown in Fig. 3, removing the reprogramming module leads to a substantial performance degradation across all metrics, while direct tokenization also results in a notable decrease in accuracy. These results demonstrate that both flow-aware embedding and reprogramming are essential for effectively adapting encrypted traffic features to LLM-based classification.

IV. CONCLUSION

This paper presents a novel framework for encrypted network traffic classification that consistently outperforms existing approaches by leveraging a frozen pretrained LLM through a reprogramming mechanism, rather than relying on flow-specific pretrained encoders. The results demonstrate that adapting flow representations to the LLM embedding space is more effective than directly tokenizing raw traffic bytes, enabling strong generalization to diverse and evolving encryption protocols. Ablation studies further confirm the critical role of the reprogramming module in bridging the representational gap between encrypted traffic features and LLMs, while revealing the limitations of direct LLM tokenization for numerical byte sequences. These findings highlight the potential of large language models beyond natural language processing for analyzing encrypted traffic, and future work will explore their robustness under heterogeneous traffic distributions, as well as few-shot and zero-shot classification scenarios.

REFERENCES

[1] J. Zhang, X. Chen, Y. Xiang, W. Zhou, and J. Wu, "Robust network traffic classification," *IEEE/ACM Transactions on Networking*, vol. 23, no. 4, pp. 1257–1270, 2015.

[2] T. Wang, X. Xie, W. Wang, C. Wang, Y. Zhao, and Y. Cui, "Netmamba: Efficient network traffic classification via pre-training unidirectional mamba," in *2024 IEEE 32nd International Conference on Network Protocols (ICNP)*, pp. 1–11, 2024.

[3] P. Velan, M. Cermák, P. Čeleda, and M. Drašar, "A survey of methods for encrypted traffic classification and analysis," *International Journal of Network Management*, vol. 25, no. 5, pp. 355–374, 2015.

[4] M. Shen, K. Ye, X. Liu, L. Zhu, J. Kang, S. Yu, Q. Li, and K. Xu, "Machine learning-powered encrypted network traffic analysis: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 1, pp. 791–817, 2023.

[5] W. Wang, M. Zhu, J. Wang, X. Zeng, and Z. Yang, "End-to-end encrypted traffic classification with one-dimensional convolution neural networks," in *2017 International Conference on Computing, Networking and Communications (ICNC)*, pp. 1–6, IEEE, 2017.

[6] C. Liu, L. He, G. Xiong, Z. Cao, and Z. Li, "Fs-net: A flow sequence network for encrypted traffic classification," in *2019 IEEE International Conference on Communications (ICC)*, pp. 1–6, IEEE, 2019.

[7] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, *et al.*, "Improving language understanding by generative pre-training," 2018.

[8] M. Zhan, J. Yang, D. Jia, and G. Fu, "Eapt: An encrypted traffic classification model via adversarial pre-trained transformers," *Computer Networks*, vol. 237, p. 110081, 2023.

[9] C. Zhang, H. Zhang, J. Qiao, Z. Li, and M.-S. Alouini, "Wireless traffic prediction with large language model," 2025.

[10] M. Jin, S. Wang, L. Ma, Z. Chu, J. Y. Zhang, X. Shi, P.-Y. Chen, Y. Liang, Y.-F. Li, S. Pan, and Q. Wen, "Time-llm: Time series forecasting by reprogramming large language models," 2024.

[11] C. Zhang, H. Zhang, J. Qiao, Z. Li, and M.-S. Alouini, "Tides: Traffic intelligence with deepseek-enhanced spatial–temporal prediction," *IEEE Journal on Selected Areas in Communications*, vol. 44, pp. 2544–2558, 2026.

[12] W. Wang, M. Zhu, X. Zeng, X. Ye, and Y. Sheng, "Malware traffic classification using convolutional neural network for representation learning," in *2017 International Conference on Information Networking (ICOIN)*, pp. 712–717, IEEE, 2017.

[13] A. Habibi Lashkari, G. Draper Gil, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of tor traffic using time based features," in *Proceedings of the 3rd International Conference on Information Systems Security and Privacy - ICISSP*, pp. 253–262, INSTICC, SciTePress, 2017.

[14] DeepSeek-AI, D. Guo, D. Yang, *et al.*, "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning," *arXiv preprint arXiv:2501.12948*, 2025.

[15] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017.

[16] L. N. Smith and N. Topin, "Super-convergence: Very fast training of neural networks using large learning rates," 2018.

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

[18] M. Seydali, F. Khunjush, B. Akbari, and J. Dogani, "Cbs: A deep learning approach for encrypted traffic classification with mixed spatio-temporal and statistical features," *IEEE Access*, vol. PP, pp. 1–1, 01 2023.

[19] M. Ge, X. Yu, and L. Liu, "Robot communication: Network traffic classification based on deep neural network," *Frontiers in Neurorobotics*, vol. 15, p. 648374, 2021.