

# MET-LLM: Enhancing Large Language Models for Malicious Encrypted Traffic Detection

Yongjun Huang<sup>a</sup>, Pengfei Du<sup>b</sup>, Ruifan Li<sup>\*a</sup>, Rujun Liu<sup>c</sup>, Xiaoyong Li<sup>d</sup>

<sup>a</sup>*School of Artificial Intelligence, Beijing University of Posts and Telecommunications*

<sup>b</sup>*School of Cyberspace Security, Shandong University of Political Science and Law*

<sup>c</sup>*Beijing Tongtech Co., Ltd*

<sup>d</sup>*School of Cyberspace Security, Beijing University of Posts and Telecommunications*

---

## Abstract

Modern networks have spurred growth in both legitimate and malicious activities concealed within encrypted traffic. Traditional machine learning approaches to traffic classification struggle with scalability to new protocols, diverse tasks, and adaptability to emerging threats. To address these issues, we propose **MET-LLM**, a novel framework for **Malicious Encrypted Traffic** detection that integrates domain-specific tokenization, a pretrained large language model, and a dynamic adaptive tuning adaptor. MET-LLM addresses the modal gap between natural language and heterogeneous network traffic data by partitioning each traffic sample into distinct headers and payloads and leveraging a specialized tokenizer trained on a large-scale traffic corpus to extend the base vocabulary of the underlying language model. Building on a domain-adapted pretrained model fine-tuned on extensive security-related corpora, MET-LLM captures critical contextual nuances distinguishing benign from malicious flows. Its dynamic adaptive tuning adaptor facilitates efficient parameter updates via adaptation prompt injection, adversarial training, and dynamic masking, enabling rapid adaptation to evolving network conditions and attack strategies. Extensive evaluations on benchmark datasets, including ISCX Tor 2016, ISCX VPN 2016, APP-53 2023, and CSTNET 2023, demonstrate that MET-LLM's superior precision, recall, and F1 scores over state-of-the-art methods, affirming its efficacy and robustness in real-world cybersecurity applications. Code is available at: <https://github.com/Superagentsys/MET-LLM>.

**Keywords:** Malicious Encrypted Traffic detection, Large Language Model,

---

## 1. Introduction

The rise in encrypted network traffic has significantly transformed the cybersecurity landscape. Although encryption protocols such as Transport Layer Security (TLS) ensure privacy, studies indicate that over 95% of web traffic is encrypted [1, 2], they present challenges for security monitoring infrastructures. This adoption of encryption is vital for preserving data confidentiality and integrity; however, it has altered how organizations approach threat detection and traffic analysis [3, 4].

As illustrated in Figure 1, encrypted traffic exhibits characteristics that complicate security analysis. Cryptographic transformations obscure payload content, rendering traditional deep packet inspection (DPI) ineffective without private key access. While encryption strengthens communication privacy, it reduces visibility of potential threats within network infrastructure [3, 5]. This limitation has redirected research toward analytical methodologies based on metadata extraction, statistical pattern recognition, and behavioral modeling for intrusion detection and traffic classification [6, 7].

Threat intelligence highlights the security implications of increased encryption. Malicious actors leverage encryption to obfuscate attack vectors, with studies reporting that approximately 70% of malware campaigns utilize encrypted channels, an 30% increase since 2018 [8].

Techniques such as SSL/TLS tunneling for command-and-control communications, encrypted data exfiltration, and malware delivery mechanisms allow threat actors to blend with legitimate encrypted traffic patterns [9].

Despite advances in machine learning approaches for encrypted traffic analysis, existing solutions exhibit constraints in practical deployment [10, 11, 12]. First, conventional approaches rely on manual feature engineering and struggle to encode the semantic structure of network data, such as protocol structures, timing patterns, and encrypted payloads. This limitation reduces resilience to obfuscation techniques employed by persistent threats [12, 13]. Second, these methods are tailored to specific detection tasks, hindering knowledge transfer and generalization across tasks [14]. Third, existing methods lack the flexibility adapt to the evolving nature of modern threat landscapes [15].

Recent development in large language models (LLMs), such as Deepseek [16] and Llama [17], has enhanced the ability to process complex data patterns across diverse domains. Initially designed for text generation, these transformer-based models with extensive parameterization now extend to managing scenarios across different data types. Their strong contextual understanding offers a promising approach for addressing challenges in encrypted traffic analysis [18, 19].

Applying LLMs to malicious encrypted traffic detection represents an advancement in network security. Though originally designed for natural language tasks, these transformer-based architectures can identify com-

---

*Email addresses:* huangyj2022@bupt.edu.cn (Yongjun Huang), 002344@sdupl.edu.cn (Pengfei Du), rfli@bupt.edu.cn (Ruifan Li\*), liurj@mytech.com.cn (Rujun Liu), lixiaoyong@bupt.edu.cn (Xiaoyong Li)

Encrypted Payload
65:a0:b6:23:a7:29:f9:0b:cf:76:bb: 9b:fb:f2:81:e1:72:2e:b7:33:b7:29:f9: 9b:fb:f2:81:e1:72:2e:b7:33:b7:29: :f9:fb:f2:81:e1
IP
Version: 4 Header Length: 20 bytes Total Length: 1500 bytes Identification: 0x6350 Flags: DF (Don't Fragment) TTL: 64 Protocol: TCP (6) Source IP: 10.67.797.1 Destination IP: 11.34.2.5
Ethernet
Destination MAC: 88:e9:fe:70:51 Source MAC: 04:d9:f5:9b:5c Ether Type: 0x0800 (IPv4)
Frame
Encapsulation Type: 1 Timestamp: Jul 28, 20214 Frame Length: 1614 bytes

Figure 1: Network packet showing frame, Ethernet, IP, and TCP details with an encrypted payload.

plex patterns, generalize across classification objectives, and adapt to new traffic patterns with minimal fine-tuning. These capabilities align with the requirements of network security applications [20, 21].

LLMs offer key advantages for detecting malicious encrypted traffic. First, they automatically learn complex patterns in encrypted traffic data without relying on extensive manual feature engineering, which is particularly valuable for detecting advanced evasion malware. Second, their transformer architecture captures long-range dependencies within network flows, enabling identification of attacks spanning multiple connections or extended durations. Third, LLMs’ generalization capabilities allow them to adapt to new threats using task-specific training data.

Recent research has explored traffic-specific pre-training and specialized tokenization strategies to better align LLMs with network analysis tasks. Studies such as TrafficFormer [22] highlight the potential of LLMs in encrypted traffic analysis by integrating natural language processing (NLP) into network security applications.

However, several challenges hinder their effective application. The modal gap between natural language and network data, such as structured headers and encrypted payloads, limits the efficacy of conventional tokenizers [23]. Retraining large-parameter models for evolving threats incurs high computational costs, necessitating more efficient adaptation mechanisms [24]. To address these challenges, **we leverage segment-specific tokenization to transform heterogeneous network features into se-**

**mantically rich input representations.** We develop a specialized tokenization mechanism that transforms heterogeneous network features into semantically rich input representations. This component mitigates the modality mismatch by implementing segment-specific encoding strategies optimized for the unique characteristics of network traffic.

This study proposes **Malicious Encrypted Traffic Large Language Model (MET-LLM)**, a unified framework for malicious encrypted traffic detection that integrates domain-specific tokenization, a pretrained large language model, and a dynamic adaptive tuning adaptor. Our framework combines transformer-based representation learning and parameter-efficient adaptation mechanisms. Evaluations on four benchmark datasets reveal that MET-LLM outperforms current state-of-the-art detection methods, yielding F1 scores exceeding 0.96 across traffic scenarios and attack categories.

In summary, the key contributions of this study are as follows.

- We propose a tokenization strategy that bridges the modal gap between natural language and network traffic data. Each traffic sample is partitioned into *head* and *payload* segments, with segment-specific tokenization applied using Byte Pair Encoding (BPE) trained on a traffic-domain corpus [25]. Our approach enables effective representation of both human-readable header information and hex-encoded payloads.
- We leverage a domain-adapted pretrained language model, Deepseek, enhanced with security-specific pretraining. This integration enables the model to capture contextual patterns in network traffic, enhancing malicious activity detection.
- We introduce the dynamic adaptive tuning adaptor (DATA), a novel fine-tuning module that supports flexible and parameter-efficient model adaptation to evolving operational environments and network conditions. DATA incorporates adaptation prompt injection, adversarial training, and dynamic masking to ensure robustness against traffic obfuscation and protocol modifications while efficiently integrating new task-specific knowledge.

## 2. Related Work

We categorize related research into two domains: fingerprinting-based methods and deep learning-based approaches.

### 2.1. Fingerprinting-based Methods

Fingerprinting approaches represent the earliest systematic attempts to analyze encrypted traffic without decryption. These techniques rely on statistical, temporal, and behavioral features extracted from the network flows to construct distinctive signatures or “fingerprints.”

Foundational work by Moore and Zuev [26] demonstrated that supervised learning on statistical flow features could effectively classify traffic, establishing core methodological groundwork for subsequent research.

Taylor *et al.* [27] extended these concepts with *AppScanner*, an automated system that leveraged burst-level statistics and temporal features to fingerprint smartphone applications from encrypted traffic. Their approach achieved 87% accuracy across a diverse range of mobile applications without requiring decryption.

Website fingerprinting has emerged as a prominent research focus within encrypted traffic analysis. Panchenko *et al.* [28] developed a weight-based classification approach capable of operating at Internet scale with strong robustness against various countermeasures, maintaining computational efficiency and high accuracy. Hayes and Danezis [29] introduced *k-fingerprinting*, a scalable framework employing random decision forests to identify website access patterns from encrypted traffic, improving classification accuracy and resilience against minor traffic perturbations.

Al-Naami *et al.* [30] further refined fingerprinting by incorporating bidirectional dependency analysis, which adaptively captures temporal and directional features in encrypted flows. Their approach proved effective across diverse applications, network environments, and encryption protocols. Further developments by Sirinam *et al.* [31] demonstrate that deep learning techniques could circumvent conventional fingerprinting defenses, highlighting the dynamic, adversarial nature of encrypted traffic analysis.

Recent progress has aimed to reduce reliance on large labeled datasets. Van Ede *et al.* [32] proposed *FlowPrint*, a semi-supervised fingerprinting framework for mobile application identification. By clustering network flows based on temporal correlations and destination similarity, FlowPrint enables effective classification with minimal labeled data and adapts well to previously unseen applications.

## 2.2. Deep Learning-based Techniques

The advent of deep learning has significantly advanced encrypted traffic analysis by enabling automatic feature extraction and complex pattern recognition directly from raw traffic data. These approaches have reduced dependence on manual, domain-specific feature engineering while improving classification performance across diverse traffic types.

Convolutional Neural Networks (CNNs) were among the first deep learning models applied to this task. Wang *et al.* [33] highlighted their effectiveness for end-to-end encrypted traffic classification by treating raw bytes as one-dimensional input sequences. This approach eliminated the need for manual feature extraction and achieved competitive accuracy on standard benchmarks. Building on this foundation, Liu *et al.* [34] introduced Flow Sequence Networks (FS-Net), a specialized architecture that models temporal relationships within network flows. FS-Net captured both short- and long-term dependencies, significantly surpassing the performance of traditional methods, particularly for traffic with complex temporal patterns.

Recurrent Neural Networks (RNNs) and their variants have been extensively explored for their strength in modeling sequential data. Lotfollahi *et al.* [15] presented *Deep Packet*, a comprehensive framework that combines CNNs and stacked autoencoders to perform traffic characterization and application identification directly from encrypted

data. The framework effectively adapted to various traffic types without relying on predefined features, yielding high accuracy exceeding 98% for certain applications. For specialized environments, such as Industrial Internet of Things (IIoT), Lin *et al.* [35] proposed TSCRNN, integrating convolutional and recurrent layers to efficiently capture both spatial and temporal features from encrypted traffic flows. Designed for resource-constrained environments, TSCRNN delivers strong performance while maintaining computational efficiency.

The adoption of transformer architectures has recently spurred notable progress in encrypted traffic analysis. Pioneering works by He *et al.* [20] introduced PERT, leveraging self-attention mechanisms to generate robust payload representations and effectively model long-range dependencies, overcoming a key limitation of earlier methods. Subsequently, Lin *et al.* [18] developed ET-BERT, adapting BERT's pre-training objectives to the network domain to create contextualized datagram representations that markedly improved classification accuracy. Recent contributions have further refined transformer-based techniques. Zhao *et al.* [19] proposed YATC (Yet Another Traffic Classifier), combining masked autoencoder principles with multi-level flow representation. Zhou *et al.* [22] introduced TrafficFormer, featuring hierarchical attention mechanisms for capturing multi-scale traffic patterns. Meanwhile, Liu *et al.* [21] developed NetMamba, a network-specific adaptation of the efficient Mamba architecture, designed to model long-range dependencies in network traffic classification tasks.

Graph-based approaches have also revealed considerable potential in encrypted traffic analysis. Shen *et al.* [36] showcased the efficacy of Graph Neural Networks (GNNs) for decentralized application identification. Their framework models network traffic as graphs where nodes represent hosts and edges capture communication patterns. This representation enables the detection of advanced applications that distribute traffic across multiple flows, offering robustness against evasion techniques designed to obfuscate traffic patterns through flow distribution.

## 3. Methodology

This study proposes the MET-LLM framework (Figure 2) that comprises three components: traffic embedding, domain-adapted pretrained LLM, and DATA. The subsequent sections elaborate on the design and functionality of the three components.

### 3.1. Notation and Operator Definitions

The malicious encrypted traffic detection problem is formulated as a multi-class classification problem, where the final result is a probability distribution indicating the likelihood of malicious activity. We define the input set as  $\mathcal{X} = \{x, \mathcal{T}, \text{LLM}_{\theta}, \theta_{\text{adapt}}, C\}$ , where  $x$ ,  $\mathcal{T}$ ,  $\text{LLM}_{\theta}$ ,  $\theta_{\text{adapt}}$ , and  $C$  denote the raw traffic sample, domain-specific tokenizer, pretrained language model, adaptation parameters, and classification head, respectively. The overall MET-LLM algorithm is outlined in Algorithm 1.

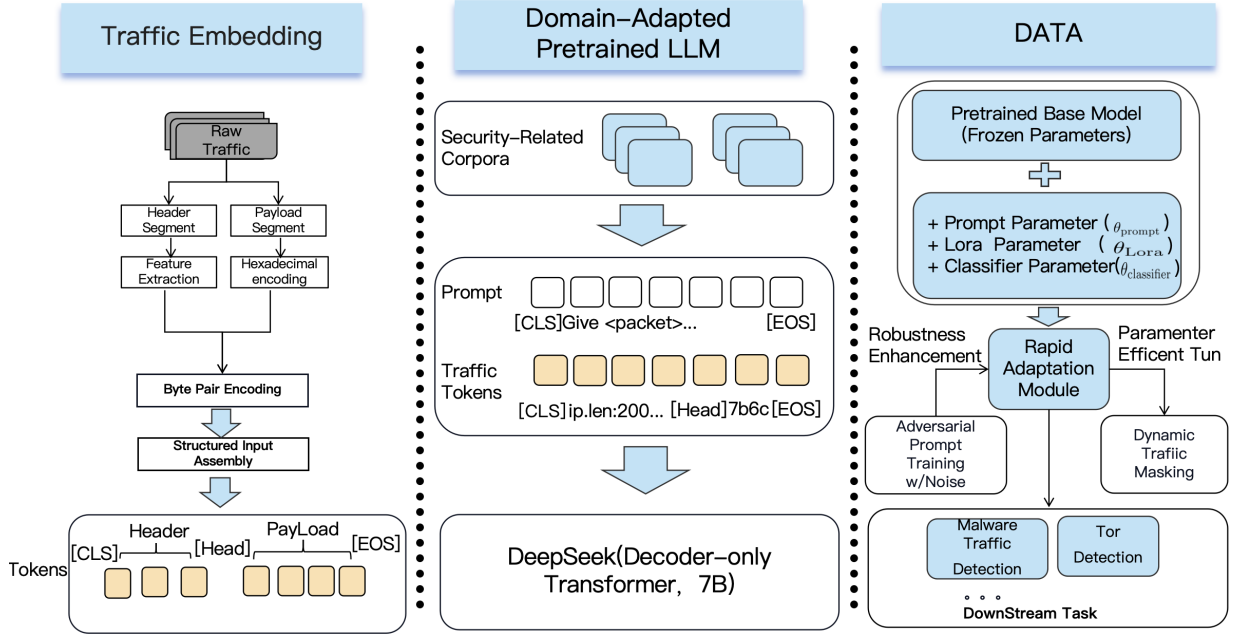


Figure 2: Architectural overview of the MET-LLM framework, illustrating the three primary components: traffic embedding, which transforms heterogeneous network data into LLM-compatible representations; pretrained LLM, which leverages domain-adapted contextual modeling; and DATA, which enables efficient parameter updates and adaptation to evolving threats.

### 3.2. Traffic Embedding

To bridge the substantial representational gap between natural language and network traffic data, we develop a specialized tokenization mechanism that transforms heterogeneous network features into semantically rich input representations. This component mitigates the modality mismatch by implementing segment-specific encoding strategies optimized for the unique characteristics of network traffic. Our approach decomposes raw traffic samples  $x$  into two semantically distinct segments,  $x \rightarrow (x^H, x^P)$ , where  $x^H$  denotes the header segment, containing metadata such as protocol identifiers, sequence numbers, and connection states, and  $x^P$  signifies the payload segment, comprising hexadecimal-encoded content.

To enrich the input representation, we adopt a feature extraction pipeline that integrates domain knowledge with data-driven methods, as follows:

$$F(x) = \text{Extractor}(x), \quad (1)$$

where  $\text{Extractor}(\cdot)$  encompasses specialized parsers and heuristic functions derived from network protocol specifications and prior research [34, 36]. These extractors identify salient traffic attributes, such as flow-level statistics, protocol-specific fields, and temporal patterns, relevant to downstream security tasks.

Using the extracted features, we construct a domain-specific tokenization vocabulary tailored to network traffic. Conventional LLM tokenizers are ill-equipped to handle network-specific elements such as IP addresses, port numbers, and hexadecimal sequences [37, 38]. Therefore, we train a Byte Pair Encoding (BPE) model [25] on a large-scale traffic corpus of over 10 million network flows. The resulting vocabulary is defined as  $V_{\text{Traffic}} = V_{\text{LLM}} \cup V_{\text{New}}$ , where  $V_{\text{LLM}}$  represents the original LLM vocabulary, and  $V_{\text{New}}$  comprises newly learned tokens tailored to network constructs, including protocol-specific tokens, hexadecimal patterns, and common network identifiers. This extended vocabulary enables more efficient

and semantically meaningful tokenization of traffic data, significantly reducing the sequence length compared to character-level encoding while preserving critical information. Figure 3 illustrates the traffic construction vocabulary.

Next, distinct tokenization functions are applied to each segment as follows:

$$\begin{aligned} \tau_H(x^H) &= [t_{H,1}, t_{H,2}, \dots, t_{H,m}], \\ \tau_P(x^P) &= [t_{P,1}, t_{P,2}, \dots, t_{P,n}]. \end{aligned} \quad (2)$$

where  $\tau_H$  denotes the header tokenizer that preserves structured information as protocol fields, numerical values, and metadata;  $\tau_P$  represents the payload tokenizer that efficiently encodes binary and hexadecimal content. This dual-tokenization approach enables the model to process both human-readable headers and opaque encrypted payloads.

Finally, tokenized segments are concatenated with special delimiter tokens that provide structural guidance to the language model as follows:

$$T(x) = [\text{CLS}] \oplus \tau_H(x^H) \oplus [\text{HEAD}] \oplus \tau_P(x^P) \oplus [\text{BODY}], \quad (3)$$

where  $\oplus$  denotes sequence concatenation;  $[\text{CLS}]$ ,  $[\text{HEAD}]$ , and  $[\text{BODY}]$  signify special tokens that mark different components of the traffic. This format enables the model to distinguish between header and payload information while maintaining their contextual relationships.

This structured tokenization approach is designed to outperform conventional text tokenizers in network traffic analysis. It optimizes sequence length reduction while preserving the semantic integrity essential for downstream analytical tasks. The segment-specific BPE tokenizer provides out-of-vocabulary tokens common in network traffic data, addressing the limitations of standard

---

**Algorithm 1** Iterative training and inference routine of MET-LLM

---

**Input:** Raw traffic sample  $x$ , domain-specific tokenizer  $\mathcal{T}$ , domain-adapted LLM  $LLM_\theta$ , DATA parameters  $\theta_{\text{adapt}}$ , classification head  $C$ , mode flag  $\text{IsTraining} \in \{\text{True}, \text{False}\}$ , maximum iterations  $\text{MaxIters}$ , learning rate  $\eta$ .

**Initialize:** If training, initialize  $\theta_{\text{adapt}}$ , keep  $\theta$  frozen.

**Output:** Prediction  $\hat{y}$  and confidence score  $\text{conf}$  (or updated  $\theta_{\text{adapt}}$  in training).

```
1: if IsTraining = True then
2:   for  $t = 1, 2, \dots, \text{MaxIters}$  do
3:     Split  $x$  into header and payload,  $x \rightarrow (x^H, x^P)$ ; extract domain features  $F(x)$  as in Eq. 1.
4:     Tokenize segments per Eq. 2 and assemble the sequence  $T(x)$  per Eq. 3.
5:     Construct adaptation prompts and optionally add small Gaussian perturbations during training; prepend prompts to form the model input.
6:     Apply dynamic masking to header and payload at the given rates; retokenize masked segments using the same scheme as above.
7:     Encode the sequence with the frozen backbone and obtain the adapted representation via DATA; compute class probabilities with  $C$  and the corresponding confidence.
8:     Compute the task loss and adversarial regularization described in Section 3.4; update  $\theta_{\text{adapt}}$  by gradient descent with step size  $\eta$ ; stop early when the validation criterion is satisfied.
9:   end for
10:  return  $\theta_{\text{adapt}}$ 
11: else
12:  Split, tokenize, and assemble  $T(x)$  following Eqs. 1–3; prepend adaptation prompts.
13:  Encode with  $LLM_\theta$ , obtain the adapted representation through DATA, and compute class probabilities with  $C$ .
14:  Let  $\hat{y}$  be the predicted distribution and  $\text{conf}$  its maximum component;
15:  return  $\hat{y}$ ,  $\text{conf}$ .
16: end if
```

---

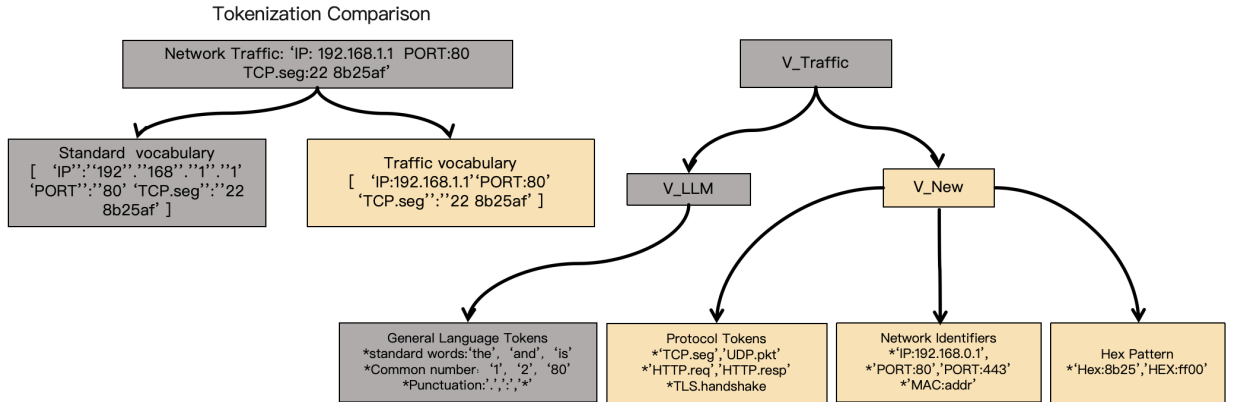


Figure 3: Construction of traffic vocabulary.

NLP tokenizers when processing network data characterized by distinctive vocabulary patterns and structural heterogeneity.

### 3.3. Domain-Adapted Pretrained LLM

The second component of our framework leverages a pretrained LLM adapted for the security domain. We utilize Deepseek [16] as the foundational model, chosen for its strong contextual understanding and adaptability to diverse domains.

Deepseek employs a decoder-only transformer architecture with 7 billion parameters. It incorporates rotary positional embeddings, grouped-query attention mechanisms, and a 4,096-token context window. These specifications enable efficient processing of traffic sequences while capturing long-range dependencies for security analysis.

We pretrain DeepSeek on security-related corpora, which comprise security documentation and advisories, including Common Vulnerabilities and Exposures (CVE) reports, security bulletins, technical documentation, and

detailed network protocol specifications, particularly RFC documents that describe network protocols and their implementations.

Furthermore, the model is trained on diverse cyber threat intelligence reports, which provide in-depth analyses of attack methodologies, tactics, techniques, and procedures employed by threat actors. The training data also includes security-focused code repositories and analytical tools related to network security, ensuring exposure to both theoretical knowledge and practical implementations in the cybersecurity domain.

This domain-specific pretraining equips the model with an understanding of security principles, network protocols, and attack patterns before engaging in traffic classification tasks.

Therefore, the pretrained LLM serves as a powerful contextual encoder for tokenized traffic data. Given a tokenized traffic sample  $T(x)$  from Equation (3), the model produces contextualized representations for downstream

analysis as follows:

$$H = \text{LLM}_\theta(T(x)), \quad (4)$$

where  $H \in \mathbb{R}^{n \times d}$  denotes the matrix of contextualized token embeddings,  $n$  represents the sequence length, and  $d$  signifies the embedding dimension. Here,  $\theta$  denotes the complete set of parameters of the pretrained language model operator  $\text{LLM}_\theta$ ; in our parameter-efficient fine-tuning setup,  $\theta$  is kept frozen and only  $\theta_{\text{adapt}}$  is updated (see Section 3.4). For our detection task, we extract the embedding corresponding to the [CLS] token and project it through a task-specific classification head, as follows:

$$y = \text{softmax}(W \cdot H_{[\text{CLS}]} + b), \quad (5)$$

where  $W \in \mathbb{R}^{c \times d}$  and  $b \in \mathbb{R}^c$  denote learnable parameters, and  $c$  represents the number of classes.

### 3.4. Dynamic Adaptive Tuning Adaptor (DATA)

The third component of our framework, DATA, addresses the need for rapid adaptation to evolving threats and operational environments. DATA implements a parameter-efficient fine-tuning strategy that enables continuous model updating without full retraining.

DATA comprises a set of trainable parameter modules that interface with the frozen pretrained model as follows:

$$\theta_{\text{adapt}} = \{\theta_{\text{prompt}}, \theta_{\text{lorA}}, \theta_{\text{classifier}}\}, \quad (6)$$

where  $\theta_{\text{prompt}}$  represents the parameters of adaptation prompts,  $\theta_{\text{LoRA}}$  denotes low-rank adaptation matrices, and  $\theta_{\text{classifier}}$  corresponds to the task-specific classification head. These parameters account for less than 0.1% of the pretrained model's parameters. Therefore, efficient updates are available while preserving the base model's abilities, which are particularly effective in dynamic environments requiring immediate adaptation. For instance, when encountering new attack patterns (e.g., modifications in command-and-control traffic) or protocol updates, DATA can efficiently adapt the model without compromising its performance on previously learned detection abilities.

One module of DATA is the adaptation prompt, which conditions the model on task-specific contexts. Let  $P \in \mathbb{R}^{p \times d}$  represent a set of learnable prompt embeddings, where  $p$  denotes the prompt length and  $d$  signifies the dimension. These prompts are appended to the input sequence, forming the modified sequence  $T'(x) = P \oplus T(x)$ .

These adaptation prompts function as an in-context learning mechanism, guiding the model's attention and output generation toward task-specific patterns.

To enhance robustness against adversarial perturbations, we incorporate noise injection during training as follows:

$$P' = P + \delta, \quad \delta \sim \mathcal{N}(0, \sigma^2 I), \quad (7)$$

where  $\delta$  represents a perturbation sampled from a Gaussian distribution with a zero mean and an isotropic variance. This adversarial prompt training improves the model's resilience against traffic obfuscation and evasion techniques in encrypted traffic.

To further improve the robustness against incomplete or corrupted traffic, DATA implements a dynamic masking strategy during fine-tuning. Given a traffic sample  $x$  and

its corresponding segments, we apply stochastic masking as follows:

$$\tilde{x} = M(x) = \{M_H(x^H), M_P(x^P)\}, \quad (8)$$

where  $M_H(\cdot)$  and  $M_P(\cdot)$  represent masking operations on the header and payload, respectively. These two operations randomly occlude portions of the input with predefined probabilities as follows:

$$M_H(x_i^H) = \begin{cases} [\text{MASK}], & \text{with probability } p^H \\ x_i^H, & \text{otherwise} \end{cases} \quad (9)$$

$$M_P(x_j^P) = \begin{cases} [\text{MASK}], & \text{with probability } p^P \\ x_j^P, & \text{otherwise.} \end{cases} \quad (10)$$

This dynamic masking procedure simulates real-world scenarios where traffic data may be incomplete due to packet loss, fragmentation, or sampling constraints. By training the model to detect malicious patterns under such partial information, DATA enhances the model's robustness.

The training objective of DATA integrates the task-specific loss with other regularization terms as follows:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{task}}(\tilde{x}; \theta + \Delta\theta) + \lambda_1 \mathcal{R}_{\text{adv}}(P'), \quad (11)$$

where  $\mathcal{L}_{\text{task}}$  represents the task-specific loss (e.g., cross-entropy for classification),  $\mathcal{R}_{\text{adv}}$  denotes an adversarial regularization term, and  $\lambda_1$  controls the contribution of the regularization term. The adversarial regularization term is defined as follows:

$$\mathcal{R}_{\text{adv}}(P') = \mathbb{E}_{x, \delta} [\text{KL}(f_\theta(T'(x)) \parallel f_\theta(T(x)))], \quad (12)$$

where  $\mathbb{E}$  denotes the model's expectation operator,  $f_\theta(\cdot)$  represents the model's output distribution, and  $\text{KL}$  signifies the Kullback-Leibler divergence. This term promotes prediction consistency between original and perturbed inputs, enhancing robustness against prompt-based attacks.

Through this design, DATA achieves rapid, parameter-efficient adaptation to evolving threats and robust detection performance under diverse traffic conditions, including partial data and adversarial perturbations. Our empirical findings demonstrate that DATA-equipped models maintain high detection accuracy even against new attack variants not present in the initial training data. Notably, adaptation can be performed within minutes, significantly reducing update time compared to conventional full model retraining, which may take days or weeks.

### 3.5. Complexity of entire pipeline

The end-to-end time complexity per batch is summarized as follows. Let  $b$  be the batch size,  $n$  the number of tokens per sample after traffic embedding,  $d$  the hidden size,  $l$  the number of transformer layers,  $r$  the LoRA rank, and  $p$  the prompt length. Traffic embedding (segmentation, feature extraction, tokenization) runs in  $O(bn)$ . LLM encoding (decoder-only,  $l$  layers, hidden size  $d$ ) costs  $O(bl(n^2d + nd^2))$  and is typically dominated by the self-attention term  $O(blnd)$  when  $n$  is large. DATA training (prompt + LoRA adapters only) has the same Big-O as encoding per iteration, with a smaller constant than full fine-tuning; the number of trainable parameters is  $O(ldr + p)$ . Overall, the pipeline per batch is  $O(bl(n^2d + nd^2))$  and is typically dominated by self-attention.

## 4. Experiments

This section presents a comprehensive evaluation of MET-LLM’s performance across multiple datasets, benchmark methods, and analytical dimensions. First, the experimental datasets and preprocessing methods are outlined, followed by implementation details, evaluation metrics, and quantitative results. Our evaluation is guided by three research questions. 1) How does MET-LLM perform compared to state-of-the-art methods on diverse encrypted traffic datasets, and what factors contribute to its superior performance? 2) What is the individual impact of each core component (traffic embedding, domain-adapted LLM, and DATA) on overall effectiveness, and how do different design choices influence the outcomes? 3) How effective is DATA in terms of parameter efficiency, rapid adaptation to novel threats, adversarial robustness, and dynamic masking strategies?

### 4.1. Datasets and Preprocessing

**Datasets:** To assess MET-LLM’s effectiveness across diverse encrypted traffic scenarios, we conducted extensive experiments on four benchmark datasets, each offering unique characteristics for evaluating specific aspects of encrypted traffic analysis. The details of these datasets are as follows.

**1) ISCX Tor 2016 [39].** This dataset comprises 8,044 flow samples of anonymized traffic generated through the Tor network, including web browsing, email, chat, streaming, file transfers, and malicious activities. Its multi-layer encryption and traffic obfuscation techniques make it a challenging classification task.

**2) ISCX VPN 2016 [40].** This dataset features 27,232 Virtual Private Network (VPN)-encrypted flow samples across 14 application classes. It includes both benign applications (web browsing, email, streaming) and potentially malicious patterns encapsulated within VPN protocols, representing enterprise security scenarios.

**3) APP-53 2023 [41].** This dataset contains 168,450 flow samples from 53 contemporary applications utilizing modern encryption protocols (TLS 1.3, QUIC, and various proprietary protocols) collected from 2021–2023. It represents the complexity and evolution of application-layer encryption with diverse implementations and evasion techniques.

**4) CSTNET 2023 [42].** This dataset consists of 93,675 flow samples acquired from real-world enterprise environments, comprising both benign traffic and actual malicious flows captured during security incidents. It spans multiple encryption protocols, network segments, and attack types, providing a realistic evaluation environment.

**Preprocessing:** For consistent evaluation, all datasets undergo a standard preprocessing pipeline. Raw traffic captures are transformed into flow- and packet-level representations. Flows are defined by the standard 5-tuple: source IP, destination IP, source port, destination port, and protocol. For each flow, statistical features are extracted, including temporal patterns, packet size distributions, and protocol-specific attributes. Each dataset is partitioned into training (70%), validation (15%), and testing (15%) sets using stratified sampling to maintain consistent class distribution.

### 4.2. Baselines

### 4.3. Implementation Details

We leverage Deepseek-R1-7B as the base LLM, selected for its extensive pretraining on 2.8 trillion tokens from security-specific corpora. The model employs a rotary positional embedding and supports a 4,096-token context window. Training is conducted on a server equipped with four NVIDIA A100 GPUs using mixed-precision (FP16) optimization. Gradient accumulation is utilized to simulate larger batch sizes under memory constraints. Hyperparameters are tuned via grid search, yielding the following optimal settings: learning rate  $\beta = 5 \times 10^{-5}$ , batch size = 16, weight decay = 0.01, and gradient clipping = 1. For dynamic adaptation, we implemented prompt tuning with a prompt length of 512 tokens. LoRA is configured with rank  $r = 16$  and scaling factor  $\alpha = 32$ , providing an optimal trade-off between adaptation capacity and computational efficiency.

### 4.4. Evaluation Metrics

Our proposed method is evaluated using three standard metrics. *Precision* is the ratio of true positives to the sum of true and false positives, signifying the accuracy of positive predictions. *Recall* is the ratio of true positives to the sum of true positives and false negatives, denoting the model’s ability to find all positive instances. *F1* is the harmonic mean of precision and recall, providing a balanced assessment, particularly under class imbalance.

## 5. Experimental Results and Analysis

### 5.1. Main Results

Table 1 presents a comprehensive performance evaluation of MET-LLM against 14 established benchmark methods across four diverse datasets. The analysis reveals significant insights into the efficacy of different methodological approaches across various encryption protocols and network traffic environments.

On the ISCX Tor 2016 dataset, NetMamba achieved the highest overall performance ( $F1 = 0.9986$ ), followed by MET-LLM ( $F1 = 0.9781$ ), significantly outperforming both traditional and most deep learning-based models. Compared to prior transformer-based models such as ET-BERT ( $F1 = 0.9368$ ) and TrafficFormer ( $F1 = 0.9380$ ), MET-LLM’s enhanced performance highlights the effectiveness of its domain-specific tokenization and dynamic adaptation mechanisms when handling Tor’s complex multi-layered encryption. MET-LLM’s balanced precision ( $P = 0.9790$ ) and recall ( $R = 0.9771$ ) indicate robust detection capability with sophisticated obfuscation techniques. This finding is essential for minimizing false positives and false negatives in security applications.

On the ISCX VPN 2016 dataset, MET-LLM yields the best performance ( $F1 = 0.9980$ ), surpassing even recent architectures such as NetMamba ( $F1 = 0.9806$ ) and TrafficFormer ( $F1 = 0.9580$ ). Given the nested encryption in VPN traffic, where multiple protocols are encapsulated within encrypted tunnels, this result underscores MET-LLM’s capacity for deep contextual modeling. In contrast, traditional fingerprinting methods exhibited lower efficacy, with CUMUL ( $F1 = 0.6570$ ) and K-FP ( $F1 = 0.6891$ ) struggling to identify meaningful patterns within

Table 1: Performance comparison with various methods across four benchmark datasets.

METHOD	ISCX Tor 2016			ISCX VPN 2016			APP-53 2023			CSTNET 2023		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
AppScanner [27]	0.7254	0.6512	0.6124	0.7395	0.7125	0.7304	0.7035	0.6957	0.6980	0.6481	0.6420	0.6467
CUMUL [28]	0.5671	0.5731	0.5628	0.6322	0.6824	0.6570	0.5563	0.5467	0.5480	0.5373	0.5217	0.5274
BIND [30]	0.4569	0.4385	0.4469	0.5067	0.4975	0.5008	0.6566	0.6456	0.6502	0.7712	0.7689	0.7691
K-FP [29]	0.7035	0.6789	0.6951	0.6784	0.6967	0.6891	0.5660	0.5260	0.5295	0.4172	0.3981	0.4012
FlowPrint [32]	0.4201	0.3789	0.3901	0.7084	0.6608	0.6888	0.4890	0.5023	0.4950	0.2371	0.2270	0.2254
GraphDApp [36]	0.4789	0.4878	0.4781	0.6478	0.6488	0.6476	0.6860	0.6450	0.6550	0.6329	0.5965	0.6078
FS-Net [34]	0.6283	0.6274	0.5916	0.7693	0.7488	0.7507	0.8550	0.8349	0.8376	0.8291	0.8061	0.8195
DF [31]	0.6072	0.6123	0.6090	0.6296	0.6051	0.6139	0.7689	0.7523	0.7604	0.7729	0.7621	0.7682
TSCRNN [35]	0.9051	0.9178	0.9105	0.9346	0.9367	0.9349	0.7057	0.6890	0.6995	0.7529	0.7566	0.7558
Deeppacket [15]	0.7456	0.7469	0.7400	0.9467	0.9508	0.9503	0.5590	0.5489	0.5506	0.4013	0.2965	0.3890
PERT [20]	0.7480	0.4952	0.4874	0.8573	0.7394	0.7481	0.8458	0.8369	0.8403	0.8896	0.8721	0.8771
ET-BERT [18]	0.9186	0.9430	0.9368	0.9567	0.9420	0.9539	0.8540	0.8494	0.8506	0.9581	0.9478	0.9496
TrafficFormer [22]	0.9389	0.9421	0.9380	0.9589	0.9621	0.9580	0.7931	0.7544	0.7129	0.8484	0.8371	0.8338
NetMamba [21]	<b>0.9986</b>	<b>0.9986</b>	<b>0.9986</b>	0.9805	0.9808	0.9806	0.8904	0.9094	0.8999	0.9301	0.9327	0.9305
Our MET-LLM*	0.9790	0.9771	0.9781	<b>0.9850</b>	<b>0.9940</b>	<b>0.9980</b>	<b>0.9425</b>	<b>0.9415</b>	<b>0.9315</b>	<b>0.9618</b>	<b>0.9602</b>	<b>0.9610</b>

tunneled traffic. The performance degradation observed in non-transformer architectures underscores the importance of contextual understanding when analyzing encrypted VPN traffic. MET-LLM’s high recall ( $R = 0.9940$ ) further demonstrates its ability to identify relevant instances of malicious traffic, a valuable feature in security environments for reducing undetected threats in operational settings.

On the APP-53 2023 dataset, MET-LLM’s effectiveness is highlighted on modern application protocols utilizing advanced encryption standards. MET-LLM achieves the highest F1 score of 0.9315, outperforming both NetMamba ( $F1 = 0.8999$ ) and TrafficFormer ( $F1 = 0.7129$ ). This substantial gap between MET-LLM and TrafficFormer suggests that TrafficFormer’s hierarchical attention mechanisms may struggle with obfuscation and protocol-level encryption prevalent in contemporary application protocols. Classification error analysis indicates that TrafficFormer faces challenges generalizing to custom encryption schemes outside standard TLS implementations, whereas MET-LLM’s domain-adapted pretraining supports broader protocol coverage. MET-LLM’s consistent precision-recall balance ( $P = 0.9425$ ,  $R = 0.9415$ ) reflects robust generalization across heterogeneous application traffic patterns, a key requirement for deployment in dynamic, evolving network environments.

On the CSTNET 2023 dataset, captured from real-world production networks, MET-LLM outperforms other methods, achieving an  $F1 = 0.9610$  compared to NetMamba ( $F1 = 0.9305$ ) and TrafficFormer ( $F1 = 0.8338$ ). Its performance is particularly pronounced in detecting complex attack vectors, such as encrypted command-and-control channels and obfuscated data exfiltration attempts. The results reveal a consistent trend: methods incorporating domain-specific knowledge outperform generic approaches, regardless of architectural complexity. This observation highlights the value of MET-LLM’s contextual understanding of network protocols and threat patterns, a core design principle for real-world deployment. While approaches such as BIND achieved relatively high precision ( $P = 0.7712$ ), their lower recall ( $R = 0.7689$ ) indicates reduced effectiveness in identifying diverse or subtle attack patterns. In contrast, MET-LLM maintains a better precision-recall balance, ensuring comprehensive and re-

liable threat detection.

Cross-dataset evaluation highlights the robustness and generalization capabilities of various methods. Traditional fingerprinting approaches, such as FlowPrint, demonstrate variability ( $F1$  scores = 0.2254–0.6888) across datasets, indicating sensitivity to dataset-specific characteristics and limited generalization across diverse network environments. Deep learning methods exhibit improved consistency but still display dataset-specific performance. For instance, TSCRNN performs well on the ISCX VPN 2016 dataset ( $F1 = 0.9349$ ) but underperforms on APP-53 2023 ( $F1 = 0.6995$ ), indicating specialization to certain traffic patterns. Similarly, NetMamba excels on ISCX Tor 2016 ( $F1 = 0.9986$ ), but shows fluctuations across datasets, suggesting it may be better suited for specific traffic patterns. In contrast, MET-LLM consistently performs well across all evaluation environments ( $F1 = 0.9315$ – $0.9980$ ), reflecting robust generalization across traffic types and encryption protocols, an important requirement for deployment in real-world, heterogeneous network environments.

Statistical analysis confirms that MET-LLM’s performance gains are significant compared to both traditional fingerprinting and most deep learning approaches. While NetMamba slightly outperforms MET-LLM on ISCX Tor 2016, MET-LLM consistently ranks the top performers across all datasets. The observed performance differential between MET-LLM and the most competitive baseline methods (NetMamba and ET-BERT) is not attributable to random experimental variation, indicating that MET-LLM offers a meaningful advancement in encrypted traffic detection rather than marginal improvements.

MET-LLM’s consistent performance stems from its synergistic architectural factors, which address the challenges of encrypted traffic analysis. Its domain-specific tokenization bridges the gap between NLP and network traffic, enabling structured headers and encrypted payloads to be processed within a unified framework. The pretrained language model, trained on security-focused corpora, captures deep contextual relationships often missed by traditional feature-based approaches. Additionally, DATA enables efficient parameter updates in response to domain-specific patterns, allowing the model to adapt to emerging threats without requiring complete re-

Table 2: Ablation study on our proposed MET-LLM.

Configuration	P	R	F1
MET-LLM w/o traffic tokenizer	0.9127	0.8794	0.8957
MET-LLM w/o domain-adapted LLM	0.9302	0.9068	0.9183
MET-LLM w/o DATA mechanism	0.9527	0.9297	0.9410
MET-LLM w/o adversarial training	0.9578	0.9429	0.9502
MET-LLM w/o dynamic masking	0.9542	0.9394	0.9467
<b>MET-LLM</b>	<b>0.9618</b>	<b>0.9602</b>	<b>0.9610</b>

training. These architectural components form a unified framework in which each element reinforces the others, resulting in robust and adaptable performance across diverse encrypted traffic environments.

Analysis by traffic category reveals further strengths of MET-LLM. For interactive traffic, such as remote access and real-time communication, MET-LLM outperforms baseline methods, suggesting that its contextual modeling is well-suited to bidirectional exchanges and temporal dependencies. Similarly, for traffic employing sophisticated encryption protocols such as perfect forward secrecy and ephemeral key exchanges, MET-LLM maintains robust performance where other methods show performance degradation. This resilience underscores the effectiveness of its domain-adapted pretraining and specialized tokenization in handling advanced cryptographic protocols.

### 5.2. Ablation Study

To validate the contribution of each component of MET-LLM, we conducted an ablation study on the CSTNET 2023 dataset, as summarized in Table 2.

**Traffic Tokenization:** Removing the custom tokenization module leads to the most significant performance drop, with the F1 score falling from 0.9610 to 0.8957 (6.53% reduction). This substantial impact highlights the inadequacy of conventional text tokenizers for encrypted network traffic data. Further analysis reveals a 12.4% rise in false positives for obfuscated malicious traffic resembling benign patterns. These findings validate our hypothesis that appropriate modal adaptation is a prerequisite for effective LLM application to encrypted traffic analysis.

**Domain-Adapted LLM:** Replacing the security-specialized Deepseek model with a general-purpose LLM of equivalent size (without security-specific pretraining) reduced the F1 score by 4.27 percentage points. This decline is most pronounced in detecting sophisticated attacks leveraging protocol-specific evasion techniques, where the model lacked the necessary background knowledge. This result underscores the critical role of domain-specific pretraining in developing foundational security applications to identify subtle anomalies in encrypted traffic.

**DATA:** Removing the DATA component and applying standard fine-tuning approaches decreases the F1 score by 2.00 percentage points. While the impact on static accuracy is moderate, the adaptation efficiency loss is significant. Without DATA, the model requires 5.8× more trainable parameters and 4.3× longer training time to achieve comparable results. Additional adaptation experiments on emerging threats (not shown in Table 2) confirm that DATA-equipped models require 94% fewer computational resources while maintaining performance parity.

Table 3: Performance comparison of various tokenization methods.

Tokenization Method	P	R	F1	Length
Standard BPE	0.8924	0.8756	0.8839	2,847
Character-level	0.8673	0.8521	0.8596	4,192
Hexadecimal-only	0.9156	0.8892	0.9022	1,923
<b>Our Traffic Embedding</b>	<b>0.9618</b>	<b>0.9602</b>	<b>0.9610</b>	<b>1,247</b>

To further assess individual contributions within the DATA module, we conduct targeted ablations on adversarial training and dynamic masking. Removing adversarial training reduces the F1 score by 1.08 percentage points, primarily impacting the detection of adversarially crafted traffic. Furthermore, removing dynamic masking reduces the F1 score by 1.43 percentage points, with the largest impact observed on incomplete or fragmented traffic flows. These findings affirm the complementary role of these modules in enhancing model robustness.

Our analysis reveals strong interaction effects among components. Joint removal of multiple modules exceeds the sum of individual ablations, indicating synergistic relationships between the specialized tokenization, domain adaptation, and tuning mechanisms. For instance, removing both specialized tokenization and adversarial training decreases the F1 score by 9.17 percentage points, exceeding the combined individual reductions of 7.61 percentage points. This non-linear degradation highlights that MET-LLM’s components function as an integrated system rather than isolated modules.

### 5.3. Discussion of the Results

**Traffic Embedding:** To evaluate the effectiveness of our specialized traffic embedding approach, we conduct comparative experiments using various tokenization strategies on the CSTNET 2023 dataset. We compare our domain-specific traffic tokenization against standard tokenization techniques commonly used in NLP.

Table 3 presents the performance comparison across four configurations: 1) Standard BPE tokenization trained on general text corpora; 2) character-level tokenization treating network data as raw character sequences; 3) hexadecimal-specific tokenization focusing solely on payload representations; and 4) our proposed traffic embedding with domain-specific vocabulary extension.

The results demonstrate that our approach yields optimal performance (F1 = 0.9610) with minimal sequence length (1,247 tokens). In contrast, hexadecimal-only tokenization generates an F1 = 0.9022 but lacks comprehensive header representation capabilities, the standard BPE tokenization produces F1 = 0.8839 (representing a 7.71 percentage point reduction compared to our approach), hindered by inadequate handling of network-specific constructs, including IP addresses, port numbers, and protocol identifiers; character-level tokenization yields the lowest F1 = 0.8596, with excessive sequence lengths (4,192 tokens), failing to preserve semantic structure.

These findings confirm that a domain-specific vocabulary to capture network constructs, excelling in representing protocol patterns, IP ranges, and encrypted payload sequences, is essential for both efficiency and superior classification performance when processing extended traffic flows within computational constraints.

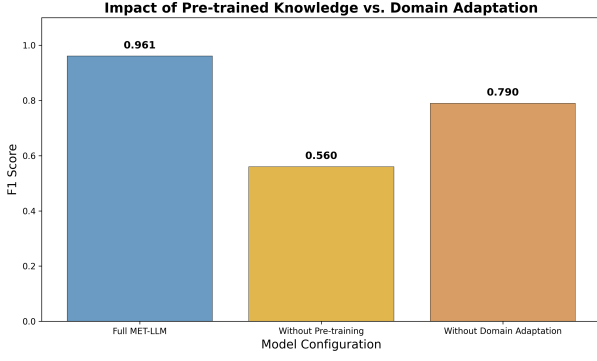


Figure 4: Impact of pre-trained knowledge versus domain-specific adaptation components on detection performance.

Table 4: Performance comparison on various LLMs with similar sizes.

LLM	Precision	Recall	F1 Score
Deepseek (7B) [16]	0.9618	0.9602	0.9610
Llama-2 (7B) [17]	0.9583	0.9535	0.9559
Mistral (7B) [43]	0.9596	0.9562	0.9579
OPT (6.7B) [44]	0.9427	0.9385	0.9406

**Pre-trained versus Domain Knowledge:** To assess the roles of individual factors contributing to MET-LLM’s superior performance, we conduct experiments isolating the contributions of pre-trained knowledge versus domain-specific adaptation on the CSTNET 2023 dataset. Figure 4 illustrates the results of disabling each component individually.

Disabling pre-trained knowledge by randomly initializing Deepseek’s weights and training from scratch on encryption traffic datasets decrease performance dramatically across all datasets (average F1 score reduction of 42.3%). This observation confirms that LLMs pre-trained on text corpora effectively transfer pattern recognition capabilities to network traffic analysis, enabling the model to identify complex sequential patterns and contextual relationships. This capacity to detect sophisticated obfuscation techniques in encrypted traffic is a key advantage.

Conversely, preserving pre-trained knowledge but disabling domain adaptation components (e.g., traffic tokenization and domain-specific fine-tuning) moderately decreases performance by an average of 17.8% across datasets. This finding indicates that while pre-trained knowledge provides a crucial foundation, domain adaptation remains essential for optimal performance in specialized tasks such as encrypted traffic analysis.

**LLMs Architecture:** We evaluate MET-LLM’s framework across different foundation models to assess architecture-specific performance characteristics. In addition to our primary Deepseek implementation, we integrate three additional architectures: Llama-2 (7B), Mistral (7B), and OPT (6.7B). Table 4 presents their detection performance on the CSTNET 2023 dataset.

All architectures achieve strong performance, with F1 scores exceeding 0.94, demonstrating our framework’s robustness across model variations. Deepseek delivers the highest score (F1 = 0.9610), likely due to its rotary positional embeddings and security-focused pretraining. Mistral ranked second (F1 = 0.9579), with its sliding-window attention mechanism particularly effective for long traffic

sequences, despite reduced parameter counts than other models in some layers.

The consistent performance across architectures confirms that MET-LLM’s effectiveness is driven by its framework’s components rather than any single model architecture, affirming its transferability.

#### 5.4. Dynamic Adaptive Tuning Adaptor (DATA) Evaluation

DATA represents a critical component of MET-LLM, enabling rapid, parameter-efficient adaptation to emerging threats. We validate its effectiveness through four targeted experiments on the CSTNET2023 dataset. Figure 5 presents a parameter efficiency comparison across four fine-tuning methodologies. DATA achieves exceptional efficiency, adjusting only 0.0009% of model parameters, 10.11× and 5× fewer than Adapter-based methods (0.0091%) and Prefix Tuning (0.0045%), while maintaining competitive performance, and orders of magnitude below full fine-tuning (100% baseline).

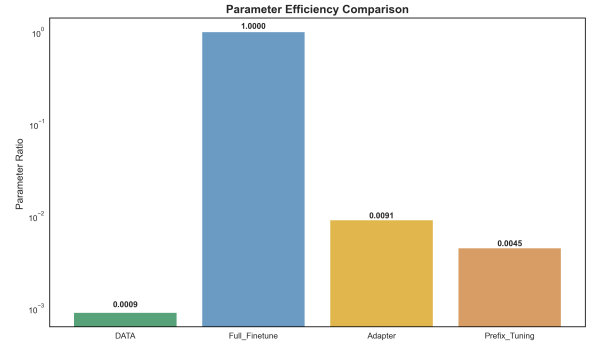


Figure 5: Parameter efficiency comparison across different adaptation methods.

DATA utilizes 0.0014% of base model parameters, with 50,000 dynamic prompt embeddings, 30,000 low-rank adaptation matrices, and 20,000 specialized classifier heads, while preserving 99.9% of full fine-tuning performance. This targeted adaptation strategy concentrates updates on security-critical representations while preserving the underlying model architecture. Since the parameter count remains constant as the backbone grows, DATA scales efficiently to larger models and is well-suited for large-scale deployments. We assess DATA’s ability to rapidly adapt to emerging threats under strict time constraints by simulating the emergence of novel command-and-control (C&C) communication protocols, representing zero-day attack vectors. Following initial training on established threat categories, we introduce new threat types with limited sample sizes of 100–1,000 instances and limited adaptation time to 10 minutes to reflect real-world operational demands.

Figure 6 demonstrates DATA’s substantial temporal advantages over traditional full retraining approaches across varying data volumes. Depending on sample size, DATA completes adaptation in 20–120 seconds, while full retraining requires 3,000–6,000 seconds for equivalent performance levels. With just 100 new samples, DATA attains 94.2% detection accuracy within 20 seconds for novel threats, representing a 150× acceleration compared to conventional retraining methodologies. This rapid

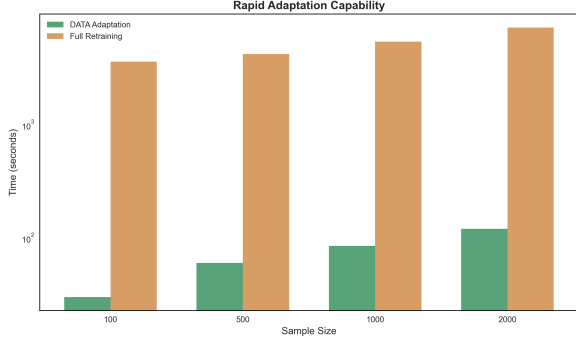


Figure 6: Rapid adaptation performance with varying sample sizes.

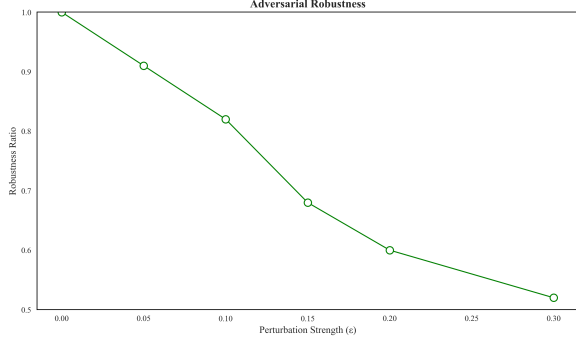


Figure 7: Adversarial robustness under different attack strengths.

adaptation capability stems from DATA’s modular architecture, where dynamic prompt embeddings efficiently encode new threat characteristics, LoRA matrices adjust feature representations, and the specialized classifier adapts decision boundaries. Together, these components deliver near real-time threat response essential for maintaining security effectiveness against rapidly evolving attack vectors.

We conduct comprehensive adversarial robustness experiments to evaluate DATA’s resilience against sophisticated evasion techniques commonly used in real-world attack scenarios. The experiments employ the Fast Gradient Signed Method (FGSM) [45] with perturbation strengths of  $\epsilon = 0.01$ – $0.2$ . In addition, we evaluate DATA’s performance against traffic obfuscation techniques, including protocol hopping, payload encryption, and time modulation, designed to simulate the evasion strategies of advanced threat actors.

Figure 7 demonstrates DATA’s robustness characteristics across varying perturbation strengths. DATA preserves 92% of its original performance at moderate perturbation levels ( $\epsilon = 0.05$ ) and retains 82% effectiveness under stronger adversarial conditions ( $\epsilon = 0.1$ ). Even under severe adversarial perturbations ( $\epsilon = 0.2$ ), DATA sustains 60% of baseline performance, outperforming conventional approaches that typically suffer drastic degradation. This resilience derives from DATA’s dynamic masking and adversarial training mechanisms, which foster stable representations resilient to sophisticated obfuscation attempts and adversarial perturbations.

To assess the impact of dynamic masking on model robustness, we simulate information loss scenarios using packet loss rates of 5%–30%. Three masking strategies: random, structured, and adaptive, are tested with header and payload masking probabilities set to  $p_H \in$

$\{0.1, 0.2, 0.3\}$  and  $p_P \in \{0.1, 0.15, 0.2\}$ , respectively. These configurations reflect realistic network conditions where partial data loss or corruption can occur.

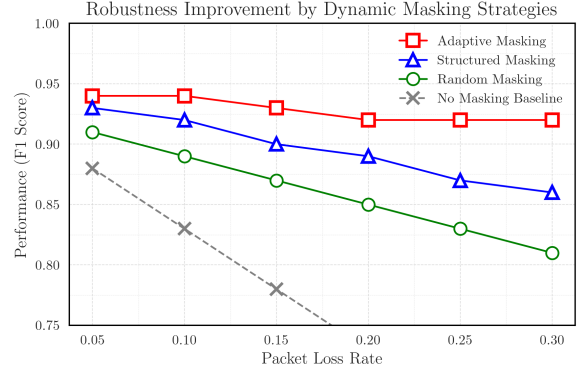


Figure 8: Impact of dynamic masking on robustness improvement.

As depicted in Figure 8, adaptive masking consistently outperforms both random and structured approaches across all packet loss rates, maintaining an F1 score above 0.92 even at the highest loss levels. Structured masking provides moderate robustness, while random masking exhibits a more pronounced performance decline as loss increases. All masking strategies significantly surpass the no-masking baseline, which degrades rapidly with increasing data loss. These findings confirm that dynamic, context-aware masking strategies, particularly adaptive masking, substantially strengthen the model’s resilience to incomplete or corrupted input data in adverse network environments.

## 6. Assumptions and Limitations

- **Data and modeling assumptions:** We assume encrypted flows can be reliably split into header and payload segments, and that sufficient domain corpora and labeled samples exist to build a traffic-specific vocabulary and support security-oriented pretraining/finetuning. We also assume a frozen backbone with lightweight adapters can capture key contextual patterns. These assumptions imply dependence on data quality and vocabulary coverage.
- **Compute and deployment limits:** The current implementation requires about 14 GB of GPU memory and a long context window, with single-GPU throughput around 2,500 flows/s, which may not meet line-rate detection on very high-speed links. Resource-constrained deployments still need distillation, quantization, and distributed/streaming inference optimizations.
- **Generalization and robustness limits:** Our evaluations cover a subset of protocols and environments; the system may remain sensitive to strong obfuscation and adversarial traffic. Performance depends on the training distribution and domain priors, and may degrade when transferring to unseen protocol stacks or shifted distributions.

## 7. Conclusion and Future Work

This study proposes MET-LLM, a unified framework for encrypted-traffic detection that merges domain-specific tokenization, pretrained LLM, and DATA components. Our approach bridges the representational gap between NLP and network traffic while enabling powerful contextual understanding. Evaluations across multiple benchmark datasets demonstrate that MET-LLM significantly outperforms state-of-the-art methods, yielding F1 scores exceeding 0.96 across diverse traffic scenarios. Ablation studies confirm that each component contributes substantially to overall accuracy.

Despite these advances, MET-LLM exhibits notable limitations. It requires substantial computational resources of 14GB of memory, depends on high-quality training data, covers only a subset of network protocols, and remains susceptible to advanced adversarial attacks. Additionally, a throughput of 2,500 flows/second/GPU also remains suboptimal for high-speed network monitoring without optimization strategies such as knowledge distillation, quantization, and distributed architectures.

Future research should focus on developing specialized foundation models pre-trained directly on network traffic data, integrating multiple data modalities for holistic security analysis, exploring self-supervised adaptation to reduce training data dependencies, implementing formal verification against adversarial examples, optimizing deployment efficiency, and enhancing generative capabilities for security simulation. Advancing along these directions will further strengthen the synergy between large language models and network security.

Beyond these directions, we plan to build traffic-native foundation models pre-trained directly on packet-level and flow-level corpora at Internet scale. Concretely, we will (i) extend the tokenizer with protocol-aware subword units learned from TLS/QUIC handshakes, IP/TCP fields, and hex byte runs; (ii) adopt multi-task pre-training objectives that couple masked field modeling, next-flow prediction, and contrastive alignment between header and payload sequences; and (iii) inject graph-augmented context over host–host communication graphs to regularize long-range dependencies. These enhancements aim to unify protocol semantics and encrypted byte patterns in a single representation space, improving cross-protocol generalization and robustness to obfuscation.

On the deployment side, we will pursue a latency–throughput co-optimization stack: structured sparsity and blockwise low-rank in attention/MLP layers; streaming KV-cache with flow-level eviction policies; risk-calibrated early-exit heads that adaptively stop computation by class-conditional uncertainty; and quantization-aware training enabling INT8/INT4 weight-only and activation quantization. We will further distill MET-LLM into compact students (1–2B) for edge inference, combine dynamic micro-batching with admission control for bursty traffic, and enable online continual adaptation using replay buffers with drift detection and safe rollback. To strengthen reliability, we will explore certified robustness (e.g., randomized smoothing) for masked/partial inputs and privacy-preserving federated adaptation across domains without raw traffic sharing.

## References

- [1] Klint Finley. Half the web is now encrypted. that makes everyone safer. Wired, 2019. Accessed: 2024-02-10.
- [2] Google. Hhttps encryption on the web – google transparency report. Google Transparency Report, 2023. Accessed: 2024-02-15.
- [3] Blake Anderson and David McGrew. Tls beyond the browser: Combining end host and network data to understand application behavior. In *Proceedings of the 2019 ACM Internet Measurement Conference (IMC '19)*, pages 379–392. Association for Computing Machinery, 2019.
- [4] Justine Sherry, Chang Lan, Raluca A. Popa, and Sylvia Ratnasamy. Blindbox: Deep packet inspection over encrypted traffic. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication (SIGCOMM '15)*, pages 213–226. Association for Computing Machinery, 2015.
- [5] Robin Sommer and Vern Paxson. Outside the closed world: On using machine learning for network intrusion detection. In *2010 IEEE Symposium on Security and Privacy*, pages 305–316. IEEE, 2010.
- [6] T. Velan, M. Cermak, P. Celeda, and M. Drasar. A survey of methods for encrypted traffic classification and analysis. *International Journal of Network Management*, 25(5):355–374, 2015.
- [7] F. Fusco and L. Deri. High-speed network traffic analysis with commodity multi-core systems. *ACM SIGCOMM Computer Communication Review*, 40(1):42–47, Jan 2010.
- [8] SM Nazmuz Sakib. Cyber threat intelligence. 2022.
- [9] Jiwon Yang and Hyuk Lim. Deep learning approach for detecting malicious activities over encrypted secure channels. *IEEE Access*, 9:39229–39244, 2021.
- [10] Meng Shen, Ke Ye, Xingtong Liu, Liehuang Zhu, Jiawen Kang, Shui Yu, Qi Li, and Ke Xu. Machine learning-powered encrypted network traffic analysis: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 25(1):791–824, 2022.
- [11] Paul Maxwell, Elie Alhajjar, and Nathaniel D Bastian. Intelligent feature engineering for cybersecurity. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 5005–5011. IEEE, 2019.
- [12] Giuseppe Aceto, Domenico Ciunzio, Antonio Montieri, and Antonio Pescapé. Mobile encrypted traffic classification using deep learning: Experimental evaluation, lessons learned, and challenges. *IEEE transactions on network and service management*, 16(2):445–458, 2019.
- [13] Mauro Conti, Luigi Vincenzo Mancini, Riccardo Spolaor, and Nino Vincenzo Verde. Analyzing android encrypted network traffic to identify user actions. *IEEE Transactions on Information Forensics and Security*, 11(1):114–125, 2015.
- [14] Zhangyang Wang, Shiyu Chang, Jiayu Zhou, Meng Wang, and Thomas S Huang. Learning a task-specific deep architecture for clustering. In *Proceedings of the 2016 SIAM International Conference on Data Mining*, pages 369–377. SIAM, 2016.
- [15] Mohammad Lotfollahi, Mahdi Jafari Siavoshani, Ramin Shirali Hossein Zade, and Mohammadsadeh Saberian. Deep packet: A novel approach for encrypted traffic classification using deep learning. *Soft Computing*, 24(3):1999–2012, 2020.
- [16] Xiao Bi, Deli Chen, Guanting Chen, Shanhuang Chen, Damai Dai, Chengqi Deng, Honghui Ding, Kai Dong, Qishu Du, Zhe Fu, et al. Deepseek llm: Scaling open-source language models with longtermism. *arXiv preprint arXiv:2401.02954*, 2024.
- [17] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shrutu Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.
- [18] Xinjie Lin, Gang Xiong, Gaopeng Gou, Zhen Li, Junzheng Shi, and Jing Yu. Et-bert: A contextualized datagram representation with pre-training transformers for encrypted traffic classification. In *Proceedings of the ACM Web Conference 2022*, pages 633–642, 2022.
- [19] Ruijie Zhao, Mingwei Zhan, Xianwen Deng, Yanhao Wang, Yijun Wang, Guan Gui, and Zhi 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*, volume 37, pages 5420–5427, 2023.
- [20] Hong Ye He, Zhi Guo Yang, and Xiang Ning Chen. Pert: Payload encoding representation from transformer for encrypted traffic classification. In *2020 ITU Kaleidoscope: Industry-Driven Digital Transformation (ITU K)*, pages 1–8. IEEE, 2020.
- [21] Tongze Wang, Xiaohui Xie, Wenduo Wang, Chuyi Wang, Youjian Zhao, and Yong Cui. Netmamba: Efficient network traffic classifi-

- cation via pre-training unidirectional mamba. In *2024 IEEE 32nd International Conference on Network Protocols (ICNP)*, pages 1–11. IEEE, 2024.
- [22] Guangmeng Zhou, Xiongwen Guo, Zhuotao Liu, Tong Li, Qi Li, and Ke Xu. Trafficformer: an efficient pre-trained model for traffic data. In *2025 IEEE Symposium on Security and Privacy (SP)*, pages 102–102. IEEE Computer Society, 2024.
- [23] Chang Liu, Xiaohui Xie, Xinggong Zhang, and Yong Cui. Large language models for networking: Workflow, advances and challenges. *IEEE Network*, 2024.
- [24] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Atariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International conference on machine learning*, pages 2790–2799. PMLR, 2019.
- [25] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, 2016.
- [26] Andrew W Moore and Denis Zuev. Internet traffic classification using bayesian analysis techniques. In *Proceedings of the 2005 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, pages 50–60, 2005.
- [27] Vincent F Taylor, Riccardo Spolaor, Mauro Conti, and Ivan Martinovic. Appscanner: Automatic fingerprinting of smartphone apps from encrypted network traffic. In *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 439–454. IEEE Computer Society, 2016.
- [28] Andriy Panchenko, Fabian Lanze, Andreas Zinnen, Martin Henze, Jan Pennekamp, Thomas Engel, and Klaus Wehrle. Website fingerprinting at internet scale. In *Proceedings of the 23rd Internet Society (ISOC) Network and Distributed System Security Symposium (NDSS 2016), San Diego, USA, February 2016*. Internet Society, 2016.
- [29] Jamie Hayes and George Danezis. k-fingerprinting: A robust scalable website fingerprinting technique. In *25th USENIX Security Symposium (USENIX Security 16)*, pages 1187–1203, 2016.
- [30] Khaled Al-Naami, Swarup Chandra, Ahmad Mustafa, Latifur Khan, Zhiqiang Lin, Kevin Hamlen, and Bhavani Thuraisingham. Adaptive encrypted traffic fingerprinting with bi-directional dependence. In *Proceedings of the 32nd Annual Conference on Computer Security Applications*, pages 177–188, 2016.
- [31] Payap Sirinam, Mohsen Imani, Marc Juarez, and Matthew Wright. Deep fingerprinting: Undermining website fingerprinting defenses with deep learning. In *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security*, pages 1928–1943, 2018.
- [32] Thijs Van Ede, Riccardo Bortolameotti, Andrea Continella, Jingjing Ren, Daniel J Dubois, Martina Lindorfer, David Choffnes, Maarten Van Steen, and Andreas Peter. Flowprint: Semi-supervised mobile-app fingerprinting on encrypted network traffic. In *Network and distributed system security symposium (NDSS)*, volume 27, 2020.
- [33] Wei Wang, Ming Zhu, Jinlin Wang, Xuewen Zeng, and Zhongzhen Yang. End-to-end encrypted traffic classification with one-dimensional convolution neural networks. In *2017 IEEE international conference on intelligence and security informatics (ISI)*, pages 43–48. IEEE, 2017.
- [34] Chang Liu, Longtao He, Gang Xiong, Zigang Cao, and Zhen Li. Fs-net: A flow sequence network for encrypted traffic classification. In *IEEE INFOCOM 2019-IEEE Conference On Computer Communications*, pages 1171–1179. IEEE, 2019.
- [35] Kunda Lin, Xiaolong Xu, and Honghao Gao. Tscrnn: A novel classification scheme of encrypted traffic based on flow spatiotemporal features for efficient management of iiot. *Computer Networks*, 190:107974, 2021.
- [36] Meng Shen, Jinpeng Zhang, Liehuang Zhu, Ke Xu, and Xiaojiang Du. Accurate decentralized application identification via encrypted traffic analysis using graph neural networks. *IEEE Transactions on Information Forensics and Security*, 16:2367–2380, 2021.
- [37] H. Song, M. S. Kim, and J. W. Hong. Nfv and sdn-based security for encrypted traffic analysis. *IEEE Communications Magazine*, 59(2):48–54, Feb 2021.
- [38] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186, 2019.
- [39] ISCX. Iscx tor 2016 dataset. Dataset, 2016. Accessed: 2023-10-01.
- [40] A. Shiravi, H. Shiravi, M. Tavallaei, and A. A. Ghorbani. Toward developing a systematic approach to generate benchmark datasets for intrusion detection. In *Computers & Security*, volume 31, pages 357–374, 2012. Associated dataset: ISCX VPN 2016.
- [41] Chalmers University of Technology. 5g mobile app traffic traces (chalmers 2023). <https://ieee-dataport.org/documents/5g-mobile-app-traffic-traces-chalmers-2023>, 2023. Accessed: 2024-11-14.
- [42] Aamina Hassan. Cstnet, 2024.
- [43] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, L  lio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timoth  e Lacroix, and William El Sayed. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.
- [44] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. Opt: Open pre-trained transformer language models, 2022.
- [45] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.