PTU: Pre-trained Model for Network Traffic Understanding

Lingfeng Peng[‡], Xiaohui Xie[‡*], Sijiang Huang[‡], Ziyi Wang[†], Yong Cui[‡*]
[‡]Department of Computer Science and Technology, Tsinghua University, Beijing, China
[†]School of Computer Science, Beijing University of Posts and Telecommunications, Beijing, China

***Abstract*—Network traffic understanding is crucial to providing high-quality network services and protecting network security. However, due to the growing complexity of networks and the rising proportion of encrypted traffic, existing methods for network traffic understanding face severe challenges. Traditional approaches rely on manually designed features or require a large amount of labeled data, while pre-trained models offer new possibilities. Nevertheless, existing pre-trained models have the following limitations: (1) Their inputs only contain features from the packet content, neglecting temporal information about network dynamics. (2) Their pre-training targets only focus on static characteristics of the data stream without understanding the process of the network transmission. This paper presents the Pre-trained model for network Traffic Understanding (PTU), an innovative model that employs self-supervised pre-training to address the challenges of network traffic understanding. In PTU, we design a traffic representation scheme that integrates static packet content and network dynamics into a unified input space. Furthermore, we propose a pre-training method that includes four tailored pre-training targets. This approach enables PTU to capture both static and dynamic characteristics of network traffic from massive amounts of unlabeled data, thereby achieving enhanced performance in downstream tasks through fine-tuning. Extensive experiments confirm PTU's state-of-the-art (SOTA) performance. In traffic classification tasks, PTU achieves an F1 score of over 0.99 and secures a more than 10% improvement in accuracy in the most challenging task of encrypted application classification.**

***Index Terms*—Network Traffic Understanding, Pre-trained Model, Traffic Representation, Pre-training Target.**

I. Introduction

Network traffic understanding refers to the systematic analysis and interpretation of data transmitted through networks to identify and extract information encapsulated within network traffic. As networks grow in complexity, providing high-quality network services and protecting network security necessitate operators to engage in a diverse range of network traffic understanding tasks [1]. These tasks include network fault diagnosis, application classification, intrusion detection, Quality of Experience (QoE) inference, and many other tasks [2]–[5]. However, due to the growing complexity and continuous expansion of networks [6], along with the increasing proportion of encrypted traffic [7], network traffic understanding faces severe challenges.

This work is supported by National Key R&D Program of China (No. 2023YFB3106304)

* Corresponding Authors: Xiaohui Xie and Yong Cui

979-8-3503-5171-2/24/$31.00 ©2024 IEEE

Many efforts have been made to address the task of network traffic understanding, yet existing methods possess certain limitations. These methods can be categorized into four categories: rule-based and fingerprint-based matching, statistical machine learning, deep learning methods, and pre-trained models. Rule-based and fingerprint-based matching approaches extract specific patterns from network traffic. Statistical machine learning approaches [4], [8], [9] extract statistical features of network traffic. Deep learning methods [10]–[13] using deep learning models like CNN and RNN to process raw network traffic in an end-to-end manner. However, the efficacy of these approaches is significantly limited by their dependence on manually designed features or requirements for extensive labeled data, which is laborious and time-consuming [14]–[16]. In today's rapidly evolving networks, where traffic characteristics and patterns are constantly changing, the generalization ability of models becomes increasingly crucial [17].

Since the introduction of the transformer architecture [18], pre-trained models have made significant progress in various fields, including computer vision [19], natural language processing [20], and many others [21]. The input data are initially transformed into token sequences and then mapped to input vectors through the embedding layer. Following this, transformer-based models undergo self-supervised pre-training on extensive unlabeled data to discover universal structures and general patterns. Subsequently, by fine-tuning on a modest amount of task-specific labeled data, pre-trained models transfer the domain knowledge acquired during pre-training to specific downstream tasks, achieving promising performance [22]. In the field of network traffic understanding, ET-BERT [23] utilizes BERT [20] to perform end-to-end traffic classification on the payload of encrypted traffic; flow-MAE [24] and YaTC [25] transform packets into grayscale images and then use MAE [19] for traffic classification. However, these methods share common limitations:

- The inputs of these methods only contain features from the packet contents, lacking temporal information about network dynamics. Mapping information from different modalities into the same input space while avoiding ambiguity and preserving the correct semantics is challenging.
- The design of the pre-training target is limited to focusing on the static characteristics of the data stream, failing to learn domain knowledge about the network transmission

process. Constructing pre-training targets that help to understand both static and dynamic aspects of network traffic remains an unsolved problem.

Consequently, these models can only deal with network traffic understanding tasks that predominantly rely on static features, e.g., traffic classification. However, when confronted with more challenging tasks, such as encrypted traffic classification where static features are less discernible, their performance significantly declines. Furthermore, their effectiveness is limited in tasks that necessitate an understanding of network dynamics, such as QoE inference. This limitation hampers the models' generalization capabilities and their ability to address a broad spectrum of network traffic understanding tasks.

To address the challenges mentioned above, we propose a novel **P**re-trained model for network **T**raffic **U**nderstanding (PTU). Firstly, we develop a traffic representation scheme that transforms network traffic into a sequence of vectors. This scheme integrates packet content and network dynamics into a unified input space, incorporating multi-modality information to ensure a meaningful representation of network traffic. The embedding layer is modified to mitigate semantic ambiguity between packet tokens and temporal tokens, effectively converting token sequences into discriminative input vectors for PTU. Secondly, we design a pre-training method with four specific targets to learn the semantic and structural attributes of network traffic, focusing on both static and dynamic aspects. The static characteristics of network traffic encompass the semantic structure within packets as well as the contextual relationships between different packets. Hence, we design the Masked Packet Reconstruction (MPR) target to aid in understanding semantics within packets and the Same Session Prediction (SSP) target to grasp the relevance between packets. The dynamic characteristics of network traffic primarily refer to temporal properties of network transmission. We design two additional pre-training targets, Historical and Future Interval Prediction (HIP and FIP), to learn the temporal features of network traffic from the perspective of network transmission.

After pre-training on a large-scale, task-agnostic dataset utilizing the four pre-training targets, PTU is adept at handling a variety of network traffic understanding tasks by fine-tuning with a minimal amount of task-specific labeled data. To validate PTU's performance, we apply it to two primary categories of network traffic understanding tasks: traffic classification and QoE inference. Experiment results demonstrate the superiority of PTU compared to various baseline models. In all tasks evaluated, PTU achieves state-of-the-art (SOTA) results. In traffic classification tasks, PTU achieves an F1 score of over 0.99. In the most challenging task of encrypted application classification, it achieves a more than 10% improvement in accuracy over previous methods. In QoE inference tasks, PTU leads by an average of over 10% in accuracy compared to other pre-trained models and achieves an average F1 score of over 0.93. Moreover, PTU exhibits strong robustness, maintaining an accuracy above 90% even when the amount of labeled data was reduced to a quarter of the original, with no significant performance degradation observed.

Our main contributions are listed as follows:

1) We propose a traffic representation scheme that converts network packets into input vectors enriched with information from two modalities–packet content and network dynamics.
2) We design a pre-training method tailored for network traffic understanding tasks, featuring four dedicated pre-training targets that leverage large-scale unlabeled data to learn a universal representation of network traffic.
3) We implement PTU and demonstrate how to apply PTU to perform various network traffic understanding tasks. Extensive experiments are conducted to verify the effectiveness and applicability of PTU. In addition to the overall performance comparison, a fine-grained analysis of how different design aspects of the model contribute to its performance is also carried out, which can enlighten further research.

II. Model Design

To tackle the diverse tasks of network traffic understanding, we propose **P**re-trained model for network **T**raffic **U**nderstanding (PTU), a model pre-trained on extensive unlabeled raw traffic and then capable of being fine-tuned for downstream tasks with minimal labeled data. In this section, we will sequentially detail the process of constructing input vectors from network traffic and the pre-training method of PTU. The entire process is illustrated in Fig. 1. The differences and comparisons between PTU and other pre-trained models in terms of traffic representation and pre-training methods are summarized in Table I. The values of the main hyper-parameters used in PTU will be discussed in Section III-B.

A. Traffic Representation

There are two main reasons for selecting the packet as the input unit. First, to understand network traffic accurately and in real-time during deployment, it is necessary to conduct fine-grained detection on a per-packet basis [5], [26]. Second, packets are semantically complete and self-contained units, which is analogous to how LLM uses semantically complete sentences as input units [27]. This similarity allows PTU to analyze the semantics of network traffic effectively.

The network and transport headers of the packet contains important information, which is crucial for network traffic understanding [6]. Removing entire header field as done by ET-BERT may lead to substantial performance loss, as we discuss in Section IV-C. To prevent the model from learning harmful biases from strong identification fields of the packet header, which could lead to incorrect causal reasoning, we remove IP addresses and MAC addresses from the packet headers [28]. Such biases would impair the model's performance upon deployment.The checksum field is also removed, as it loses the original semantics after removing the IP addresses. Subsequently, packet content is segmented into token sequences by dividing every L_p bits into individual tokens.

Network dynamics are also crucial for specific network traffic understanding tasks, e.g., QoE inference. To address

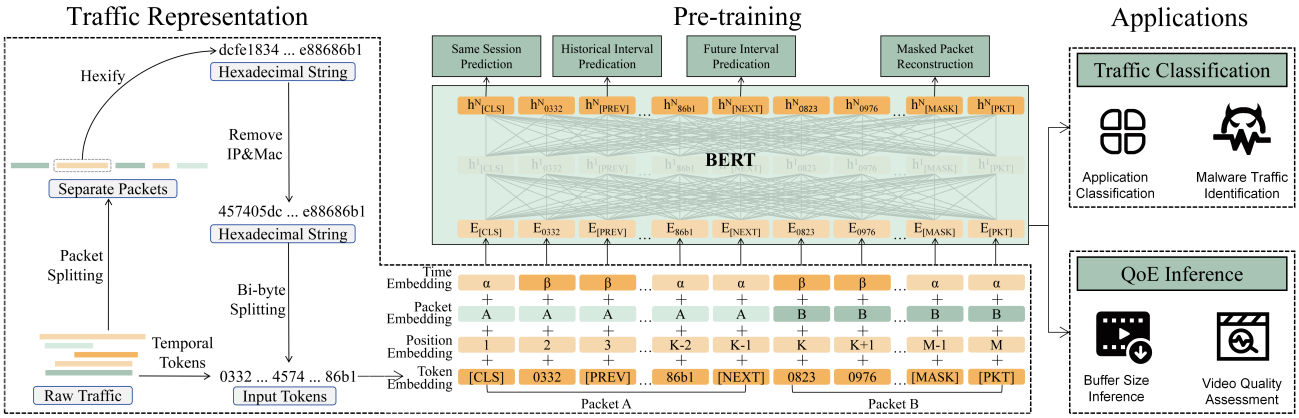


Fig. 1: The architecture and application of PTU

a broad spectrum of network traffic understanding tasks, we incorporate network dynamic features for input construction. Specifically, we choose to incorporate the inter-arrival time of packets belonging to the same session flow. A session flow represents the bidirectional data transmission process between two endpoints in network transmission. It is defined as a sequence of packets with the same protocol, IP, and port quintuple (or with the source and destination IP and port reversed). The inter-arrival time of these packets collectively reflects the end-to-end network dynamics.

For the n^{th} packet within a session, we incorporate the most recent min(n,N_t) inter-arrival time of that session flow. This approach incorporates temporal information from the preceding period of each packet, enabling the model to learn and comprehend network dynamics. Subsequently, the token sequence is truncated or padded to N_i tokens in length, forming the final input tokens, as illustrated in the lower-middle of Fig. 1.

Given the substantial variability in packet inter-arrival times, ranging from the order of seconds to nanoseconds [29], it is necessary to perform normalization. The inter-arrival time is normalized into the range (0,1), and then L_t decimal places are taken to form the temporal token corresponding to this inter-arrival time. Given most packet inter-arrival times are relatively small while a minority of inter-arrival times may be several orders of magnitude greater than the average, min-max normalization would not work properly. To better distribute the inter-arrival times evenly across the entire token space, we use (1) for normalization, inspired by [30], [31].

$$temporal\_token = sigmoid(log10(inter\_arrival)) \quad (1)$$

This transformation provides a finer granularity for numerically smaller inter-arrival times, ensuring a uniform projection from the inter-arrival times to the entire token space.

Subsequently, an embedding layer is used to convert the token sequence into input vectors for PTU. The embedding layer utilized by most LLMs [20] consists of three components: token embedding, position embedding, and segment embedding. However, due to the inherent differences between network traffic and natural language, the embedding layer needs to be modified to process the network traffic. Firstly, in network traffic, the semantic relationship between tokens belonging to different packets is distinct from the relationship among tokens within the same packet. To differentiate tokens of different packets, the segment embedding, originally used to distinguish between sentences, is modified into packet embedding. The packet embedding assigns tokens from different packets with distinct embedding values. Secondly, the input tokens of PTU comprise both packet tokens and temporal tokens, which are semantically distinct. To differentiate between these two modalities, we assign independent lookup table weights in the token embedding for each type of token, thereby preventing semantic ambiguity and confusion. Additionally, to enable the model to recognize the distinction and differences between packet tokens and temporal tokens, we add time embedding. The time embedding provides packet tokens and temporal tokens with distinct embedding values, 0 for non-temporal token and 1 for temporal token, highlighting the heterogeneity between different modalities. The input vector for PTU is obtained by summing the outputs of token embedding, position embedding, packet embedding, and time embedding. The structure of the embedding layer and the process of generating input vectors is depicted in the lower-middle of Fig. 1.

B. Pre-training Method

Before being applied to specific downstream tasks, Transformer-based models are pre-trained on large-scale unlabeled data to learn general characteristics and extract meaningful representations. These general features apply to various downstream tasks and enhance performance, as they reflect the universal patterns and structures in the data rather than task-specific information.

Existing pre-training methods are either not directly applicable to the scenario of network traffic understanding or have certain shortcomings. The pre-training objectives of models like BERT and GPT are primarily designed for the natural language process. However, there exists a considerable disparity between network traffic and natural language. For

TABLE I: Traffic Representation and Pre-Training Methods Comparison

Models	Packet Header	Packet Payload	Bias Removal	Temporal Info	Intra-packet Structure	Inter-packet Relation	Network Dynamic
ET-BERT [23]	✘	✔	✔	✘	✔	✔	✘
YaTC [25]	✔	✔	✘	✘	✔	✘	✘
PTU(Ours)	✔	✔	✔	✔	✔	✔	✔

instance, there is a significant difference in the structural organization between natural language and network traffic. Natural language can be depicted solely by the content. In contrast, beyond the value of the byte stream, the temporal characteristics of network transmission are also indispensable for a comprehensive description of network traffic.

Models like ET-BERT attempt to bridge the gap between the structural form of network traffic and natural language. It achieves this by conceptualizing BURSTs—aggregates of packet payloads heading in the same direction—as analogous to sentences in natural language, thereby repurposing BERT's pre-training tasks. Nonetheless, this methodology falls short in addressing the semantic disparities that exist between network traffic and natural language. Furthermore, it overlooks the dynamic and evolving nature of network traffic, which is a critical oversight in the context of comprehensive traffic understanding. Consequently, there is an imperative need to conceive more comprehensive pre-training methodologies that are specifically calibrated to the nuanced attributes of network traffic.

For network traffic, the crucial information includes both the static features of the byte stream and the dynamic characteristics of the network transmission process. Combining static and dynamic attributes is essential to fully describe network traffic. However, existing pre-trained models only focus on a small part of this information, as shown in Table I. For static features, the individual packet is the fundamental unit of network traffic. Therefore, the semantic meaning of bytes within a packet is crucial. The session flow, on the other hand, represents the communication stream between endpoints. The structure of session flow and the relationships among packets are vital for comprehending the structure and semantics of network traffic. In terms of dynamic characteristics, the temporal information of the network transmission process is essential.

To ensure PTU achieves a holistic comprehension of network traffic, encompassing both the temporal dynamics and the packet content, we design four dedicated pre-training targets: Masked Packet Reconstruction for intra-packet semantics, Same Session Prediction for session structure, Historical Interval Prediction and Future Interval Prediction for temporal information. Together, these pre-training targets constitute a robust framework, empowering PTU to learn all the essential domain knowledge about network traffic that are imperative for a wide array of traffic understanding tasks.

For pre-training input, we concatenate the tokens corresponding to two packets and append a special token [PKT] at the end of each packet's token sequence for separation. Similarly we add a special token [CLS] at the onset of the input token sequence to aggregate information across the entire input. To obtain a deep bidirectional contextualized representation of network traffic, the sequence of input vectors is transformed into output vectors by passing it through a BERT-based model backbone, during which contextual information is acquired. The process of pre-training is illustrated in Fig. 1. In the following, we elaborate the methodology and rationale behind PTU's four pre-training targets.

Masked Packet Reconstruction (MPR): This target, akin to the MLM target in BERT [20], facilitates PTU in discerning the semantic architecture encapsulated within network packets and distilling bidirectionally contextualized representation from packet content. Furthermore, it ensures that the token embedding is adept at encapsulating the distinctive semantic nuances of the content tokens within the packets. During pre-training, we randomly select $P_{masked}\%$ of the packet tokens and replace them with [MASK] with a probability of $P_{mask}\%$, replace them with a random token with a probability of $P_{random}\%$, and leave them unchanged with a probability of (100-P_{mask}-P_{random})%. The model is trained to restore the selected tokens based on the context, utilizing the negative log-likelihood function as the loss function, which is formally defined as follows:

$$L_{MPR} = -\sum_{i=1}^{k} log(P(MASK_i = token_i|\hat{X};\theta)) \quad (2)$$

Where θ represents the trainable parameters of PTU, k is the total number of selected packet tokens, $\hat{X}$ is the input sequence X after the masking process, $MASK_i$ is the output result token by PTU of the i^{th} masked token and $token_i$ is the original value of the i^{th} masked token.

Same Session Prediction (SSP): To enhance PTU's capability in discerning logical interconnections among packets, we substitute the NSP target in BERT [20] with the Same Session Prediction (SSP) target. This novel target is meticulously crafted to help the model discern the characteristics and boundaries of session flows, thereby achieving a profound comprehension of the hierarchical fabric inherent in network traffic, enabling a more nuanced understanding of network traffic dynamics. During pre-training, two packets A and B are concatenated and input into the PTU, where B is the subsequent packet after A in the same session with a probability of $P_{next}\%$, a randomly selected packet from A's session with a probability of $P_{session}\%$, and from a different session flow with (100-P_{next}-$P_{session}$)% probability. The output vector corresponding to the [CLS] token is fed into a binary classifier to tell whether the two packets belong to the same session. This process enables the [CLS] output vector to encapsulate a summary of the entire input sequence's information, making this output vector suitable for downstream network traffic

understanding tasks. The loss function for a given pair of input packets $\overline{P}$ = (P_A, P_B) and their true label y ∈ [0,1] is formally defined as:

$$L_{SSP} = -log(P(y|\overline{P};\theta)) \tag{3}$$

Historical and Future Interval Prediction (HIP and FIP): These targets are meticulously engineered to facilitate PTU's comprehension of network traffic dynamics. They are crafted to enable the model to discern the intricate processes that data packets undergo during transmission. Furthermore, these objectives enable PTU to extract the holistic end-to-end status of the network's underlying infrastructure. For the Historical Interval Prediction target, we select P_{time}% of the temporal tokens and replace them with [PREV], requiring the model to infer the original temporal tokens. For the Future Interval Prediction target, the inter-arrival time between each packet and the next packet in the session flow is recorded during data processing. Then, with a P_{time}% chance, we replace the terminal token [PKT] of each packet with [NEXT] and require the model to predict the corresponding next interval. Considering the semantic proximity of numerically adjacent temporal tokens, perturbation is introduced by adding Gaussian noise to the target temporal tokens. This approach enables the model to discern the accurate semantics of temporal tokens and grasp the similarity of adjacent intervals. The total loss function for these two targets is defined as:

$$L_{HIP} = -\sum_{i=1}^{t} log(P(TIME_i = hi_i|\hat{X};\theta)) \tag{4}$$

$$L_{FIP} = -log(P(Fi = fi|\hat{X};\theta)) \tag{5}$$

Where t is the total number of selected temporal tokens, hi_i is the expected value of the i^{th} chosen temporal token, $TIME_i$ is the predicted token of the i^{th} chosen temporal token, Fi and fi correspond to the expected and the predicted value of the future inter-arrival time.

The overall pre-training loss is the sum of the loss from the four targets mentioned above:

$$L = L_{MPR} + L_{SSP} + L_{HIP} + L_{FIP} \tag{6}$$

III. Application and Implementation

In this section, we will first discuss how to apply PTU to various traffic understanding tasks, including traffic classification and QoE inference. Subsequently, we will elaborate on the methodologies and details concerning the implementation of PTU.

A. Application of PTU

We apply PTU at the packet level for network traffic understanding tasks to achieve real-time processing and fine-grained analysis. The model's architecture used for downstream tasks is fundamentally consistent with pre-training. Starting with the pre-trained model weights, the entire PTU model is fine-tuned on a small set of task-specific labeled data. Due to the presence of the SSP target, the output vector corresponding to the [CLS] token aggregates information from the entire input sequence during pre-training, so we utilize the output vector corresponding to [CLS] for downstream tasks. To apply PTU to a variety of traffic understanding tasks, the sole modification required is the addition of a task-specific output layer atop the [CLS] token's output vector.

In this paper, PTU is applied to two major categories of tasks: traffic classification and QoE inference. Traffic classification involves categorizing network traffic into predefined categories. Typically, a session flow is assigned to a single category, which represents its inherent and intrinsic attributes. This task predominantly relies on the static features of network traffic. However, when dealing with encrypted traffic, the static features become obscured and less discernable due to encryption. In such cases, temporal features of the network traffic also contribute to the classification of encrypted traffic. QoE inference refers to predicting the QoE metrics of network traffic within the current time slice. Typically, a session flow exhibits varying QoE metric values at different moments. This task necessitates a comprehensive understanding of both the upper-layer application information encapsulated within the packet content and the dynamics of the underlying network.

Benefiting from the domain knowledge about packet content and network dynamics acquired during pre-training, PTU achieves satisfactory performance on a variety of downstream tasks with only a small amount of labeled data. The superior performance and low dependence on the volume of labeled data endow PTU with high practical value, making it applicable to various scenarios and network traffic understanding tasks.

B. Implementation of PTU

In this study, we employ the stardard BERT [20] as the backbone of PTU, which comprises 12 attention heads and 12 layers, utilizing a 768-dimensional embedding vector for each token. And the implementation is based on Uer-py [32] and PyTorch 2.1.2.

Selecting an appropriate vocabulary size can significantly enhance the pre-trained model's performance [33]. Empirically, most BERT-based pre-trained models opt for a vocabulary size of around tens of thousands or so. To obtain a comparable vocab size, we segment the byte sequence of the packet into tokens every L_p=16 bits, setting the vocabulary size to $2^{16} = 65536$. We add most N_t=10 inter-arrival time and truncate the total input length to N_i=128 tokens for each packet. We run a hyper-parameter tuning step using a coarse grid search. The following settings yield the best performance: P_{masked}=20, P_{mask}=80, P_{random}=10, P_{next}=30, $P_{session}$=30, and P_{time}=50.

For pre-training, approximately 100GB of unlabeled network traffic data is collected from public datasets [34] [35]. This traffic data encompasses various network protocols, including but not limited to QUIC, TLS, FTP, HTTP, and SSH, and contains both encrypted and plaintext packets. Notably, there is no overlap between the unlabeled datasets used for pre-training and the labeled task-specific datasets used for performance evaluation, ensuring the accuracy of performance

assessments and validating the generalization ability and robustness of PTU. The pre-training corpus and the resulting pre-trained model weight is the same for all downstream tasks, alleviate the pressure for data collection.

For fine-tuning, the inputs are similar to pre-training, and the network traffic understanding tasks are performed at the granularity of individual packet. For each task, a maximum of 5000 samples per category is selected. The dataset is then divided into training, validation, and testing sets in an 8:1:1 ratio.

All experiments are conducted on a server equipped with NVIDIA GeForce RTX 3090 GPUs. During the pre-training phase, four GPUs are utilized, whereas for fine-tuning, a single GPU suffices. For pre-training, the batch size is set to 32, with a total of 500,000 steps completed over approximately five days. During fine-tuning, the batch size is maintained at 32 throughout 10 epochs. Depending on the size of the labeled dataset, the fine-tuning process will take about 3 to 10 hours.

IV. Experiment

We conduct evaluation experiments on a variety of network traffic understanding tasks divided into two major categories: traffic classification and QoE inference. The evaluation experiments aim to address the following research questions (RQ):

- RQ1: How does PTU perform on the two major categories of downstream tasks–traffic classification and QoE inference–compared to previous approaches?
- RQ2: Whether the components of traffic representation and the four tailored pre-training targets positively influence PTU's performance on downstream tasks.
- RQ3: How does PTU perform in few-shot learning scenarios? Few-shot learning refers to learning and generalizing from a minimal number of labeled data.
- RQ4: How is the inference efficiency of PTU, and what factors influence it?

A. Evaluation Setup

Datasets. The tasks of traffic classification are conducted on the following datasets: Application classification and service classification on ISCX-VPN-2016 [36], malware traffic identification on USTC-TFC-2016 [37], encrypted application classification on CSTNET-TLS-1.3 [23], network attack identification on CIC-IoT-2022 [38], mobile application classification on Cross-Platform(iOS) [39] and mobile application classification on Cross-Platform(Android) [39]. The tasks of QoE inference are conducted on the following datasets: buffer level classification, video resolution classification, and video state classification on CC-YouTube-QoE-2018 [3], MOS inference on KoNViD-1k [40] and MOS inference on LIVE Netflix [41]. For the two MOS inference tasks, due to the lack of publicly available packet capture files, we construct inputs based on the method used in LSTM-QoE [13], using video metadata and video features within every 100ms time slice and truncated the total input length to 128 tokens.

Evaluation Metrics. For traffic classification tasks and QoE classification tasks, the inference result corresponds to one label out of a finite set of labels for each input. We compute two well-established metrics for these tasks: Accuracy and F1 score [39] [42]. For the task's overall performance, the macro-average of each class is calculated to avoid biased results due to data imbalance [43]. For the two MOS inference tasks mentioned above, the inference result is a sequence of MOS scores. Following the evaluation metrics established in prior literature for these tasks [44], we calculate the Pearson Linear Correlation Coefficient (PLCC) and Spearman Rank Correlation Coefficient (SRCC) between predicted and actual MOS sequences of the video stream.

Baselines. For comparison, several SOTA methods are chosen. (1) Fingerprint-based matching: Flowprint [39]; (2) Statistical machine learning: CUMUL [8] and Requet [3]; (3) Deep learning methods: Deeppacket [10], DA-QOE [44] and LSTM-QOE [13]; (4) Pre-trained model: ET-BERT [23] and YaTC [25]. Flowprint, CUMUL, and Deeppacket, designed for traffic classification tasks, are tested on corresponding traffic classification datasets. Requet, proposed to address QoE metric classification tasks, is evaluated on the discrete label QoE dataset CC-YouTube-QoE-2018 [3]. LSTM-QOE and DA-QOE, designed to solve MOS inference tasks for video streams, are tested on datasets featuring continuous MOS scores, namely LIVE Netflix [41] and KoNViD-1k [40]. Performance evaluation of the pre-trained models ET-BERT, YaTC, and our model PTU is conducted across all datasets to validate their generalization ability.

B. Overall Comparison

The evaluation results are listed in Table II, III, and IV. In summary, PTU achieves SOTA performance across all tasks tested, with a significant accuracy improvement compared to previous methods. PTU's high accuracy and superior performance on downstream tasks are primarily attributed to the domain knowledge of packet content and network dynamics acquired during pre-training. This knowledge is efficiently transferred to specific downstream tasks with the aid of a small amount of labeled data.

Traffic Classification: PTU achieves an accuracy rate exceeding 99% in all traffic classification tasks conducted, outperforming other methods. In the most challenging task of encrypted application classification, PTU achieves a more than 10% improvement in accuracy, demonstrating the exceptional performance of PTU.

FlowPrint exhibits a significant performance decline on datasets with stronger encryption protocols, highlighting the incompetence of matching-based methods in dealing with encrypted traffic. CUMUL performs poorly overall, indicating its heavy dependence on manually selected features, as its default statistical feature set does not apply well in most scenarios. Deeppacket experiences pronounced performance degradation in certain scenarios, reflecting the weak robustness of deep learning models.

Pre-trained models demonstrate superior performance across all tasks tested, suggesting that the domain knowledge acquired through pre-training can substantially improve per-

TABLE II: Performance Evaluation Results of Traffic Classification tasks

Dataset	VPN-App [36]		VPN-Ser [36]		iOS [39]		Android [39]		TFC-2016 [37]		TLS-1.3 [23]		CIC-IoT [38]	
Metric	AC	F1	AC	F1	AC	F1	AC	F1	AC	F1	AC	F1	AC	F1
Flowprint [39]	0.8675	0.6629	0.7894	0.7752	0.9322	0.9159	0.8799	0.8845	0.8097	0.6679	0.1184	0.1070	0.4992	0.4554
CUMUL [8]	0.5208	0.4237	0.5879	0.5727	0.2763	0.1975	0.3667	0.2097	0.5763	0.5775	0.5807	0.5385	0.9289	0.9138
Deeppacket [10]	0.9545	0.9524	0.9339	0.9357	0.9104	0.9052	0.7985	0.8203	0.9624	0.9611	0.8024	0.8203	0.7838	0.7793
ET-BERT [23]	0.9604	0.9665	0.9572	0.9570	0.9743	0.9743	0.9228	0.9059	0.9821	0.9821	0.8978	0.8835	0.7645	0.7543
YaTC [25]	0.9403	0.9407	0.9550	0.9589	0.9721	0.9728	0.9355	0.9342	0.9786	0.9795	0.8792	0.8884	0.9333	0.9324
PTU(Ours)	**0.9969**	**0.9969**	**0.9986**	**0.9981**	**0.9984**	**0.9994**	**0.9980**	**0.9992**	**0.9992**	**0.9993**	**0.9945**	**0.9940**	**0.9947**	**0.9942**

TABLE III: Performance Evaluation Results of QoE Inference tasks with discrete labels

Dataset	CC-Buffer [3]		CC-Resolution [3]		CC-State [3]	
Metric	AC	F1	AC	F1	AC	F1
Requet [3]	0.9202	0.9105	0.6695	0.6689	0.7685	0.8031
ET-BERT [23]	0.8083	0.8154	0.8379	0.8376	0.6816	0.6782
YaTC [25]	0.8166	0.7914	0.8552	0.8487	0.7207	0.7243
PTU(Ours)	**0.9274**	**0.9232**	**0.9493**	**0.9477**	**0.8597**	**0.8576**

formance in traffic classification tasks. However, ET-BERT, which does not incorporate information from the packet header, struggles to identify the flood attacks in the CIC-IoT dataset. These flood attacks are primarily composed of overwhelming influxes of data packets sent at a high velocity, but the payloads of these packets are legitimate, rendering the payload-only model ET-BERT inadequate. YaTC, on the other hand, neglects the dimension of network dynamics, failing to comprehensively and accurately understand network traffic, thus leading to a significant decline in performance on the TLS-1.3 dataset with the highest level of encryption.

The input vectors of PTU aggregate information from packet headers, packet payloads, and network dynamics. This multi-modal information promotes PTU to extract features from multiple dimensions of network traffic comprehensively. The four pre-training targets help the model construct contextualized and universal representation, learning domain knowledge about the structure of packets and the characteristics of transmission. With a modest amount of fine-tuning, PTU transfers the domain knowledge to specific traffic classification tasks and achieves excellent performance across various scenarios, surpassing 99% in accuracy and F1 score. These results demonstrate the effectiveness, robustness, and generalization ability of PTU.

TABLE IV: Performance Evaluation Results of QoE Inference tasks with continuous labels

Dataset	KoNViD-1k [40]		LIVE Netflix [41]	
Metric	PLCC	SRCC	PLCC	SRCC
DA-QoE [44]	0.8270	0.8185	0.8144	0.8280
LSTM-QoE [13]	0.8090	0.7960	0.8230	0.7250
ET-BERT [23]	0.8161	0.8043	0.7533	0.7408
YaTC [25]	0.8042	0.8394	0.8323	0.8496
PTU(Ours)	**0.9429**	**0.9741**	**0.9608**	**0.9860**

QoE Inference: In all QoE inference tasks evaluated, PTU maintains high accuracy and achieves SOTA performance. Compared to other pre-trained models, PTU leads by an average of 10% in accuracy and achieves an average F1 score of over 0.93. These results confirm PTU's capability to handle a wide range of network traffic understanding tasks.

Requet meticulously constructs and selects statistical features based on the characteristics of YouTube video streams, achieving an accuracy rate above 90% in the binary classification task CC-Buffer for YouTube video streams. However, Requet exhibits a significant decline in performance on other tasks, indicating that its chosen set of statistical features lacks universality and only applies to specific scenarios and tasks.

Previous pre-trained models, namely ET-BERT and YaTC, do not exhibit superior performance over other specifically designed smaller models in QoE inference tasks. The poor efficacy is due to the fact that these pre-trained models only utilize features derived from packet content for input construction, neglecting temporal information. They also fail to acquire domain knowledge about the network transmission process during pre-training. Consequently, when dealing with QoE inference tasks that require understanding both application behavior and network dynamics, these models show poor performance.

In contrast, PTU leverages multi-modal information from multiple dimensions of network traffic for input construction, enabling a complete, comprehensive, and accurate depiction of network traffic. The four tailored pre-training targets allow PTU to acquire domain knowledge regarding both the static packet content and dynamic network transmission, generating discriminative and meaningful representations of network traffic. Consequently, PTU can grasp the characteristics of application behavior and the temporal features of the underlying network, achieving superior performance in QoE inference tasks and outperforming all other methods. These results confirm PTU's generalization ability and capability to handle a broad range of network traffic understanding tasks.

Answer to RQ1: PTU achieves SOTA performance in all tested network traffic understanding tasks. In traffic classification tasks, PTU achieves an F1 score of over 0.99 and a more than 10% improvement in accuracy over previous methods. In QoE inference tasks, PTU leads by an average of over 10% in accuracy compared to other pre-trained models and achieves an average F1 score of over 0.93.

TABLE V: Ablation Study

Dataset	CSTNET-TLS-1.3 [23]		CIC-IoT-2022 [38]		CC-Buffer [3]		CC-Resolution [3]		CC-State [3]	
Metric	AC	F1	AC	F1	AC	F1	AC	F1	AC	F1
PTU(full model)	**0.9945**	**0.9945**	**0.9947**	**0.9942**	**0.9274**	**0.9273**	**0.9493**	**0.9496**	**0.8597**	**0.8593**
w IP	0.9657	0.9673	0.9922	0.9924	0.8981	0.8987	0.9257	0.9247	0.8206	0.8208
w/o header	0.9291	0.9297	0.9713	0.9807	0.8612	0.8614	0.8918	0.8936	0.8134	0.8141
w/o temporal token	0.9422	0.9355	0.9092	0.9054	0.8303	0.8359	0.8581	0.8579	0.6889	0.6881
w/o time embedding	0.9527	0.9553	0.9106	0.9098	0.8321	0.8402	0.8612	0.8630	0.7029	0.7109
w/o pre-training	0.9132	0.9136	0.9806	0.9814	0.8467	0.8495	0.9138	0.9123	0.7872	0.7875
w/o MTM	0.9142	0.9103	0.9754	0.9731	0.9032	0.9051	0.9271	0.9268	0.8461	0.8458
w/o SSP	0.9358	0.9359	0.9903	0.9902	0.9178	0.9191	0.9284	0.9276	0.8431	0.8429
w/o HIP	0.9467	0.9429	0.9913	0.9889	0.8742	0.8705	0.9152	0.9147	0.7968	0.7934
w/o FIP	0.9582	0.9601	0.9926	0.9904	0.8698	0.8684	0.9226	0.9213	0.8031	0.8042

C. Ablation Study

To verify the contribution of each component, ablation study is conducted across multiple datasets. In Table V, "w IP" denotes the input construction that retains IP address, MAC address, and checksum fields; "w/o header" refers to the input construction that excludes packet headers; "w/o temporal token" indicates the input construction without incorporating inter-arrival times; "w/o time embedding" denotes the absence of the time embedding within the embedding layer. The pre-trained model weights used for the four categories mentioned above are consistent with the full model. The four categories listed below, however, maintain consistency with the full model in terms of input construction but differ in the pre-training method: "w/o pre-training" refers to the model weights are randomly initialized for fine-tuning; "w/o MPR", "w/o SSP", "w/o HIP" and "w/o FIP" stands for pre-training without corresponding pre-training targets proposed in Section II-B.

Traffic Representation: Retaining strong identification fields such as IP addresses leads to a moderate accuracy decrease. The limited performance loss can be attributed to the consistent correlation between IP addresses and labels within the training and testing subsets of a given dataset. However, when deployed in production environments, the presence of these fields may cause a more pronounced performance degradation. Discarding packet header fields, such as port numbers and protocol types, deprives PTU of crucial information, thus significantly impairing performance.

The exclusion of temporal tokens results in the loss of critical information regarding the network's temporal dynamics, leading to an incomplete and less accurate understanding of network traffic, thus resulting in lower accuracy across all traffic understanding tasks, with a pronounced effect on QoE inference tasks. The removal of time embedding introduces ambiguity and confusion, as PTU struggles to differentiate between temporal and packet tokens semantically. As a result, PTU faces difficulty in extracting meaningful and discriminative representations from network traffic, causing a decrease in accuracy across all tasks, with a particularly significant impact on QoE inference tasks that rely more heavily on temporal information.

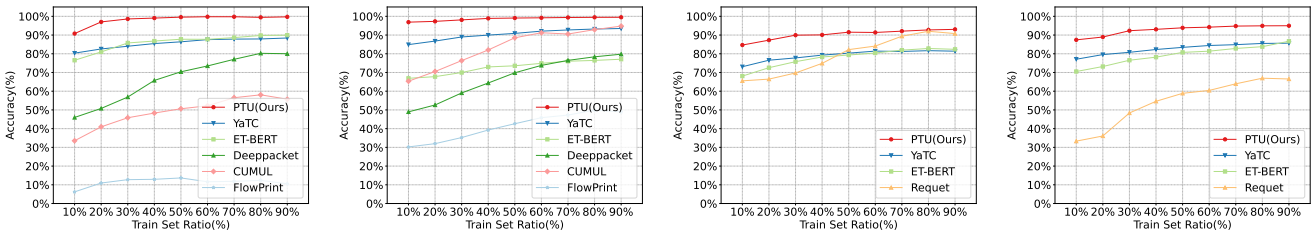
Pre-training Targets: Our experiments demonstrate that the four tailored pre-training targets are essential for constructing a deep contextualized and discriminative representation of network traffic. The complete absence of pre-training leads to the most severe performance degradation, validating the importance of universal domain knowledge obtained through pre-training. MPR focuses on the semantic and structural features of packet content, and SSP focuses on the semantic relationship between packets and the structure of session flows. The absence of both MPR and SSP results in comparable levels of performance degradation, suggesting that intra-packet features and inter-packet relationships hold similar importance for network traffic understanding. The omission of HIP or SIP both leads to a significant decline in the performance across all tested tasks, with a particularly pronounced impact in QoE inference tasks. This indicates that even for tasks that primarily rely on static packet content, like traffic classification, incorporating domain knowledge of network dynamics can still significantly improve performance. For QoE inference tasks, which heavily depend on the comprehension of the underlying network transmission processes, understanding network dynamics is essential.

These results signify the importance of a comprehensive understanding of network traffic, which includes both static and dynamic elements, for effectively tackling the various challenges associated with network traffic analysis tasks. The pre-training phase equips the PTU with foundational network domain knowledge, which is pivotal for achieving optimal performance in downstream tasks.

Answer to RQ2: Experimental results demonstrate that the components of traffic representation and each pre-training target are indispensable for the superior performance of PTU. These components are crucial in comprehensively and accurately understanding network traffic.

D. Few-shot Learning Analysis

To assess the robustness of PTU in few-shot learning scenarios, we deviate from the original 8:1:1 split ratio and instead vary the proportion of training set from 10% to 90% in increments of 10%. The results are presented in Fig. 2. Experimental results demonstrate that in few-shot learning scenarios, pre-trained models such as ET-BERT, YaTC, and



(a) CSTNET-TLS-1.3 (b) CIC-IoT-2022 (c) CC-YouTube-Buffer (d) CC-YouTube-Resolution

Fig. 2: Few-shot Analysis

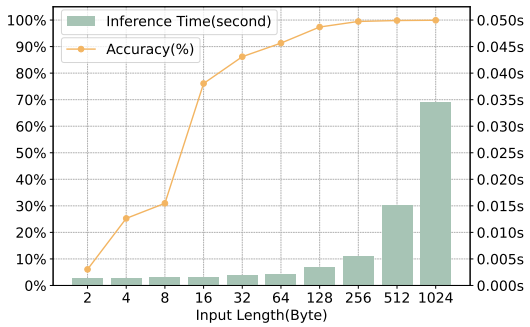


Fig. 3: The effect of input length on inference efficiency of PTU

PTU outperform other methods and exhibit less performance degradation when facing the scarcity of labeled training data. The self-supervised pre-training on large-scale unlabeled data enables the models to learn the underlying structure and universal patterns of network traffic, thereby diminishing their dependence on the quantity of task-specific labeled data. PTU, in particular, only exhibits a significant performance decline when the proportion of training data is minimal, i.e., at 10%. Even under these conditions, PTU maintains an accuracy level above 90%, eliminating concerns about over-fitting. Furthermore, PTU consistently outperforms the other two pre-trained models by approximately 10% in accuracy across all tested tasks. This consistent lead in accuracy underscores PTU's superior performance and robustness, which can be attributed to the universal and meaningful representations of network traffic acquired through pre-training. These results suggest that PTU can operate effectively even when the amount of labeled training data is constrained, thereby substantially reducing its deployment costs associated with data collection and labeling. It is worth noting that a smaller number of training samples also implies a reduced consumption of computational resources. That is, the PTU can still maintain a high performance with limited computility.

Answer to RQ3: The experimental results demonstrate that PTU maintains high accuracy even with a limited amount of labeled data, highlighting its exceptional robustness and generalization ability. PTU exhibits a low dependency on the quantity of labeled data, reducing data collection costs for deployment and making it suitable for production environments.

E. Efficiency Study

The efficiency of a model, which refers to its inference speed, significantly impacts its utility value. Low efficiency not only makes the model unsuitable for real-time monitoring but also substantially increases its consumption of computational resources, thereby significantly increasing operation costs. The efficiency of transformer-based models is primarily determined by the input sequence's length, as the attention operation's complexity is quadratic with respect to the length [45]. Therefore, the impact of input length on the efficiency and accuracy of PTU is investigated by truncating the input length to 2^n where n ranges from 1 to 10. The accuracy and average inference time when the batch size is set to 1 is shown in Fig. 3. The experimental results indicate that beyond an input length of 256 bytes, further extensions show no significant accuracy improvement. Conversely, longer input sequences lead to a substantial increase in inference time, negatively impacting PTU's practical value. Based on these findings, we select 256 bytes as the input sequence length for all other experiments conducted in this paper.

With an input length of 256 bytes and a batch size of 1, PTU can process 200 samples per second, whereas when the batch size is increased to 32, which is used in all other experiments, the PTU can handle approximately 3000 samples per second. Building on this, experiments employing linear attention could provide an additional acceleration of 3 to 4 times; however, simple replacement leads to a certain degree of performance degradation. Therefore, in this study, we opt to use the vanilla attention.

Answer to RQ4: Extended input lengths enhance PTU's performance by providing the model with a more comprehensive context. On the other hand, longer input results in a significant prolongation of the inference time, reducing the model's practicality. We select an input length of 256 bytes as a compromise between accuracy and efficiency.

V. Related Work

Network traffic understanding refers to analyzing, monitoring, and interpreting data streams transmitted across networks. This process involves extracting, analyzing, and comprehending information encapsulated within network traffic. To provide high-quality network services, a wide range of network traffic understanding tasks must be undertaken, including intrusion detection, application classification and QoE inference.

In general, methods of traffic understanding can be classified into four categories: rule-based and signature-based matching, statistical machine learning, deep learning methods, and pre-trained models.

Rule-based and Signature-based Matching: These approaches involve extracting specific patterns and matching them with predefined rules or pre-established signature libraries. A straightforward approach for identifying applications is through transport layer port numbers. Raimund et al. [46] perform QoE inference by analyzing packets' application layer header information. Flowprint [39] generates fingerprints for application classification based on IP addresses and other elements. However, rule-based and signature-based matching is highly dependent on plaintext information and faces challenges adapting to network environments' growing complexity and encryption.

Statistical Machine Learning: These methods begin by extracting statistical features from network traffic and then applying traditional machine learning algorithms, such as the k-nearest neighbors and the random forest. AppScanner [4] extracts forty flow-level statistical features and employs the support vector machine for website classification. Requet [3] constructs statistical temporal features from chunks and utilizes the random forest to infer key QoE metrics for YouTube video streams. Nevertheless, to accurately extract statistical features, statistical machine learning relies heavily on the large volume of labeled datasets, which can incur substantial costs. Moreover, statistical machine learning requires a wealth of expert knowledge to construct and select discriminative statistical features for different scenarios, thus limiting their generalization capability.

Deep Learning Methods: These techniques leverage advanced deep learning models, such as CNN and RNN, to perform network traffic understanding tasks in an end-to-end manner, eliminating the need for manually crafted features. Deeppacket [10] applies one-dimensional CNN to directly process packet payloads for service classification. ReClive [12] employs LSTM to analyze temporal sequences of client GET requests for QoE inference in video streaming. LSTM-QoE [13] predicts the QoE of video streams using LSTM based on temporal information such as the time elapsed since the last play. However, these models depend heavily on large volumes of labeled data to achieve satisfactory accuracy, significantly increasing deployment costs. Additionally, they may have limited generalization capability and require retraining or even redesign when encountering new traffic patterns or novel network traffic understanding tasks.

Pre-trained Models: The idea of pre-training has a long history, such as word2vec [47]. Many studies have leveraged the idea of pre-trained embedding vectors to address the challenges of traffic understanding tasks. [48]–[52] ET-BERT [23] introduces the concept of BURST and employing BERT for encrypted traffic classification. Flow-MAE [24] and YaTC [25] utilize the MAE from the field of computer vision to encode entire flows for application identification. However, these methods are limited to the features of static packet content and neglect temporal information of the network dynamics, such as packet inter-arrival times, routing paths, and transmission delay, thus failing to address a broader range of network traffic understanding tasks, e.g., QoE inference.

VI. Limitations And Future Work

Currently, with a single consumer-grade GPU, the inference speed of the PTU is approximately 2000 samples per second, which significantly lags behind the core network's requirement of processing tens of millions of packets per second [53]. Moreover, the PTU introduces an additional delay of 5 ms for each flow, which is non-negligible [54]. At present, applying the PTU to the real-world core network is still challenging.

In future work, we plan to explore measures to accelerate PTU, such as increasing the batch size, employing faster linear attention mechanisms [55] and using faster backbone model like TTT [56], to further improve the inference speed of PTU while maintain its excellent performance. Additionally, we will assess PTU's capability for continuous learning in a production environment where traffic patterns are constantly evolving. We will also apply PTU to a broader spectrum of tasks, including user behavior analysis, bandwidth allocation, and network fault diagnosis to better illustrate PTU's performance and versatility.

VII. Conclusion and Future Work

In this paper, we propose the **P**re-trained model for network **T**raffic **U**nderstanding (PTU), a pre-trained model designed for network traffic understanding tasks. PTU is pre-trained on unlabeled large-scale task-agnostic data, acquiring deep and unbiased representations of network traffic. The pre-training process enables PTU to be fine-tuned on a limited amount of labeled data and achieve exceptional performance across a wide range of network traffic understanding tasks. The efficacy of PTU is evaluated on two network traffic understanding tasks: traffic classification and QoE inference. Experimental results demonstrate that PTU surpasses previous methods and achieves SOTA results in all tasks, showcasing its superior performance and generalization ability.

REFERENCES

[1] A. Shahraki, M. Abbasi, A. Taherkordi, and A. D. Jurcut, "A comparative study on online machine learning techniques for network traffic streams analysis," *Computer Networks*, vol. 207, p. 108836, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128622000512 I

[2] S. Kamamura, Y. Takei, M. Nishiguchi, Y. Hayashi, and T. Fujiwara, "Network anomaly detection through ip traffic analysis with variable granularity," *IEEE Access*, vol. 11, pp. 129 818–129 828, 2023. I

[3] C. Gutterman, K. Guo, S. Arora, X. Wang, L. Wu, E. Katz-Bassett, and G. Zussman, "Requet: real-time qoe detection for encrypted youtube traffic," in *Proceedings of the 10th ACM Multimedia Systems Conference*, ser. MMSys '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 48–59. [Online]. Available: https://doi.org/10.1145/3304109.3306226 I, IV-A, IV-A, IV-A, III, V, V

[4] 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)*, 2016, pp. 439–454. I, V

[5] X. Hu, W. Gao, G. Cheng, R. Li, Y. Zhou, and H. Wu, "Toward early and accurate network intrusion detection using graph embedding," *IEEE Transactions on Information Forensics and Security*, vol. 18, pp. 5817–5831, 2023. I, II-A

[6] M. Abbasi, A. Shahraki, and A. Taherkordi, "Deep learning for network traffic monitoring and analysis (ntma): A survey," *Computer Communications*, vol. 170, pp. 19–41, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0140366421000426 I, II-A

[7] E. Papadogiannaki and S. Ioannidis, "A survey on encrypted network traffic analysis applications, techniques, and countermeasures," *ACM Comput. Surv.*, vol. 54, no. 6, jul 2021. [Online]. Available: https://doi.org/10.1145/3457904 I

[8] A. Panchenko, F. Lanze, J. Pennekamp, T. Engel, A. Zinnen, M. Henze, and K. Wehrle, "Website fingerprinting at internet scale," in *Network and Distributed System Security Symposium*, 2016. [Online]. Available: https://api.semanticscholar.org/CorpusID:15302617 I, IV-A, II

[9] M. H. Mazhar and Z. Shafiq, "Real-time video quality of experience monitoring for https and quic," in *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, 2018, pp. 1331–1339. I

[10] M. Lotfollahi, M. Jafari Siavoshani, R. Shirali Hossein Zade, and M. Saberian, "Deep packet: a novel approach for encrypted traffic classification using deep learning," *Soft Computing*, vol. 24, no. 8, pp. 1999–2012, 2020. I, IV-A, II, V

[11] K. Lin, X. Xu, and H. Gao, "Tscrnn: A novel classification scheme of encrypted traffic based on flow spatiotemporal features for efficient management of iiot," *Computer Networks*, vol. 190, p. 107974, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128621001067 I

[12] S. C. Madanapalli, A. Mathai, H. H. Gharakheili, and V. Sivaraman, "Reclive: Real-time classification and qoe inference of live video streaming services," in *2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS)*, 2021, pp. 1–7. I, V

[13] N. Eswara, S. Ashique, A. Panchbhai, S. Chakraborty, H. P. Sethuram, K. Kuchi, A. Kumar, and S. S. Channappayya, "Streaming video qoe modeling and prediction: A long short-term memory approach," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 3, pp. 661–673, 2020. I, IV-A, IV-A, IV, V

[14] S. Rezaei and X. Liu, "Deep learning for encrypted traffic classification: An overview," *IEEE Communications Magazine*, vol. 57, no. 5, pp. 76–81, 2019. I

[15] M. Shen, Y. Liu, L. Zhu, K. Xu, X. Du, and N. Guizani, "Optimizing feature selection for efficient encrypted traffic classification: A systematic approach," *IEEE Network*, vol. 34, no. 4, pp. 20–27, 2020. I

[16] O. Aouedi, K. Piamrat, S. Hamma, and J. Perera, "Network traffic analysis using machine learning: an unsupervised approach to understand and slice your network," *annals of telecommunications - annales des télécommunications*, 11 2021. I

[17] S. Wang, J. F. Balarezo, S. Kandeepan, A. Al-Hourani, K. G. Chavez, and B. Rubinstein, "Machine learning in network anomaly detection: A survey," *IEEE Access*, vol. 9, pp. 152 379–152 396, 2021. I

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 6000–6010. I

[19] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," *ArXiv*, vol. abs/2010.11929, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:225039882 I

[20] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *North American Chapter of the Association for Computational Linguistics*, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:52967399 I, II-A, II-B, II-B, III-B

[21] E. Alsentzer, J. R. Murphy, W. Boag, W.-H. Weng, D. Jin, T. Naumann, and M. B. A. McDermott, "Publicly available clinical bert embeddings," 2019. I

[22] C. Zhou, Q. Li, C. Li, J. Yu, Y. Liu, G. Wang, K. Zhang, C. Ji, Q. Yan, L. He, H. Peng, J. Li, J. Wu, Z. Liu, P. Xie, C. Xiong, J. Pei, P. S. Yu, and L. Sun, "A comprehensive survey on pretrained foundation models: A history from bert to chatgpt," 2023. I

[23] X. Lin, G. Xiong, G. Gou, Z. Li, J. Shi, and J. Yu, "Et-bert: A contextualized datagram representation with pre-training transformers for encrypted traffic classification," in *Proceedings of the ACM Web Conference 2022*, ser. WWW '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 633–642. [Online]. Available: https://doi.org/10.1145/3485447.3512217 I, I, IV-A, IV-A, II, III, IV, V, V

[24] Z. Hang, Y. Lu, Y. Wang, and Y. Xie, "Flow-mae: Leveraging masked autoencoder for accurate, efficient and robust malicious traffic classification," in *Proceedings of the 26th International Symposium on Research in Attacks, Intrusions and Defenses*, ser. RAID '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 297–314. [Online]. Available: https://doi.org/10.1145/3607199.3607206 I, V

[25] R. Zhao, M. Zhan, X. Deng, Y. Wang, Y. Wang, G. Gui, and Z. Xue, "Yet another traffic classifier: A masked autoencoder based traffic transformer with multi-level flow representation," in *AAAI Conference on Artificial Intelligence*, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:259716837 I, I, IV-A, II, III, IV, V

[26] E. Letsoalo and S. Ojo, "Session hijacking attacks in wireless networks: A review of existing mitigation techniques," in *2017 IST-Africa Week Conference (IST-Africa)*, 2017, pp. 1–9. II-A

[27] N. M. An, S. Waheed, and J. Thorne, "Capturing the relationship between sentence triplets for LLM and human-generated texts to enhance sentence embeddings," in *Findings of the Association for Computational Linguistics: EACL 2024*, Y. Graham and M. Purver, Eds. St. Julian's, Malta: Association for Computational Linguistics, Mar. 2024, pp. 624–638. [Online]. Available: https://aclanthology.org/2024.findings-eacl.43 II-A

[28] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé, "Distiller: Encrypted traffic classification via multimodal multitask deep learning," *Journal of Network and Computer Applications*, vol. 183-184, p. 102985, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804521000126 II-A

[29] J. Fu, O. Hagsand, and G. Karlsson, "Queuing behavior and packet delays in network processor systems," in *2007 15th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*, 2007, pp. 217–224. II-A

[30] G. Yakovlev, J. B. Rundle, R. Shcherbakov, and D. L. Turcotte, "Inter-arrival time distribution for the non-homogeneous poisson process," 2005. [Online]. Available: https://arxiv.org/abs/cond-mat/0507657 II-A

[31] F. T. Lima and V. M. Souza, "A large comparison of normalization methods on time series," *Big Data Research*, vol. 34, p. 100407, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2214579623000400 II-A

[32] Z. Zhao, H. Chen, J. Zhang, X. Zhao, T. Liu, W. Lu, X. Chen, H. Deng, Q. Ju, and X. Du, "Uer: An open-source toolkit for pre-training models," *EMNLP-IJCNLP 2019*, p. 241, 2019. III-B

[33] C. Toraman, E. H. Yilmaz, F. Şahinuç, and O. Ozcelik, "Impact of tokenization on language models: An analysis for turkish," *ACM Transactions on Asian and Low-Resource Language Information Processing*, vol. 22, no. 4, p. 1–21, Mar. 2023. [Online]. Available: http://dx.doi.org/10.1145/3578707 III-B

[34] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *International Conference on Information Systems Security and Privacy*, 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:4707749 III-B

[35] I. Sharafaldin, A. H. Lashkari, S. Hakak, and A. A. Ghorbani, "Developing realistic distributed denial of service (ddos) attack dataset and taxonomy," in *2019 International Carnahan Conference on Security Technology (ICCST)*, 2019, pp. 1–8. III-B

[36] G. Draper-Gil, A. H. Lashkari, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of encrypted and vpn traffic using time-related features," in *International Conference on Information Systems Security and Privacy*, 2016. [Online]. Available: https://api.semanticscholar.org/CorpusID:21535780 IV-A, II

[37] 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)*, 2017, pp. 712–717. IV-A, II

[38] S. Dadkhah, H. Mahdikhani, P. K. Danso, A. Zohourian, K. A. Truong, and A. A. Ghorbani, "Towards the development of a realistic multidimensional iot profiling dataset," *2022 19th Annual International Conference on Privacy, Security & Trust (PST)*, pp. 1–11, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:251763546 IV-A, II, V

[39] T. van Ede, R. Bortolameotti, A. Continella, J. Ren, D. J. Dubois, M. Lindorfer, D. Choffness, M. van Steen, and A. Peter, "FlowPrint: Semi-Supervised Mobile-App Fingerprinting on Encrypted Network Traffic," in *NDSS*. The Internet Society, 2020. IV-A, IV-A, IV-A, IV-A, II, V

[40] V. Hosu, F. Hahn, M. Jenadeleh, H. Lin, H. Men, T. Szirányi, S. Li, and D. Saupe, "The konstanz natural video database (konvid-1k)," in *2017 Ninth International Conference on Quality of Multimedia Experience (QoMEX)*, 2017, pp. 1–6. IV-A, IV-A, IV

[41] C. G. Bampis, Z. Li, A. K. Moorthy, I. Katsavounidis, A. Aaron, and A. C. Bovik, "Study of temporal effects on subjective video quality of experience," *IEEE Transactions on Image Processing*, vol. 26, no. 11, pp. 5217–5231, 2017. IV-A, IV-A, IV

[42] W. Zheng, C. Gou, L. Yan, and S. Mo, "Learning to classify: A flow-based relation network for encrypted traffic classification," in *Proceedings of The Web Conference 2020*, ser. WWW '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 13–22. [Online]. Available: https://doi.org/10.1145/3366423.3380090 IV-A

[43] A. Sun and E.-P. Lim, "Hierarchical text classification and evaluation," in *Proceedings 2001 IEEE International Conference on Data Mining*, 2001, pp. 521–528. IV-A

[44] L. Li, P. Chen, W. Lin, M. Xu, and G. Shi, "From whole video to frames: Weakly-supervised domain adaptive continuous-time qoe evaluation," *IEEE Transactions on Image Processing*, vol. 31, pp. 4937–4951, 2022. IV-A, IV-A, IV

[45] Y. Wu, S. Kan, M. Zeng, and M. Li, "Singularformer: learning to decompose self-attention to linearize the complexity of transformer," in *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, ser. IJCAI '23, 2023. [Online]. Available: https://doi.org/10.24963/ijcai.2023/493 IV-E

[46] R. Schatz, T. Hoßfeld, and P. Casas, "Passive youtube qoe monitoring for isps," in *2012 Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, 2012, pp. 358–364. V

[47] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013. [Online]. Available: https://arxiv.org/abs/1301.3781 V

[48] L. Gioacchini, M. Mellia, L. Vassio, I. Drago, G. Milan, Z. B. Houidi, and D. Rossi, "Cross-network embeddings transfer for traffic analysis," *IEEE Transactions on Network and Service Management*, vol. 21, pp. 2686–2699, 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID:264948139 V

[49] M. Ring, A. Dallmann, D. Landes, and A. Hotho, "Ip2vec: Learning similarities between ip addresses," in *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, 2017, pp. 657–666. V

[50] D. Cohen, Y. Mirsky, M. Kamp, T. Martin, Y. Elovici, R. Puzis, and A. Shabtai, "Dante: A framework for mining and monitoring darknet traffic," in *Computer Security – ESORICS 2020*, L. Chen, N. Li, K. Liang, and S. Schneider, Eds. Cham: Springer International Publishing, 2020, pp. 88–109. V

[51] L. Gioacchini, L. Vassio, M. Mellia, I. Drago, Z. B. Houidi, and D. Rossi, "i-darkvec: Incremental embeddings for darknet traffic analysis," *ACM Trans. Internet Technol.*, vol. 23, no. 3, aug 2023. [Online]. Available: https://doi.org/10.1145/3595378 V

[52] Z. B. Houidi, R. Azorin, M. Gallo, A. Finamore, and D. Rossi, "Towards a systematic multi-modal representation learning for network data," in *Proceedings of the 21st ACM Workshop on Hot Topics in Networks*, ser. HotNets '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 181–187. [Online]. Available: https://doi.org/10.1145/3563766.3564108 V

[53] I. Drago, M. Mellia, and A. D'Alconzo, *Big Data in Computer Network Monitoring*. Cham: Springer International Publishing, 2018, pp. 1–8. [Online]. Available: https://doi.org/10.1007/978-3-319-63962-8_26-1 VI

[54] I. Parvez, A. Rahmati, I. Guvenc, A. I. Sarwat, and H. Dai, "A survey on low latency towards 5g: Ran, core network and caching solutions," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 3098–3130, 2018. VI

[55] S. Wang, B. Z. Li, M. Khabsa, H. Fang, and H. Ma, "Linformer: Self-attention with linear complexity," 2020. VI

[56] Y. Sun, X. Li, K. Dalal, J. Xu, A. Vikram, G. Zhang, Y. Dubois, X. Chen, X. Wang, S. Koyejo, T. Hashimoto, and C. Guestrin, "Learning to (learn at test time): Rnns with expressive hidden states," 2024. [Online]. Available: https://arxiv.org/abs/2407.04620 VI