electronics

MDPI

Article

Beyond Byte-Level Modeling: Structure-Aware and Adaptive Traffic Classification for Encrypted Networks

Gyeong-Min Yu [1], Yoon-Seong Jang [2], Ju-Sung Kim [2], Seung-Woo Nam [2], Ji-Min Kim [1], Yang-Seo Choi [3] and Myung-Sup Kim [1,*]

[1] Department of Computer Science and Software Engineering, Korea University, Sejong-si 30019, Republic of Korea; rudals2710@korea.ac.kr (G.-M.Y.); illiard1209@korea.ac.kr (J.-M.K.)
[2] Department of Computer and Information Science, Korea University, Sejong-si 30019, Republic of Korea; brave1094@korea.ac.kr (Y.-S.J.); jsung0514@korea.ac.kr (J.-S.K.); nam131119@korea.ac.kr (S.-W.N.)
[3] Department of Cyber Security Research Division, Electronics and Telecommunications Research Institute, Daejeon 34129, Republic of Korea; yschoi92@etri.re.kr
* Correspondence: tmskim@korea.ac.kr

Abstract

The widespread adoption of encryption protocols such as TLS 1.3 has significantly reduced the visibility of packet payloads, limiting the effectiveness of traditional traffic analysis methods. Recent deep learning approaches attempt to learn representations directly from raw byte sequences; however, in encrypted environments, byte-level patterns often exhibit high entropy and unstable ordering, raising concerns about their reliability. In this work, we revisit the roles of content and structural information in traffic classification and argue that effective modeling should move beyond content-only representations. We propose a structure-aware framework that models hierarchical relationships across fields, layers, and sessions while representing byte information using compact, permutation-invariant summaries. In addition, we introduce a hierarchical shuffle pretraining strategy to capture relational dependencies and an adaptive inter-level gating mechanism to dynamically integrate multi-level representations. Extensive experiments on multiple datasets with varying levels of encryption demonstrate that byte-level sequential patterns are not always essential, while structural information provides consistent complementary cues. Furthermore, the importance of different structural levels varies across datasets, highlighting the need for adaptive multi-level modeling. The proposed method achieves strong performance across diverse datasets, including highly encrypted traffic, while maintaining robustness under domain shifts and limited data scenarios. These results suggest that combining compact content representations with structural context and adaptive integration is a promising direction for encrypted traffic analysis.

Keywords: encrypted traffic classification; network traffic analysis; structural representation learning; hierarchical pretraining; protocol structure modeling

Academic Editors: Maria Liz Crespo, Alexander Gegov, Patrick Siarry, Xin Xu, Xin Yuan and Kui Jiang

Received: 30 March 2026
Revised: 17 April 2026
Accepted: 23 April 2026
Published: 25 April 2026

1. Introduction

The widespread adoption of encrypted communication protocols such as TLS 1.3 has fundamentally changed the nature of network traffic analysis. Modern protocols encrypt most application-layer payloads as well as several protocol extensions, substantially reducing the amount of observable plaintext information in network traffic [1]. As a result, traditional content-based inspection techniques such as deep packet inspection (DPI), which rely on direct access to packet payloads [2], have become increasingly ineffective for tasks

such as traffic classification, anomaly detection, and network monitoring. In practice, the information available for analysis is now largely limited to coarse flow-level statistics, timing patterns, and minimal unencrypted metadata [3].

To cope with this limited visibility, recent approaches have increasingly adopted deep learning models that operate directly on raw packet byte sequences. CNN-based [4,5], LSTM-based [6], and Transformer-based [7] architectures have been applied to learn representations from encrypted traffic and have reported promising empirical performance [8–11]. However, this approach introduces an inherent challenge: under strong encryption, most byte values within protocol fields are produced by cryptographic transformations and therefore exhibit high entropy and near-random distributions. As a result, fine-grained byte-level ordering patterns may become less reliable, raising questions about what information such representations capture and how well they generalize across different environments.

In practice, byte-level patterns in encrypted traffic can vary across datasets and implementations, which may cause models to depend on dataset-specific regularities. While such representations can achieve high accuracy in controlled settings, their robustness and interpretability remain limited, particularly when byte-level signals are heavily influenced by encryption.

These observations suggest that relying solely on byte content may not be well aligned with the properties of encrypted traffic. A natural question then arises: what information remains stable and observable under encryption? We argue that useful signals can still be found in the protocol structure, such as field boundaries, length information, and the hierarchical organization of packets, layers, and sessions. For example, in TLS, message formats—including field boundaries, length fields, and relationships between protocol elements—remain explicitly defined and observable even when field contents are encrypted [1]. Such structural characteristics provide additional context that is less affected by encryption and can complement content-based representations.

Based on this observation, we argue that effective traffic modeling should move beyond content-only representations by incorporating structural information. Rather than replacing content entirely, our goal is to combine compact content representations with structural context that remains observable under encryption. To this end, we propose a structure-aware traffic representation framework that preserves protocol boundaries and represents each field as a structural token. Each token summarizes intra-field byte distributions using permutation-invariant statistics (e.g., mean and maximum pooling), enabling the model to reduce sensitivity to unstable byte ordering while retaining useful distributional information.

To further capture relationships across different structural levels, we introduce a Hierarchical Shuffle Pretraining strategy. The key idea is to train the model to recover the correct ordering of structural units (field, layer, and packet) from shuffled inputs, thereby encouraging the learning of hierarchical dependencies. In addition, we employ a hierarchical attention pooling mechanism with inter-level gating to adaptively integrate representations across multiple structural levels.

We evaluate the proposed framework on multiple datasets spanning diverse encryption scenarios, including ISCX VPN 2016, ISCX TOR 2016, CSTNET TLS1.3, and three large-scale private datasets collected under different environments. In particular, we construct two datasets (Private-1 and Private-2) that contain the same application classes but differ in network conditions, enabling evaluation under distribution shifts.

Unlike many prior studies that rely on auxiliary identifiers such as IP addresses or port numbers—even when partially masked—we restrict the input to application-layer (L7) information only. In this context, although application-layer payloads are largely encrypted (e.g., TLS 1.3), not all protocol information becomes inaccessible. Certain structural sig-

nals—such as field boundaries, length information, and partially observable metadata (e.g., handshake-related fields or protocol headers)—remain visible and can be extracted through protocol parsing. Experimental results demonstrate that the proposed approach achieves strong classification performance under this constrained setting, yielding up to 24.5% absolute improvement over the Transformer-based baseline ET-BERT [8] in highly encrypted L7 scenarios (e.g., TLS 1.3), while leveraging both content summaries and structural context.

The main contributions of this work are as follows:

- Structure-aware modeling for encrypted traffic.We analyze the limitations of byte-centric representations through controlled perturbation experiments and show that content signals alone are not sufficient under encryption. We demonstrate that structural information provides complementary cues that improve robustness when combined with content representations.
- Hierarchical shuffle pretraining for relational structure learning. We introduce a self-supervised pretraining strategy that learns dependencies across hierarchical protocol units, including field ordering, layer composition, and packet organization, without requiring labeled data.
- Adaptive integration of multi-level representations. We design an inter-level gating mechanism that dynamically integrates representations from different hierarchy levels, allowing the model to adapt to dataset-specific characteristics.
- Robust performance under encrypted L7 traffic. We demonstrate that the proposed approach achieves strong and consistent performance across diverse datasets, including highly encrypted traffic, while maintaining robustness under domain shifts.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 describes the proposed framework. Section 4 presents experimental results. Section 5 concludes the paper.

2. Related Work

Recent advances in encrypted traffic analysis have led to a variety of representation learning approaches for modeling network traffic. These methods differ in their representation units, the extent to which structural information is incorporated, and the design of their learning objectives. In this work, we organize prior studies into three categories: byte-centric approaches , partially structural approaches, and structurally aware approaches.

In this work, we define structure as the protocol-defined organization of traffic, consisting of (i) hierarchical units (fields, layers, packets, and sessions), (ii) unit-level attributes such as field length and positional offsets, and (iii) relational constraints governing ordering and containment among these units. Based on this perspective, Table 1 summarizes representative approaches according to their ability to model these structural elements. Specifically, O, △, and X denote full, partial, and no structural modeling capability, respectively. Prior work can be broadly categorized into three groups: byte-centric approaches, partially structural approaches, and structurally aware approaches. We evaluate structural modeling capability along three dimensions: Unit, Attr, and Rel.

Unit refers to whether protocol-defined hierarchical units (field, layer, packet, session) are explicitly represented as modeling primitives in the input representation. We assign O only when multiple protocol levels are jointly and explicitly encoded. △ indicates that only a subset of units (e.g., field or packet) is partially represented. X denotes that no protocol-aligned units are modeled. Importantly, segmentation based on fixed-length chunks or patches (e.g., burst or patch embeddings) is not considered hierarchical unit modeling unless it aligns with protocol-defined boundaries. Attr refers to whether structural attributes—such as field length, positional offset, or direction—are explicitly encoded as semantic features. O indicates explicit and semantically meaningful encoding. △ indicates

indirect or partial use (e.g., positional embeddings without protocol alignment). X indicates no structural attribute modeling. Rel refers to whether structural relationships among units—such as ordering, hierarchy, or containment—are explicitly modeled or directly enforced by the learning objective. O is assigned only when such relationships are directly learned through dedicated objectives (e.g., order recovery across hierarchical units). △ indicates implicit learning through sequence modeling or classification tasks (e.g., order prediction or autoregressive modeling). X indicates that structural relationships are not modeled. Notably, tasks that capture co-occurrence or group membership (e.g., Same Burst Prediction) are not considered relational modeling.

Table 1. Comparison of traffic representation learning approaches based on structural modeling capability.

Model	Pretraining Objective	Structure Modeling		
		Unit	Attr	Rel
Byte-centric approaches				
PERT [12]	MLM (byte reconstruction)	X	X	X
ETBERT [8]	MLM + SBP	X	X	X
YaTC [13]	MAE (2D patch reconstruction)	X	X	X
Flow-mae [14]	Masked patch reconstruction	X	X	X
Netmamba [15]	MAE/stride reconstruction	X	△	X
Partially structural approaches				
PTU [10]	MLM + HIP/FIP	X	△	X
Trafficformer [9]	MBM + SODF	X	X	△
Flowletformer [11]	Masked field + flowlet order	△	△	△
LiM [16]	Feature-based representation	△	O	X
Structurally aware approaches				
GBC [17]	Autoregressive protocol modeling	△	△	△
Nethira [18]	Sequence + field masking	X	X	△
Netfound [19]	MLM-based representation	△	△	X
Mletc [20]	Multi-level reconstruction	△	△	X
Proposed				
Ours	Hierarchical shuffle + gating	**O**	**O**	**O**

2.1. Byte-Centric Approaches

Early traffic representation learning models primarily operate on byte-derived tokens, treating network traffic as sequences of bytes similar to textual data. These approaches operate purely on byte-derived tokens without explicitly defining protocol units such as fields, layers, or packets. Therefore, all structural components (Unit, Attr, Rel) are marked as X in Table 1. Even when segmentation strategies such as burst- or patch-based embeddings are used (e.g., [13,14]), these do not align with protocol-defined hierarchical units. PERT [12] applies masked language modeling (MLM) on byte tokens extracted from packet payloads. ET-BERT [8] extends this paradigm by introducing burst-level tokenization and a Same Burst Prediction (SBP) objective to capture contextual relationships among packets. YaTC [13] transforms packet byte sequences into two-dimensional matrices and applies masked autoencoder (MAE) training, enabling image-like representations of traffic data. Subsequent works such as Flow-MAE [14] and NetMamba [15] further improve scalability and efficiency for modeling long byte sequences. These approaches have demonstrated strong performance, particularly due to their ability to learn representations directly from raw traffic. However, their representation units remain byte-derived tokens, and protocol-defined boundaries such as fields or layers are not explicitly modeled. As a result, structural information is only implicitly captured through byte patterns.

2.2. Partially Structural Approaches

To address this limitation, several studies incorporate additional structural signals derived from temporal dynamics or protocol headers. Some approaches focus on temporal relationships between packets. These methods incorporate certain structural signals, such as timing, ordering, or header attributes. However, such information is not explicitly modeled as hierarchical protocol units, nor is it jointly integrated across multiple levels. As a result, they are categorized as partially structural (△) in Table 1. PTU [10] introduces Historical Interval Prediction (HIP) and Future Interval Prediction (FIP) to model packet inter-arrival times, while TrafficFormer [9] learns ordering relationships among packets using sequence order detection objectives. Other methods incorporate protocol-level information. FlowletFormer [11] introduces field-level tokenization for header fields, and feature-based approaches such as LiM [16] utilize protocol-defined header attributes as input features. While these approaches integrate useful structural cues—such as timing, ordering, or header attributes—the structural information is typically incorporated in a limited or task-specific manner. In particular, hierarchical relationships across fields, layers, packets, and sessions are not jointly modeled within a unified representation.

2.3. Structurally Aware Approaches

More recent studies aim to explicitly incorporate protocol structure into representation learning. GBC [17] models protocol-aware sequences using autoregressive generation, which captures ordering patterns within a single level. Nethira [18] introduces reconstruction objectives across different granularities, while MLPT [20] constructs multi-level representations. These approaches demonstrate that incorporating structural information can improve representation quality. However, their learning objectives remain primarily focused on reconstructing observed token values. As a result, structural relationships—such as ordering or hierarchy—are not directly enforced but are instead learned implicitly through value reconstruction or sequence modeling. In contrast, the proposed method explicitly models structural relationships by introducing a pretraining objective that directly enforces hierarchical ordering constraints. This allows the model to learn not only structural representations but also the dependencies among them.

2.4. Limitations and Motivation

The above categorization highlights key differences among existing approaches. Based on this comparison, several limitations can be identified. First, byte-centric methods do not explicitly encode protocol-defined structure. Second, partially structural approaches incorporate structural signals but do not model them in a unified hierarchical framework. Third, even structurally aware methods often rely on value reconstruction objectives, where structural relationships are not directly enforced.

In encrypted environments, where many byte values are generated by cryptographic transformations, the reliability of fine-grained byte-level patterns may be reduced. This motivates the need to explore representations that can better leverage information that remains observable under encryption. Importantly, our goal is not to replace content-based representations but to complement them with structural information. To this end, we propose a structure-aware framework that explicitly models hierarchical relationships across fields, layers, and packets, while representing content using compact summaries. In addition, we introduce a pretraining objective that encourages the model to capture structural relationships by recovering shuffled hierarchical orders.

By combining content summaries with structural context, the proposed approach provides a unified framework for capturing both value-based and relational information in network traffic. While prior approaches either do not model structural relationships

or capture them implicitly, our method introduces an explicit mechanism to learn such dependencies. This distinction is particularly relevant in encrypted environments, where structural cues can provide complementary information to content-based representations. Overall, by jointly considering units, attributes, and relationships, the proposed framework aims to improve the robustness and expressiveness of traffic representations.

3. Methodology

This section presents a structure-aware representation learning framework for encrypted traffic analysis. Structural information provides global constraints defined by protocol organization, while content representations provide evidence. By integrating both aspects, the model aims to learn representations that remain robust under encryption and domain variation. Unlike conventional self-supervised approaches that focus on recovering masked tokens, the proposed framework introduces a pretraining objective that explicitly targets structural relationships across protocol hierarchy levels. Specifically, the proposed pretraining tasks explicitly model ordering constraints at field ordering within a layer, layer ordering within a packet, and packet ordering within a session.

The framework consists of four components: dataset preprocessing, hierarchical tokenization, structure-aware input representation, and hierarchical shuffle pretraining. The learned representations are subsequently used for downstream traffic classification via hierarchical attention pooling, as illustrated in Figure 1.

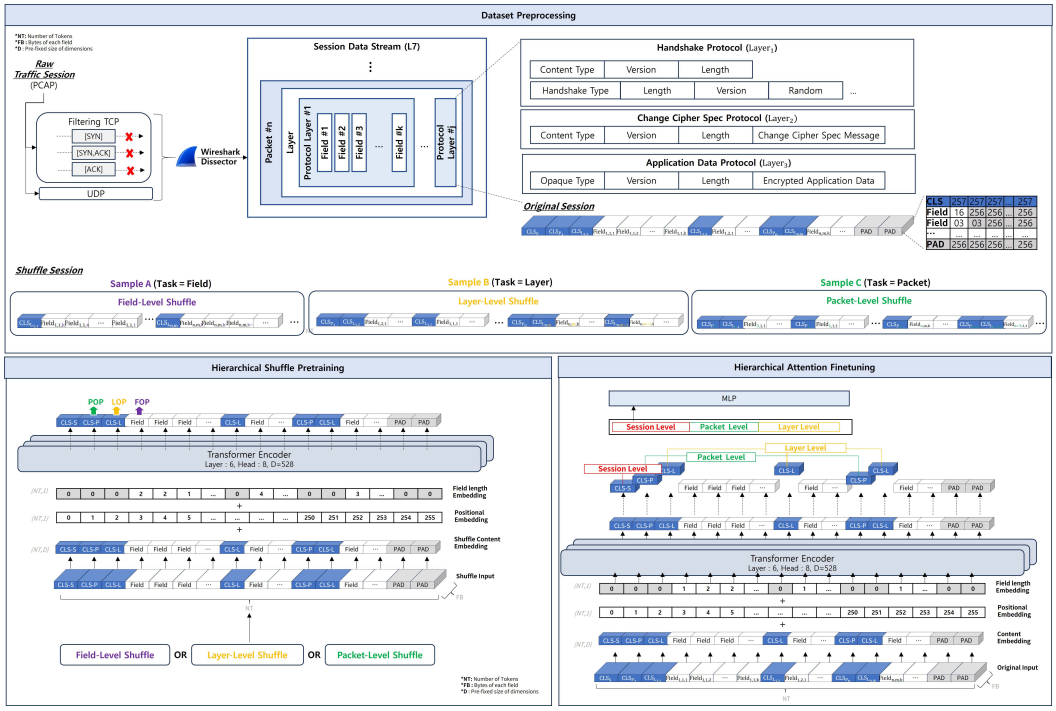

Figure 1. Overview of the proposed framework. (**Top**) Dataset preprocessing and tokenization. (**Bottom-left**) Hierarchical Shuffle Pretraining with three shuffle tasks. (**Bottom-right**) Fine-tuning with hierarchical attention pooling.

3.1. Dataset Preprocessing

Before training, preprocessing is applied to remove patterns that may induce shortcut learning. For TCP traffic, packets corresponding to the three-way handshake (`SYN`, `SYN-ACK`, and `ACK`) are excluded. If included, the model can exploit this fixed ordering pattern to solve the pretraining tasks (e.g., packet order prediction) without learning general structural dependencies. In particular, the presence of such predictable sequences allows the model to rely on trivial positional cues rather than inferring valid ordering relationships across

diverse protocol interactions. By removing handshake packets, we reduce this shortcut signal and encourage the model to learn more generalizable structural patterns beyond fixed protocol initialization sequences.

3.2. Hierarchical Tokenization

Network traffic naturally follows a hierarchical structure defined by protocol specifications, which we formalize as hierarchical units (sessions, packets, layers, and fields), as illustrated in Figure 2. A session S consists of packets, each packet consists of protocol layers, and each layer consists of fields:

$$S = \{P_1, \ldots, P_n\}, \quad P_i = \{L_{i,1}, \ldots, L_{i,m}\}, \quad L_{i,j} = \{F_{i,j,1}, \ldots, F_{i,j,k}\}. \tag{1}$$

To obtain this hierarchical structure, raw packets are parsed using a protocol dissector (Wireshark, version 4.4.8),which decomposes each packet into protocol layers and fields according to standard protocol specifications. This enables consistent extraction of field-level units across diverse traffic types. Each extracted field is treated as an individual token. Importantly, even encrypted or opaque fields (e.g., random values, opaque segments, or application-layer data) are preserved as single tokens without further decomposition. This design allows the model to retain structural consistency while operating under fully encrypted settings. To explicitly encode this hierarchy, we introduce multi-level aggregation tokens: CLS_S (session), CLS_P (packet), and CLS_L (layer). Each token aggregates information within its corresponding structural scope. While protocol parsing is used to obtain hierarchical units, the model itself does not rely on protocol-specific identifiers such as field names or header types. Instead, structure is represented implicitly through token organization, allowing generalization across heterogeneous environments.

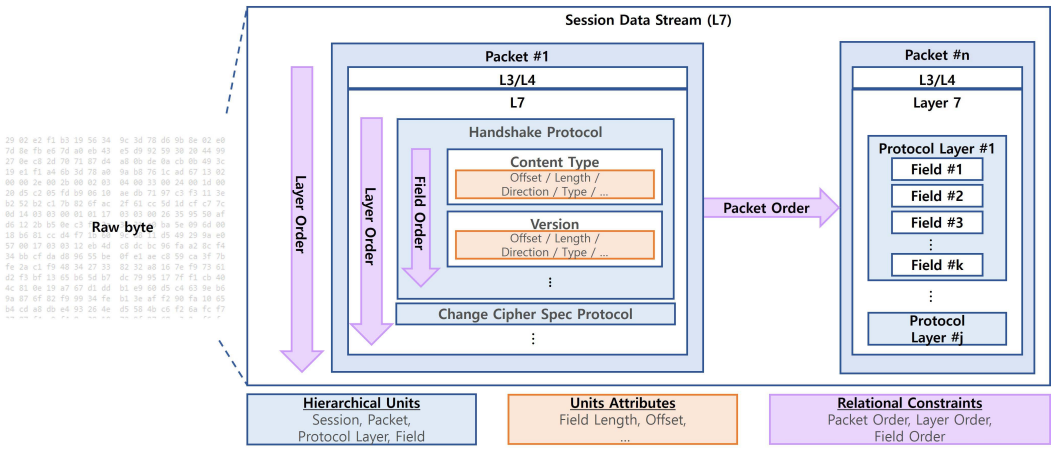

Figure 2. Structural elements (Unit, Attr, Rel) of network traffic used in the proposed framework.

The input sequence is constructed by interleaving hierarchical CLS tokens and field tokens. The total number of tokens is limited to 256 to ensure computational efficiency and compatibility with Transformer-based architectures. For each field, raw byte sequences are truncated or padded to a maximum length of 1400 bytes, which is determined based on the typical Maximum Transmission Unit (MTU) size. This design ensures that the majority of packet contents can be represented while maintaining a consistent input dimension. Unlike fixed-length segmentation approaches, this representation aligns tokens with protocol-defined boundaries rather than arbitrary byte chunks.

3.3. Structure-Aware Input Representation

Each token embedding is composed of three components:

$$e = e_{\text{content}} + e_{\text{position}} + e_{\text{length}}. \tag{2}$$

3.3.1. Content Embedding

Each field consists of a variable-length byte sequence. To construct a fixed-dimensional representation, byte sequences are first padded to a maximum length. However, padding positions do not carry semantic information and must be excluded from aggregation.

Let $\mathbf{x} = \{x_1, \ldots, x_L\}$ denote the byte sequence of a field, and let $\mathbf{m} \in \{0,1\}^L$ be a binary mask indicating valid (non-padding) positions. The content embedding is computed using masked permutation-invariant aggregation:

$$e_{\text{mean}} = \frac{1}{\sum_i m_i} \sum_{i=1}^{L} m_i \cdot x_i, \tag{3}$$

$$e_{\text{max}} = \max_{i:m_i=1} x_i, \tag{4}$$

$$e_{\text{content}} = e_{\text{mean}} \oplus e_{\text{max}}, \tag{5}$$

where $\oplus$ denotes concatenation.

This masked aggregation ensures that padding values do not influence the representation, allowing the model to extract robust statistics from variable-length inputs. Rather than modeling exact byte ordering, this approach captures distributional characteristics of byte values. This design is motivated by the observation that, in encrypted traffic, byte sequences often exhibit high entropy, making local ordering patterns unstable and less informative. While certain protocol fields (e.g., low-entropy headers such as TLS record types) may retain partially observable sequential patterns, we intentionally adopt a permutation-invariant design to improve robustness and generalization under encryption. This design reflects a trade-off that prioritizes robustness over fine-grained sequence modeling in encrypted environments. Therefore, permutation-invariant aggregation provides a more robust representation than order-sensitive encoders. Importantly, this does not eliminate the role of content. Instead, content representations provide compact summaries, while structural components provide contextual constraints.

3.3.2. Positional Embedding

Each token is assigned a global positional index, which is encoded as a positional embedding (e_{position}). Because hierarchical CLS tokens define structural boundaries, positional embeddings implicitly encode hierarchical placement within the session.

3.3.3. Field Length Embedding

In addition to content representations, we incorporate structural attributes of network traffic, as illustrated in Figure 2. Specifically, unit-level attributes such as field length and positional offset (corresponding to the Units Attribute in Figure 2) are leveraged as complementary signals that remain observable under encryption. Field length is encoded as a learnable embedding (e_{length}). Length information often reflects protocol-level structure (e.g., TLS record sizes) and remains observable even under encryption. This allows the model to explicitly encode structural attributes alongside content summaries.

3.4. Hierarchical Shuffle Pretraining

The structure-aware input representations described above are first processed by a shared Transformer encoder, which serves as the backbone of the proposed framework. The encoder consists of 6 layers with 8 attention heads and a hidden dimension of 528. The feed-forward network dimension is set to 2112, with GELU activation, pre-layer normalization, and a dropout rate of 0.1. Each input token is represented in a 528-dimensional space. Content embeddings are obtained by mapping each byte to a 32-dimensional vector, followed by mean and max pooling across valid positions. The resulting 64-dimensional vector is projected to 528 dimensions and combined with positional and field-length embeddings. During pretraining, task-specific prediction heads are attached to the shared encoder. Each head consists of a two-layer MLP ($\text{Linear}(528, 128) \rightarrow \text{ReLU} \rightarrow \text{Dropout} \rightarrow \text{Linear}(128, \text{num\_classes})$).

To explicitly model structural relationships, we propose Hierarchical Shuffle Pretraining. In addition to modeling hierarchical units and structural attributes, we explicitly capture relational constraints among protocol units, as illustrated in Figure 2. To this end, we propose Hierarchical Shuffle Pretraining, which is designed to learn ordering and containment relationships across hierarchical units (corresponding to the Rel component in Figure 2). Unlike generic permutation-based objectives, each shuffle operation is restricted within protocol-defined boundaries (e.g., fields within a layer, layers within a packet). This ensures that the model learns valid structural constraints rather than arbitrary sequence patterns. Specifically, the proposed pretraining tasks explicitly enforce ordering constraints at each hierarchy level: (i) field ordering within a layer, (ii) layer ordering within a packet, and (iii) packet ordering within a session.

Three tasks are defined:

- Field Order Prediction (FOP).
  Given a shuffled sequence of fields within a layer, the model predicts the original position (offset) of each field. Formally, let $\{F_1, \ldots, F_k\}$ denote the fields in a layer. After random shuffling, the model outputs a probability distribution over k possible positions for each field.
- Layer Order Prediction (LOP).
  Given a shuffled sequence of layers within a packet, the model predicts the original position of each layer. Each layer representation is mapped to a discrete offset corresponding to its original ordering within the packet.
- Packet Order Prediction (POP).
  Given a shuffled sequence of packets within a session, the model predicts the original packet ordering. This task captures directional and temporal dependencies at the session level.

All tasks share a common Transformer encoder, while task-specific prediction heads are used to map representations to offset distributions. Each prediction head performs multi-class classification over possible positions within the corresponding hierarchy level. The loss for each task is defined using cross-entropy:

$$\mathcal{L}_{\text{field}} = \text{CE}(\hat{y}^{(f)}, y^{(f)}), \quad \mathcal{L}_{\text{layer}} = \text{CE}(\hat{y}^{(l)}, y^{(l)}), \quad \mathcal{L}_{\text{packet}} = \text{CE}(\hat{y}^{(p)}, y^{(p)}), \tag{6}$$

and the total loss is:

$$\mathcal{L} = \lambda_f \mathcal{L}_{\text{field}} + \lambda_l \mathcal{L}_{\text{layer}} + \lambda_p \mathcal{L}_{\text{packet}}. \tag{7}$$

Importantly, the supervision signals are obtained in a self-supervised manner: the original ordering before shuffling is used as the ground-truth offset label, requiring no external annotations. Because shuffled configurations often violate protocol-consistent structures,

correct prediction requires learning hierarchical dependencies rather than relying on local token cues alone. This distinguishes the proposed objective from generic permutation recovery, as it enforces structure-aware learning aligned with protocol organization.

3.5. Hierarchical Attention Fine-Tuning

During fine-tuning, token representations $\mathbf{h}_i$ from the Transformer encoder are aggregated into hierarchy-level representations using attention pooling:

$$\mathbf{v}_\ell = \sum_{i \in C_\ell} \alpha_i^{(\ell)} \mathbf{h}_i, \tag{8}$$

where C_ℓ denotes the set of tokens belonging to hierarchy level ℓ (layer, packet, or session).

The attention weights are computed using a learnable scoring function:

$$s_i^{(\ell)} = \mathbf{w}_\ell^\top \mathbf{h}_i, \quad \alpha_i^{(\ell)} = \frac{\exp(s_i^{(\ell)}) \cdot m_i^{(\ell)}}{\sum_j \exp(s_j^{(\ell)}) \cdot m_j^{(\ell)}}, \tag{9}$$

where $\mathbf{w}_\ell$ is a learnable parameter and $m_i^{(\ell)}$ is a binary mask that selects tokens belonging to level ℓ.

This attention pooling aggregates tokens within each hierarchy level into a single representation by emphasizing more informative tokens.

The resulting level-wise representations are then combined using an inter-level gating mechanism:

$$\mathbf{g} = \sum_\ell g_\ell \mathbf{v}_\ell. \tag{10}$$

Unlike standard cross-attention, which models interactions across tokens, the proposed mechanism first aggregates tokens within each structural level and then performs gating across levels. This separation enables explicit modeling of hierarchical structure and allows the model to adaptively select the most informative level for each input.

Overall, the proposed framework does not rely solely on structural information, but integrates structural constraints with content-derived summaries. By adaptively combining multi-level representations, the model achieves more robust and expressive traffic representations across encrypted and heterogeneous environments.

4. Experiments

This section evaluates the proposed method. All experiments are conducted using only application-layer (L7) information to assess model generalization under constrained visibility. We first describe the datasets and preprocessing procedures, followed by the experimental setup and baseline models. We then present results including baseline comparisons, analysis of the pretraining strategy, evaluation of hierarchical pooling, and structure-versus-content ablation studies.

4.1. Datasets

To evaluate the robustness of the proposed method across diverse traffic environments, the authors used three datasets collected by the authors and three public benchmarks widely used in application traffic classification research.

The public datasets include ISCX VPN 2016 [21], ISCX Tor 2016 [22], and CSTNET–TLS 1.3 [8], which are commonly used benchmarks in deep-learning-based traffic classification studies and enable direct comparison with previously proposed approaches. The private datasets were collected in a controlled network environment using a traffic mirroring system. Traffic generated while executing applications on a client PC was captured via mir-

roring, while process-level collection using Microsoft Network Monitor enabled accurate session-level labeling for each application. To minimize noise, only a single application was executed during data collection, preventing interference from concurrent processes. As a result, the datasets provide clean, low-noise traffic with reliable ground-truth labels and reflect a wide range of modern applications. To improve reproducibility, we provide detailed dataset statistics (Table 2), including class distribution and encryption ratios. In addition, three datasets collected by the authors are used to evaluate the proposed method in realistic network environments, referred to as Private-1, Private-2, and Private-3. Private-1 and Private-2 are collected from different environments while sharing the same set of 48 application classes, enabling evaluation of cross-environment generalization across service categories, including gaming, peer-to-peer services, social networking, streaming, and web services, collected from desktop applications, browser-based services, and Android applications. Private-3 significantly expands the number of application classes to 300 and includes recently emerging applications such as ChatGPT, Gemini, MS Copilot, and Perplexity, allowing for evaluation under up-to-date (2025) and large-scale traffic conditions.

Table 2. Summary of datasets used in the experiments. Y and N indicate Yes (public) and No (private), respectively.

Dataset	Public	Class	Sessions (Raw)	Sessions (Filtered)	Encryption
Private-1	N	48	71,841	50,166	∼27%
Private-2	N	48	70,379	32,402	∼47%
Private-3	N	300	399,722	351,536	∼88%
ISCX VPN 2016 [21]	Y	16	187,336	4802	∼1%
ISCX Tor 2016 [22]	Y	14	57,605	26,165	∼10%
CSTNET–TLS 1.3 [8]	Y	120	46,372	46,372	∼99%

Figure 3 presents the protocol distribution across datasets, grouping application protocols into six representative categories: TLS, HTTP, P2P, STUN, QUIC, and Others. CSTNET–TLS 1.3 [8] consists almost entirely of encrypted TLS traffic, while Private-3 also exhibits a high proportion of TLS (approximately 88%). This reflects the increasing prevalence of encrypted protocols in modern network environments, although the distribution varies across datasets.

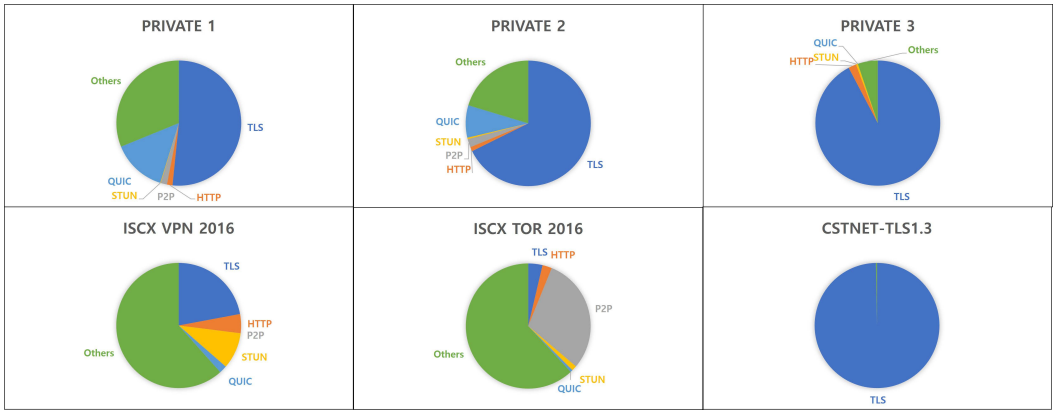

Figure 3. Protocol distribution across datasets, with application protocols grouped into six categories. This reflects the protocol composition used in the preprocessing stage, where protocol parsing is employed to construct field-level tokens. (Green indicates Others, sky blue indicates QUIC, orange indicates HTTP, yellow indicates STUN, and gray indicates P2P.)

4.2. Dataset Cleaning

Before training and evaluation, all datasets undergo a cleaning process to retain only sessions containing meaningful application-level interactions. First, lower-layer control protocols are removed: ARP, RARP, ICMP, and ICMPv6 are excluded, and only IPv4-based TCP and UDP sessions are retained. Next, incomplete sessions and flows without application-layer payload are discarded; for TCP traffic, sessions missing SYN or FIN packets are removed. Finally, background services unrelated to the target application classes are filtered out. Protocols such as DNS are excluded to prevent the model from exploiting superficial cues (e.g., textual matching of domain names) and to encourage learning intrinsic flow-level structural patterns. Similarly, NTP and SSDP are removed as they reflect system-level operations rather than application-specific behavior.

Table 2 summarizes the dataset statistics before and after cleaning. As shown in the table, the number of sessions is substantially reduced in several datasets, particularly [21], where a large portion of flows correspond to non-application or incomplete traffic. In contrast, datasets such as CSTNET–TLS 1.3 [8] remain unchanged, as they already consist of well-formed application-layer sessions.

4.3. Experimental Setup

All datasets are split into training and test sets at an 8:2 ratio. Pretraining is conducted on Private-3 due to its scale and diversity, followed by fine-tuning on each downstream dataset. Detailed training configurations are summarized in Table 3. Unless otherwise specified, all experiments follow the same training configuration.

Table 3. Training and optimization settings.

Parameter	Value
Train/Test Split	8:2
Pretraining Dataset	Private-3
Pretraining Epochs	30
Fine-tuning Epochs	50
Optimizer	AdamW
Learning Rate	1×10^{-5}
Batch Size	32

Each traffic session is represented as a hierarchical token sequence with a maximum length of 256 tokens, where each token corresponds to a protocol field $F_{i,j,k}$ representing up to 1400 bytes. Figure 4 illustrates the average token composition per session across datasets.

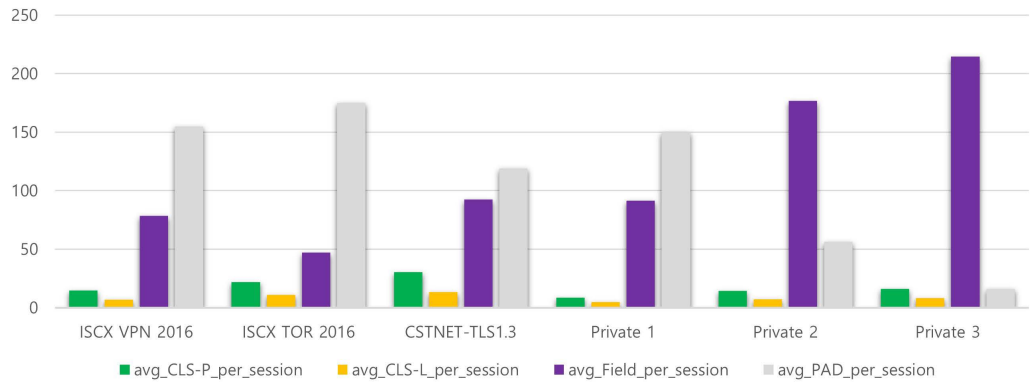

Figure 4. Average hierarchical token composition per session across datasets, including packet-level (CLS-P), layer-level (CLS-L), field, and padding tokens.

The model is implemented in PyTorch (version 2.2.2) with CUDA 12.1 and trained on a server equipped with an NVIDIA A100 80 GB PCIe GPU. Performance is evaluated using Accuracy (AC) [23] and Macro F1-score (F1) [24], with the latter emphasizing balanced performance under class imbalance.

4.4. Baseline Methods

The proposed model is compared with four representative deep learning-based traffic classification approaches: ET-BERT [8], YaTC [13], TrafficFormer [9], and NetFound [19]. These models are selected as representative Transformer-based approaches with publicly available implementations, ensuring fair and reproducible comparisons. For each baseline, two input settings are considered: (1) L7-only, where only application-layer payloads are used, and (2) Header + Payload (Masked), where L3/L4 headers are included but IP addresses and port numbers are masked. Pretrained weights for all baseline models are obtained from their publicly available implementations. Importantly, we do not modify the original architectures or pretraining procedures of the baseline models, and the proposed structure-aware framework is applied only to our method. All models are fine-tuned under a unified training configuration to ensure controlled and fair comparison, following the settings described in Section 3.

4.5. Baseline Comparison

Table 4 reports the classification performance using only application-layer (L7) information. The proposed method consistently achieves the highest accuracy across all datasets, demonstrating its strong discriminative capability even without relying on lower-layer protocol information. On the VPN 2016 dataset [21], although the proposed method attains the highest accuracy (90.0%), the F1 score (79.8%) is relatively low compared to the accuracy. This discrepancy suggests the presence of class imbalance or uneven per-class performance, as F1 score reflects the harmonic mean of precision and recall and is more sensitive to misclassification in minority classes. Notably, on the TLS 1.3 dataset [8]—where more than 99% of payload content is encrypted—the proposed method achieves 95.0% accuracy and 93.8 F1 score. This result indicates that the model remains highly effective even in strongly encrypted environments, supporting the claim that it does not rely solely on raw byte content. Furthermore, for the Private-1 and Private-2 datasets, which consist of the same application classes collected under different environments, baseline models exhibit noticeable performance variations (up to approximately 10% in accuracy). In contrast, the proposed method maintains relatively consistent accuracy, suggesting improved robustness to domain shifts and environmental differences.

Table 4. Classification performance (%) across datasets using only application-layer (L7) information. Bold indicates the best result per column.

Method	ISCX VPN 2016 [21]		ISCX Tor 2016 [22]		CSTNET TLS1.3 [8]		Private-3		Private-2		Private-1	
	AC	F1	AC	F1	AC	F1	AC	F1	AC	F1	AC	F1
ET-BERT [8]	82.0	84.3	86.7	79.6	69.5	68.9	32.7	37.9	36.2	36.5	48.6	46.7
YaTC [13]	88.7	**88.4**	94.9	**94.8**	72.8	74.4	68.8	68.0	69.5	69.3	48.1	45.1
TrafficFormer [9]	79.6	71.6	86.7	83.2	69.5	69.1	33.0	37.2	35.2	34.8	47.4	42.8
NetFound [19]	59.3	47.2	73.6	69.0	24.8	21.3	23.2	16.8	35.4	32.1	48.2	44.9
Proposed	**90.0**	79.8	**95.6**	92.6	**95.0**	**93.8**	**79.2**	**70.8**	**80.3**	**73.3**	**82.1**	**72.3**

Table 5 presents the results when L3 and L4 header information is incorporated, with IP addresses and port numbers masked. Among the baseline methods, YaTC [13] generally achieves strong performance across datasets, which can be attributed to its

ability to explicitly model packet boundaries and sequential structures. The proposed method achieves the best performance on most datasets, particularly on VPN 2016 [21], Tor 2016 [22], CSTNNET – TLS 1.3 [8], and Private-3, confirming its effectiveness in leveraging structural information beyond application-layer signals.

Table 5. Classification performance (%) across datasets using L3 and L4 protocol headers, where IP addresses and port numbers are masked. Bold indicates the best result per column.

Method	ISCX VPN 2016 [21]		ISCX Tor 2016 [22]		CSTNET TLS1.3 [8]		Private-3		Private-2		Private-1	
	AC	F1	AC	F1	AC	F1	AC	F1	AC	F1	AC	F1
ET-BERT [8]	76.1	64.8	85.1	80.1	81.3	78.9	71.5	74.4	61.8	61.3	63.1	59.3
YaTC [13]	90.5	79.3	97.6	90.3	89.7	88.5	89.1	85.0	**79.1**	**70.3**	85.5	**72.3**
TrafficFormer [9]	76.9	66.1	88.2	71.7	81.5	77.6	72.4	72.1	60.6	57.3	64.1	60.8
NetFound [19]	75.2	70.9	86.7	83.0	76.9	75.7	70.1	69.4	57.2	54.1	71.2	69.2
Proposed	**93.6**	**79.8**	**98.3**	**90.8**	**94.2**	**92.9**	**92.1**	**89.9**	78.3	70.2	**86.1**	71.3

However, when incorporating L3 and L4 header information, larger performance variations across datasets are observed not only for the proposed method but also for baseline models. This phenomenon can be attributed to the inherent sensitivity of lower-layer protocol features to environmental factors. Even when IP addresses and port numbers are masked, residual patterns such as protocol configurations, flow dynamics, and network-specific behaviors may differ significantly across data collection environments. As a result, models that rely on such features tend to exhibit increased performance variance when evaluated across datasets collected under different conditions.

In this context, the performance gap observed on the Private datasets reflects the domain-specific nature of L3/L4 features rather than a limitation of the proposed model. Notably, on the Private-2 dataset, YaTC [13] slightly outperforms the proposed method, suggesting that packet-level structural cues can be particularly effective in certain environments. In addition, the proposed method shows a noticeable performance gap between Private-1 and Private-2, further indicating that lower-layer features are sensitive to environmental variations. Nevertheless, despite this variability, the proposed method consistently achieves competitive or superior performance on most datasets, demonstrating its ability to effectively leverage structural information while maintaining strong generalization capability across diverse network conditions.

Table 6 compares the computational cost of the evaluated models in terms of memory usage and inference time. The proposed method requires moderate memory (3123 MB) and model inference time excluding preprocessing (34.8 ms/batch), remaining comparable to Transformer-based baselines while being significantly more efficient than NetFound [19]. These results indicate that the proposed model achieves a favorable trade-off between performance and computational efficiency.

Table 6. Computational cost comparison of evaluated models.

Model	Memory (MB)	Inference Time (ms/batch)
ET-BERT [8]	1526	37.6
YaTC [13]	1463	25.6
TrafficFormer [9]	2671	30.4
NetFound [19]	11,120	114.1
Proposed	3123	34.8

The experimental results reveal a clear distinction between application-layer (L7) information and packet-level header features. When only L7 information is used, the

proposed method consistently achieves stable and strong performance across all datasets, including the highly encrypted TLS 1.3 setting. This indicates that the model effectively captures structural patterns observable under encryption without relying on raw byte content. In contrast, incorporating L3/L4-based header features can improve performance on certain datasets but also increases sensitivity to environmental variations. This is because such features inherently reflect network-specific characteristics.

Overall, these results highlight a fundamental trade-off: while header-level information can provide additional discriminative power, it may reduce generalization under domain shifts. More importantly, the performance gains are most pronounced in highly encrypted settings (e.g., TLS 1.3), where byte-level representations become less informative due to high entropy. In such cases, the proposed permutation-invariant content representation and structure-aware modeling may help the model rely less on unstable byte patterns and instead place greater emphasis on more stable structural cues. In contrast, when lower-layer header information is incorporated, the performance gap becomes smaller and more dependent on dataset characteristics, as these features introduce environment-specific variability. We further observe that approaches preserving packet-level boundaries ([13], proposed) tend to exhibit relatively stable performance in certain settings. This observation suggests that packet boundary information itself may serve as a useful structural signal. The proposed method appears to balance this trade-off by leveraging hierarchical structural representations, achieving strong performance in encrypted environments while maintaining robustness across diverse network conditions. These findings suggest that structure-centric learning can be beneficial for improving both classification performance and generalization.

4.6. Pretraining Strategy Analysis

To analyze the contribution of hierarchical pretraining, we conduct ablation experiments using different combinations of structural learning tasks across three hierarchy levels: field (F), layer (L), and packet (P).

Table 7 summarizes the results. The impact of pretraining varies across datasets. For ISCX VPN 2016 and CSTNET-TLS 1.3, the performance differences between configurations are relatively small, and training without pretraining already achieves competitive results. This suggests that the proposed input representation and architecture are effective in capturing structural information even without pretraining. However, on Private-3, pretraining leads to a substantial improvement, with the full combination ($F + L + P$) significantly outperforming other settings. This indicates that pretraining becomes more beneficial when the data distribution is more complex or diverse. Across different task combinations, no single-level pretraining consistently improves performance. While field-level (F) and packet-level (P) tasks provide moderate gains in some cases, the layer level (L) alone is less effective. One possible reason is that the number of layer-level tokens is relatively limited compared to other levels (as shown in Figure 4), resulting in a weaker learning signal. In contrast, combining multiple levels generally yields more stable and competitive performance across datasets. This suggests that structural information from different hierarchy levels provides complementary cues.

Table 7. Effect of pretraining task combinations on classification accuracy (%). F/L/P denote field-, layer-, and packet-level shuffle tasks, respectively. Bold indicates the best result per column.

Pretraining (F/L/P)	ISCX VPN 2016 [21]	Private-3	CSTNET-TLS 1.3 [8]
None	89.5	52.0	94.4
F only	**90.5**	56.3	94.2
L only	88.1	49.9	92.7
P only	89.4	56.6	94.6
F + L	**90.5**	56.6	**95.0**
L + P	89.6	58.0	94.7
F + P	89.9	57.9	94.6
F + L + P (Proposed)	90.0	**79.2**	**95.0**

To further examine the role of pretraining, we evaluate the model under few-shot settings and compare it with ET-BERT [8]. Few-shot evaluation is particularly useful for analyzing pretraining effects, as differences in representation quality become more evident when labeled data is limited. As shown in Figure 5, the proposed method consistently outperforms ET-BERT across varying numbers of labeled samples, with larger performance gaps observed in low-data regimes. This indicates that pretraining contributes to improved data efficiency, enabling the model to achieve better performance with fewer labeled samples.

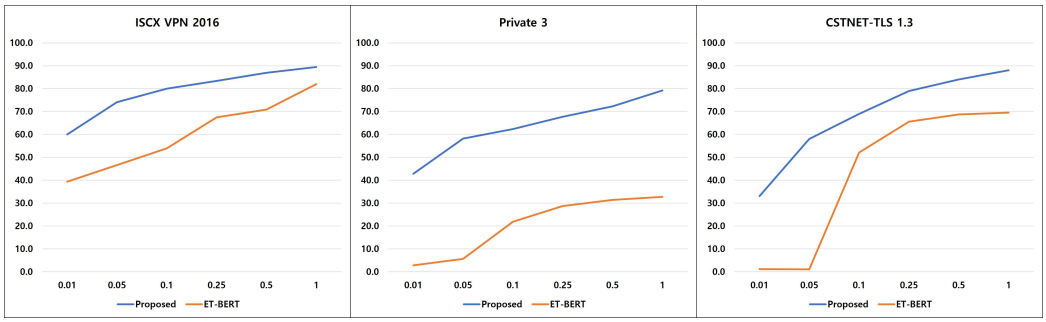

Figure 5. Few-shot classification performance across datasets. The proposed method consistently outperforms ET-BERT, with the largest gains in extremely low-data regimes (1% and 5% labeled data).

4.7. Hierarchical Pooling Analysis

To evaluate the contribution of different structural levels during classification, we compare several pooling configurations using layer-level (L), packet-level (P), and session-level (S) representations. Each representation is obtained by aggregating the corresponding tokens within its hierarchy level, and classification is performed using either a single-level representation, combinations of multiple levels, or all levels with the proposed inter-level gating.

Table 8 presents the results. When using a single-level representation, performance varies across datasets. For example, on TLS, packet-level representation (P) achieves higher accuracy (94.8%) and F1 score (93.5) compared to other levels, while on VPN, session-level representation (S) provides competitive accuracy (91.1%) but lower F1 (77.2). These results indicate that no single hierarchy level consistently dominates across different datasets. When combining multiple levels, the performance becomes more stable. However, simple mean pooling leads to a significant degradation (e.g., VPN accuracy drops from 90.0% to 86.7%, and TLS from 95.0% to 87.0%). This suggests that treating all representations equally is suboptimal, as it ignores the relative importance of each hierarchy level. In particular, multiple tokens exist within certain levels (e.g., packet- or layer-level representations within a session), and uniform averaging may dilute informative signals. The gating results further reveal dataset-dependent behavior. On TLS, the model assigns dominant weights to

packet-level representations, whereas on VPN, higher weights are assigned to session-level representations. This demonstrates that the model adaptively adjusts the contribution of each hierarchy level depending on the input characteristics.

Unlike conventional attention mechanisms that operate on token-level interactions, the proposed gating performs selection across pre-aggregated hierarchical representations. This allows the model to determine which structural level is more informative for each input. Finally, combining all hierarchy levels with the proposed gating achieves the overall good performance across datasets. This confirms that multi-level structural representations are complementary and that adaptive weighting across hierarchy levels is essential for effective traffic classification.

Table 8. Effect of hierarchical pooling strategies across datasets. "L", "P", and "S" denote layer-, packet-, and session-level representations, respectively. "Mean" refers to simple averaging of representations across all levels without hierarchical gating. Bold values indicate the best performance or dominant contributions in each column.

Config	ISCX VPN 2016 [21]		CSTNET–TLS1.3 [8]		ISCX VPN 2016 [21] Gate			CSTNET–TLS1.3 [8] Gate		
	AC	F1	AC	F1	L	P	S	L	P	S
L	**91.1**	**80.3**	92.7	91.3	1.00	0.00	0.00	1.00	0.00	0.00
P	90.2	79.9	94.8	93.5	0.00	1.00	0.00	0.00	1.00	0.00
S	91.1	77.2	91.9	89.7	0.00	0.00	1.00	0.00	0.00	1.00
L + P	90.2	78.0	**95.1**	92.9	0.28	0.71	0.00	0.01	0.99	0.00
L + S	90.0	75.2	94.9	**93.6**	0.59	0.00	0.41	0.99	0.00	0.01
P + S	90.9	78.8	94.9	93.5	0.00	0.85	0.15	0.00	0.99	0.01
Mean	86.7	77.8	87.0	85.4	-	-	-	-	-	-
All (Proposed)	90.0	79.8	95.0	**93.6**	0.24	0.22	**0.54**	0.01	**0.98**	0.01

4.8. Structure vs. Content Analysis

To investigate whether the proposed model relies primarily on structural rather than content information, we conduct perturbation experiments that independently modify content-related and structure-related input components.

Four content perturbations are evaluated: B0 (Original, no perturbation), B1 (No-Byte: byte embeddings removed entirely), B2 (Byte Shuffle: byte values within each field are randomly permuted), and B3 (Byte Random: all byte values are replaced with uniformly random values). B1 tests whether the model can classify using structure alone; B2 disrupts sequential byte patterns while preserving the value distribution; B3 removes both sequential patterns and distributional cues.

Two structural perturbations are also considered: S1 (No Length: field length embeddings are removed) and S2 (Hierarchy Shuffle: hierarchical protocol units—packets, layers, and fields—are randomly permuted at inference time).

Table 9 presents the results of content and structure perturbation experiments. For content perturbations, different trends are observed between [21] and [8]. On [21], all three perturbations (B1–B3) result in similar performance drops (approximately −11 F1), regardless of whether byte values are removed, shuffled, or randomized. This suggests that the model does not rely heavily on precise byte values, but rather on higher-level structural cues. On [8], however, the behavior differs. Byte shuffle (B2) leads to a relatively small performance decrease (ΔF1 = −1.0), while removing or randomizing byte values (B1, B3) results in more substantial degradation. This indicates that intra-field byte ordering is not a critical factor, while the presence and distribution of byte values still provide useful signals.

Table 9. Structure vs. content perturbation results (%). Δ indicates the drop from the B0 baseline. Bold values indicate the best performance

Type	Perturbation	ISCX VPN 2016 [21]				CSTNET–TLS 1.3 [8]			
		Acc	F1	ΔAcc	ΔF1	Acc	F1	ΔAcc	ΔF1
Content	B0 (Proposed)	**90.0**	**79.8**	—	—	**95.1**	**93.6**	—	—
	B1 (No-Byte)	88.7	68.3	−1.3	−11.5	87.5	85.1	−7.6	−8.5
	B2 (Byte Shuffle)	88.7	68.7	−1.3	−11.1	94.8	92.6	−0.3	−1.0
	B3 (Byte Random)	87.8	68.8	−2.2	−11.0	88.0	85.9	−7.1	−7.7
Structure	S1 (No Length)	83.0	56.6	−17.0	−23.2	93.3	92.6	−1.8	−1.0
	S2 (Shuffle)	87.4	67.1	−2.6	−12.7	93.1	92.2	−2.0	−1.4
	S3 (No Length + Shf.)	83.8	62.1	−6.2	−17.7	91.1	89.2	−4.0	−4.4
	S4 (No Content + Shf.)	86.2	63.1	−3.8	−16.7	84.4	81.1	−10.7	−12.5

Δ indicates performance drop relative to B0.

For structural perturbations, the impact is more pronounced on [21]. Removing length information (S1) leads to the largest drop (ΔF1 = −23.2), highlighting the importance of length-based structural cues. Hierarchy shuffling (S2) also degrades performance, suggesting that positional and relational information contributes to classification. In contrast, on [8], individual structural perturbations (S1, S2) result in relatively small performance changes. However, when both structural and content information are disrupted (S4), the performance drops significantly (ΔF1 = −12.5). This indicates that, under encrypted settings, the model relies on a combination of weak signals rather than any single dominant feature.

Overall, the results suggest that both structural and content-related signals contribute to the model's performance, but their roles differ across datasets. In non-encrypted settings, strong structural cues such as length play a dominant role. In contrast, under encrypted environments, where byte-level information becomes less reliable, the model relies more on structural signals that remain observable while still leveraging limited content-related cues when available.

4.9. Embedding Strategy and Attention Analysis

To further analyze how byte embedding strategies influence representation learning, we compare three content embedding methods—mean/max aggregation, linear projection, and CNN-based embedding—as well as a structure-only configuration with no byte embeddings.

Table 10 presents the effect of different content embedding strategies with and without structural embeddings. Across both datasets, the choice of content embedding has a significant impact on performance. In particular, the Mean/Max aggregation consistently achieves the best results, especially on [8], outperforming both linear and CNN-based approaches. This suggests that explicitly modeling byte-level sequential patterns is not always necessary, especially in encrypted environments where byte sequences exhibit high entropy. Instead, compact representations that capture aggregated characteristics of byte values are more effective.

Table 10. Effect of content embedding strategy on classification accuracy (%). ✓/× in the Struct. Emb. column indicate whether structural embeddings (position + length) are included. Bold values indicate the best performance in each column.

Struct. Emb.	Content Emb.	ISCX VPN 2016 [21]		CSTNET-TLS 1.3 [8]	
		AC	F1	AC	F1
✓	None (struct. only)	88.76	68.36	87.50	85.10
×	Mean/Max	83.04	56.61	94.76	92.62
	Linear	83.14	58.84	88.64	86.20
	CNN	86.78	65.74	85.97	84.86
✓	Mean/Max	88.87	69.22	**95.15**	**93.62**
	Linear	88.76	69.35	87.16	85.99
	CNN	**88.97**	**70.23**	86.67	85.78

This observation is further supported by the attention analysis in Figure 6. This interpretability is enabled by the proposed field-level tokenization, which allows the model to explicitly attend to semantically meaningful protocol fields. As a result, the attention distribution provides a direct and interpretable indication of which structural components contribute to the model's prediction. The model consistently assigns high attention weights to high-entropy fields such as app_data, data, and key_exchange fields across all embedding strategies.These fields correspond to large encrypted payloads, where fine-grained byte ordering is largely randomized, but distributional characteristics such as value statistics and payload size remain informative.

As a result, permutation-invariant aggregation (e.g., Mean/Max) is well aligned with the model's behavior, as it captures the dominant statistical patterns present in these fields. In contrast, methods that emphasize local ordering (e.g., CNN) or direct projection are less effective, as the underlying byte sequences lack stable sequential structure under encryption. At the same time, incorporating structural embeddings (e.g., position and length) consistently improves performance across configurations, although the gains are modest. This indicates that structural information provides complementary cues that are not captured by content embeddings alone.

Overall, the experimental results provide a consistent picture of how content and structural information contribute to traffic classification. Content-based signals alone are not always sufficient, particularly under encrypted settings where byte-level information becomes less reliable. At the same time, structural information provides additional context that complements content representations. Furthermore, the importance of different structural levels varies across datasets, making it difficult to rely on a fixed representation. This highlights the need for multi-level modeling and adaptive integration. Taken together, these results suggest that effective traffic classification is achieved not by relying solely on content or structure, but by combining both, where content embeddings provide compact summaries of byte distributions and structural information offers complementary contextual cues.

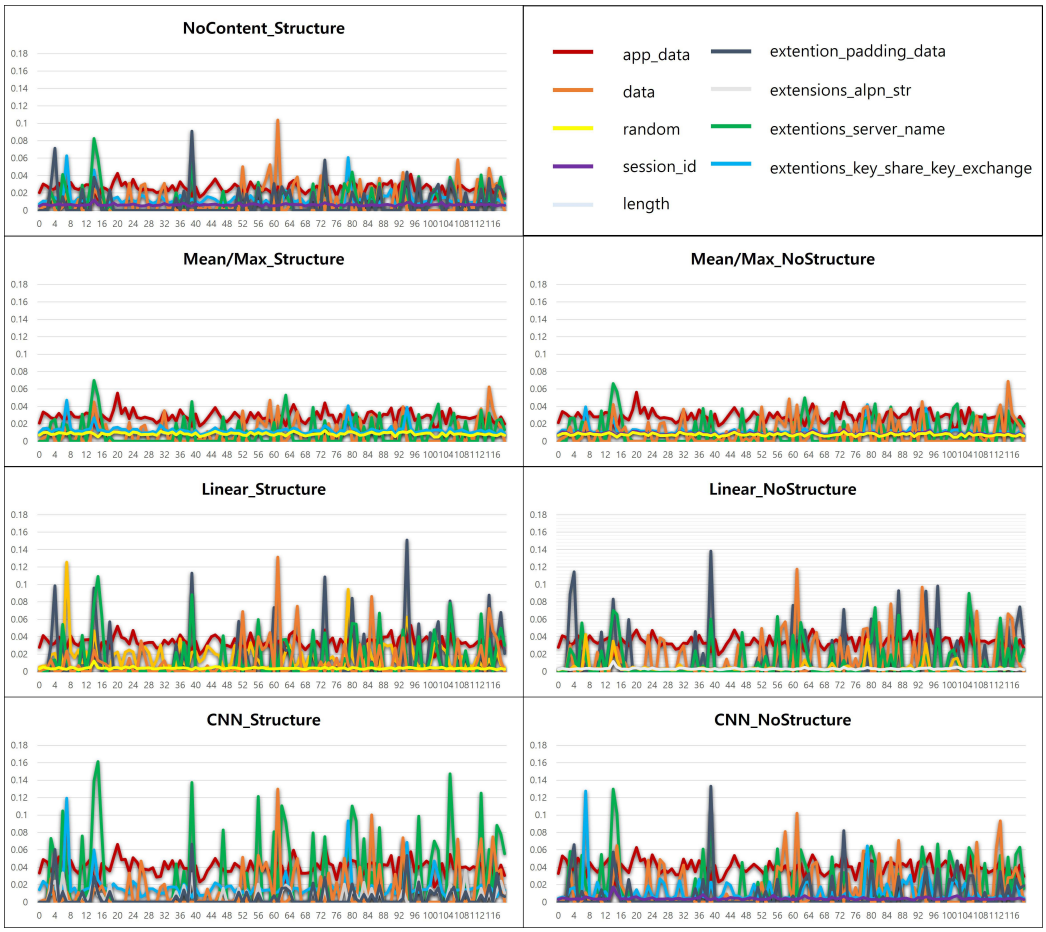

Figure 6. Attention weight distributions over representative TLS protocol fields under different byte embedding strategies (CSTNET–TLS 1.3). The model consistently emphasizes app_data regardless of the embedding configuration, suggesting that structural cues dominate over byte-level content.

5. Conclusions

This paper presented a structure-aware representation learning framework for encrypted network traffic classification. Motivated by the observation that encryption reduces the reliability of fine-grained byte-level patterns while preserving protocol-defined organization, the proposed approach shifts the focus from byte-centric modeling to the joint utilization of structural and content-derived signals.

The experimental results provide several key findings. First, perturbation analysis shows that while content-level signals alone are not sufficient, structural information alone is also not dominant. Instead, effective traffic classification requires combining compact content representations with structural context, particularly under strong encryption. Second, the hierarchical shuffle pretraining strategy improves representation quality by encouraging the model to capture ordering relationships across multiple levels. Although the performance gains are dataset-dependent, the proposed pretraining consistently enhances data efficiency, particularly in low-data regimes. Third, the hierarchical attention pooling with inter-level gating allows adaptive integration of multi-level representations. The analysis shows that different datasets rely on different structural levels, indicating that fixed aggregation strategies are insufficient and that adaptive selection is necessary.

Overall, the results suggest that effective traffic representation learning is achieved not by relying solely on content or structure but by combining both in a complementary manner. The proposed framework provides a unified approach that integrates structural constraints with compact content summaries, enabling robust performance across encrypted and heterogeneous environments.

https://doi.org/10.3390/electronics15091828

Despite its effectiveness, the proposed framework has several limitations that warrant further investigation. The method relies on protocol dissection to obtain hierarchical units, introducing a dependency on external parsing tools such as Wireshark. Variations in dissector implementations, heuristic rules, or protocol updates may affect the consistency of field-level tokenization, potentially impacting model robustness in deployment environments. Furthermore, the framework assumes that protocol structures are correctly formed and consistently observable. In practice, malformed packets, obfuscated traffic, or adversarial manipulations may violate such assumptions, limiting the effectiveness of structure-based modeling.

Future work will focus on reducing this dependency by exploring parser-agnostic approaches that can infer structural boundaries directly from raw traffic. In addition, improving robustness to irregular or adversarial traffic will be an important direction, for example, by incorporating noise-aware training strategies or learning more flexible structural representations. Finally, to enable practical deployment in real-world network environments, we plan to investigate model compression techniques such as knowledge distillation, as well as lightweight structural representations that preserve essential hierarchical information while reducing computational overhead.

Author Contributions: Conceptualization, M.-S.K.; Methodology, G.-M.Y.; Software, G.-M.Y.; Validation, Y.-S.J. and Y.-S.C.; Formal analysis, G.-M.Y.; Investigation, Y.-S.J.; Resources, S.-W.N. and J.-S.K.; Data curation, G.-M.Y.; Writing—original draft preparation, G.-M.Y.; Writing—review and editing, G.-M.Y. and M.-S.K.; Visualization, J.-M.K.; Supervision, M.-S.K.; Project administration, M.-S.K.; Funding acquisition, M.-S.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) funded by the Korea government (MSIT) (00235509, Development of security monitoring technology based network behavior against encrypted cyber threats in ICT convergence environment) and was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. RS-2025-02219319, Development of Standards for Quantum Cryptography based Zero Trust Secure Network/Service and Control/Management against Quantum Computer Attacks).

Data Availability Statement: The dataset and codes presented in this study are available upon request from the primary author and can be accessed at https://github.com/rudals2710/Modeling_Structure-Aware-and-Adaptive-Traffic-Classification-for-Encrypted-Networks.git, accessed on 24 April 2026.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Rescorla, E. The Transport Layer Security (TLS) Protocol Version 1.3. RFC 8446, August 2018. Available online: https://www.rfc-editor.org/rfc/rfc8446 (accessed on accessed on 24 April 2026).
2. Papadogiannaki, E.; Ioannidis, S. A survey on encrypted network traffic analysis applications, techniques, and countermeasures. *ACM Comput. Surv. (CSUR)* **2021**, *54*, 1–35.
3. Wickramasinghe, N.; Shaghaghi, A.; Tsudik, G.; Jha, S. SoK: Decoding the Enigma of Encrypted Network Traffic Classifiers. In Proceedings of the IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 12–14 May 2025; pp. 1825–1843.
4. Wang, W.; Zhu, M.; Wang, J.; Zeng, X.; Yang, Z. End-to-end encrypted traffic classification with one-dimensional convolution neural networks. In Proceedings of the IEEE International Conference on Intelligence and Security Informatics (ISI), Beijing, China, 22–24 July 2017.
5. Wang, Z.; Zeng, Q.; Liu, Y.; Li, P. Malware traffic classification using convolutional neural network for representation learning. In Proceedings of the International Conference on Information Networking (ICOIN), Da Nang, Vietnam, 11–13 January 2017; pp. 712–717.

6. Yang, Y.; Kang, K.; Jiang, L.; Gao, Z.; Guo, Y.; Zhang, J.; Deng, J. HAST-IDS: Learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection. *IEEE Access* **2017**, *5*, 26954–26964.
7. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is all you need. In Proceedings of the Advances in Neural Information Processing Systems 30 (NeurIPS), Long Beach, CA, USA, 4–9 December 2017; pp. 5998–6008.
8. Lin, X.; Xiong, G.; Gou, G.; Li, Z.; Shi, J.; Yu, J. ET-BERT: A contextualized datagram representation with pre-training transformers for encrypted traffic classification. In Proceedings of the ACM Web Conference 2022, Virtual, 25–29 April 2022; pp. 633–642.
9. Zhou, G.; Guo, X.; Liu, Z.; Li, T.; Li, Q.; Xu, K. TrafficFormer: An efficient pre-trained model for traffic data. In Proceedings of the IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 12–14 May 2025; pp. 1844–1860.
10. Peng, L.; Xie, X.; Huang, S.; Wang, Z.; Cui, Y. PTU: Pre-trained model for network traffic understanding. In Proceedings of the IEEE International Conference on Network Protocols (ICNP), Charleroi, Belgium, 28–31 October 2024; pp. 1–12.
11. Liu, L.; Li, R.; Li, Q.; Hou, M.; Jiang, Y.; Xu, M. FlowletFormer: Network behavioral semantic aware pre-training model for traffic classification. *arXiv* **2025**, arXiv:2508.19924.
12. He, H.Y.; Yang, Z.G.; Chen, X.N. PERT: Payload encoding representation from transformer for encrypted traffic classification. In *Proceedings of the 2020 ITU Kaleidoscope: Industry-Driven Digital Transformation (ITU K), Ha Noi, Vietnam, 7–11 December 2020*; IEEE: Piscataway, NJ, USA, 2020; pp. 1–8.
13. Zhao, R.; Zhan, M.; Deng, X.; Wang, Y.; Wang, Y.; Gui, G.; Xue, Z. Yet another traffic classifier: A masked autoencoder based traffic transformer with multi-level flow representation. In Proceedings of the AAAI Conference on Artificial Intelligence, Washington, DC, USA, 7–14 February 2023; Volume 37, pp. 5420–5428.
14. Hang, Z.; Lu, Y.; Wang, Y.; Xie, Y. 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, Hong Kong China, 16–18 October 2023; pp. 297–314.
15. Wang, T.; Xie, X.; Wang, W.; Wang, C.; Zhao, Y.; Cui, Y. NetMamba: Efficient network traffic classification via pre-training unidirectional mamba. In *Proceedings of the 2024 IEEE 32nd International Conference on Network Protocols (ICNP), Charleroi, Belgium, 28–31 October 2024*; IEEE: Piscataway, NJ, USA, 2024; pp. 1–10.
16. Wickramasinghe, N.; Shaghaghi, A.; Ferrari, E.; Jha, S. Less is More: Simplifying network traffic classification leveraging RFCs. In Proceedings of the Companion Proceedings of the ACM Web Conference, Sydney, NSW, Australia, 28 April–2 May 2025; pp. 1398–1401.
17. Zhao, D.; Jiang, B.; Liu, S.; Cui, S.; Shen, M.; Han, D.; Lu, Z. Language of network: A generative pre-trained model for encrypted traffic comprehension. *arXiv* **2025**, arXiv:2505.19482.
18. Lin, C.; Zhang, W.; Luo, H.; Meng, X.; Zhang, Y. Nethira: A heterogeneity-aware hierarchical pre-trained model for network traffic classification. *arXiv* **2026**, arXiv:2601.22494.
19. Guthula, S.; Beltiukov, R.; Battula, N.; Guo, W.; Gupta, A. NetFound: Foundation model for network security. *arXiv* **2023**, arXiv:2310.17025.
20. Park, J.T.; Choi, Y.S.; Cho, B.S.; Kim, S.H.; Kim, M.S. Multi-level pre-training for encrypted network traffic classification. *IEEE Access* **2025**, *13*, 68643–68659.
21. Draper-Gil, G.; Lashkari, A.H.; Mamun, M.S.I.; Ghorbani, A.A. Characterization of encrypted and VPN traffic using time-related features. In Proceedings of the 2nd International Conference on Information Systems Security and Privacy (ICISSP), Rome, Italy, 19–21 February 2016; pp. 407–414.
22. Lashkari, A.H.; Draper-Gil, G.; Mamun, M.S.I.; Ghorbani, A.A. Characterization of Tor traffic using time-based features. In Proceedings of the 3rd International Conference on Information Systems Security and Privacy (ICISSP), Porto, Portugal, 19–21 February 2017; pp. 253–262.
23. Ghasempour, A. *Support Vector Regression to Predict Power Consumption*; Technical Report; University of New Mexico: Albuquerque, NM, USA, 2024.
24. Tharwat, A. Classification assessment methods. *Appl. Comput. Inform.* **2021**, *17*, 168–192.