Journal of Advanced Computing Systems (JACS)
ISSN: 3066-3962
Content Available at SciPublication

Tokenized Flow-Statistics Encrypted Traffic Analysis: Comparative Evaluation of 1D-CNN, BiLSTM, and Transformer on ISCX VPN-nonVPN 2016 (A1+A2, 60 s)

Meng-Ju Kuo[1], *Boning Zhang*[2], *Haozhe Wang*[3]

[1] *Department of Electrical and Computer Engineering, CMU, PA, USA*
[2] *Computer Science, Georgetown University, DC, USA*
[3] *Operations Research and Information Engineering, Cornell, NY, USA*
mengju.kuo0688@outlook.com

DOI: 10.69987/JACS.2023.30804

Keywords

Encrypted traffic analysis, VPN detection, traffic classification, flow features, deep learning, 1D-CNN, BiLSTM, Transformer, robustness, ISCX VPN-nonVPN 2016

Abstract

End-to-end encryption is now the default for major Internet applications, reducing the effectiveness of payload-based deep packet inspection for security monitoring and traffic engineering. This paper evaluates payload-agnostic encrypted traffic analysis using only time-based bidirectional flow statistics derived from packet headers. We study the ISCX VPN-nonVPN 2016 dataset (60 s flow timeout) and conduct two supervised tasks: (i) Scenario A1 binary VPN detection and (ii) Scenario A2 14-class VPN-service identification across seven applications captured under VPN and non-VPN conditions. Because the released dataset provides engineered flow feature vectors rather than packet sequences, we introduce a structured tokenization that maps the 23 time-based features into a 6×4 token matrix capturing rate, inter-arrival, and burst/idle dynamics. On this representation we compare a 1D convolutional neural network (1D-CNN), a bidirectional LSTM (BiLSTM), and a Transformer encoder, and we report Accuracy, Macro-F1, ROC-AUC, and PR-AUC under a fixed 70/15/15 split (seed=42) with class-weighted cross-entropy and early stopping. On A1, the best model is an MLP baseline (Macro-F1=0.716), while the 1D-CNN achieves Macro-F1=0.689 with ROC-AUC=0.751. On the more challenging A2 task, the MLP reaches Macro-F1=0.389 and the 1D-CNN reaches Macro-F1=0.346. Feature-group masking ablations show that ACTIVE and FLOWIAT-related timing statistics contribute most to A2 performance, and a token-length study shows monotonic gains up to the full 6-token representation. Finally, cross-domain robustness tests (train on non-VPN flows, test on VPN flows, and vice versa) reveal large performance degradation (Macro-F1≈0.12–0.34), highlighting the need to evaluate encrypted traffic models under realistic tunneling shifts.

1. Introduction

Modern network visibility is increasingly constrained by encryption. TLS 1.3 encrypts more handshake information and accelerates setup compared to earlier versions [4], while DNS-over-HTTPS (DoH) encapsulates DNS messages inside HTTPS requests and therefore inherits HTTPS confidentiality and integrity [5]. QUIC goes further by integrating transport and security to provide encrypted, multiplexed streams over UDP [6] and by using TLS 1.3 as the cryptographic handshake and key schedule [7]. From a privacy perspective, this is a major win. From an operational perspective, it removes payload access and even reduces the amount of stable plaintext protocol metadata that used to be available to deep packet inspection (DPI) systems.

Network operators and defenders nevertheless need to answer practical questions: Is a flow going through a VPN? Which application category is responsible for congestion? Are policy-violating tunnels present? Are anomalous encrypted sessions indicative of malware or data exfiltration? In enterprise networks, these questions influence firewall policy, traffic shaping, incident response triage, and auditing. In access networks, they affect quality-of-service (QoS) and capacity planning. The core tension is that these tasks must increasingly be performed without seeing the payload.

Encrypted Traffic Analysis (ETA) studies how to infer application or behavior from payload-free side channels. Depending on the vantage point, the observable side channels include packet sizes, packet directions, timing, burst structure, transport-level flags, and limited handshake metadata. ETA can be framed as a supervised learning problem (traffic classification), an anomaly detection problem, or a clustering problem. Surveys of ML-based traffic classification emphasize that statistical patterns often remain discriminative even when payload is unavailable and that the problem spans networking and data mining [8]. More recent overviews document the rapid adoption of deep learning for encrypted traffic identification and discuss open issues such as dataset bias, non-stationarity, and lack of standardized evaluation [9], [20].

Deep learning models are especially compelling when raw packet sequences are available. Deep Packet is a representative approach that uses deep neural networks to integrate representation learning and classification for encrypted traffic [10], and subsequent work reports improvements by modeling contextual relationships using Transformers (ET-BERT) [12]. Such approaches benefit from long sequences, where convolution, recurrence, or attention can capture higher-order temporal dependencies beyond simple statistics. However, in many real-world settings and public datasets, raw packet traces are not available due to privacy and storage concerns, and researchers instead work with aggregated flow statistics.

Flow-feature datasets offer clear advantages: compactness, ease of sharing, and reduced privacy risk relative to payload or full packet traces. They also align with common operational tooling: flow exporters and analyzers can compute summary statistics in near real time. CICFlowMeter (formerly ISCXFlowMeter) is one example of a public tool that generates bidirectional flows and computes dozens of statistical features, including duration, packet/byte counts, and inter-arrival summaries in forward and backward directions [3]. The drawback is methodological: engineered feature vectors are unordered, whereas many modern deep models are designed for sequences. A naïve approach that treats a feature vector as a sequence imposes arbitrary ordering and makes model interpretation difficult.

This work studies payload-agnostic ETA using the ISCX VPN-nonVPN 2016 dataset [1], published by the Canadian Institute for Cybersecurity (CIC) and distributed via UNB [2]. The dataset contains seven application categories (Browsing, Chat, File Transfer, Mail, P2P, Streaming, and VoIP) captured under both VPN and non-VPN settings. It is widely used as a benchmark for VPN detection and encrypted service classification. We focus specifically on the 60 s flow-timeout feature sets for Scenario A1 (binary VPN detection) and Scenario A2 (VPN-service classification) because they provide a clean, reproducible setting for comparing models.

Our threat model is a passive observer who can collect packet headers and timestamps and export bidirectional flow statistics, but who cannot decrypt payloads. This matches many operational constraints: traffic may be end-to-end encrypted, and policy may prohibit payload inspection. We do not assume access to sensitive endpoint instrumentation or TLS session keys.

The central idea is to represent engineered flow features in a way that is compatible with sequence models while remaining faithful to their semantics. We propose a simple tokenization that maps the 23 time-based flow features into a 6×4 token matrix that preserves the structure of timing groups (rate, forward/backward inter-arrival, flow inter-arrival, and burst/idle dynamics). This tokenization is not meant to replace packet-level sequences; rather, it provides a principled bridge between compact flow tables and sequence architectures.

This paper makes three contributions. First, it provides fully reproducible experimental evaluations on the public ISCX VPN-nonVPN 2016 60 s feature sets for Scenario A1 (binary VPN detection) and Scenario A2 (14-class VPN-service classification). Second, it introduces a semantically grounded tokenization for time-based flow statistics and compares a 1D-CNN, a BiLSTM, and a Transformer encoder on top of it, alongside a strong MLP baseline. Third, it quantifies robustness under a realistic tunneling shift by training on non-VPN flows and testing on VPN flows (and vice versa) for a 7-class application task, and it reports masking ablations and token-length analyses to explain which timing groups matter most.

VPN detection is a particularly relevant ETA task because VPNs are widely used for legitimate privacy and remote access, but they are also used to bypass policy controls and to hide traffic destinations. From the viewpoint of a monitoring system that cannot decrypt payloads, VPN detection resembles a side-channel inference problem: the model must learn artifacts of tunneling such as encapsulation overhead, modified packetization, and changes in burst timing. Even when application data remains end-to-end encrypted, VPN tunneling can introduce additional encryption layers and intermediate endpoints, potentially altering the distributions of inter-arrival time and burstiness.

ETA methods are often categorized by the level of aggregation. Packet-level methods use sequences of packet sizes and directions; flow-level methods use aggregated statistics such as means and variances; and session-level methods may incorporate longer-term context such as destination IP reputation or SNI fields when available. This paper focuses on flow-level ETA because it is widely deployable: flow exporters are common, and feature extraction can be performed at high throughput. At the same time, flow-level models are known to be sensitive to exporter configuration (timeouts, sampling) and capture conditions, which motivates our explicit robustness evaluation.

Another important theme in the ETA literature is the tension between discriminative performance and generalization. Many published results are obtained on fixed train/test splits of benchmark datasets, but real networks evolve: protocols change, applications update, and congestion patterns fluctuate. Systematic reviews of encrypted malicious traffic detection emphasize that lack of standardized datasets and evaluation protocols makes it difficult to compare models fairly [20]. Similarly, empirical studies on encrypted malware traffic highlight that noisy labels and non-stationary distributions can dominate performance [13]. Our cross-domain experiments provide a concrete measurement of this issue in a VPN/non-VPN shift setting.

Finally, we note that privacy considerations matter in ETA. Because our pipeline uses only aggregated timing and rate statistics derived from packet headers, it is more privacy-preserving than payload inspection and avoids collecting application content. Nevertheless, ETA can still enable inference about user behavior. In deployments, this implies the need for clear governance, transparency, and appropriate access controls. Our contribution is methodological: we show how to conduct reproducible evaluations on public flow-feature datasets and how to report robustness and ablation analyses that can inform responsible use.

2. Materials and Methods

2.1 Dataset and tasks. We use the public ISCX VPN-nonVPN 2016 dataset [2] and the accompanying time-related feature sets introduced in [1]. We restrict experiments to the released ARFF files with a 60 s flow timeout, corresponding to Scenario A1 and Scenario A2. Scenario A1 labels each flow as VPN or Non-VPN. Scenario A2 provides 14 labels: for each of seven applications, a VPN and non-VPN variant (e.g., BROWSING and VPN-BROWSING). Table I summarizes the datasets and tasks. Both scenarios include 15,515 labeled flows and 23 numeric features.

2.2 Feature representation and assumptions. The released dataset consists of time-based bidirectional flow statistics computed from packet header observations rather than payload. This class of representation is common in ETA because it is payload-agnostic by construction: it can be extracted even when application content is encrypted. Examples of included statistics are flow duration, per-direction inter-arrival time summaries, and active/idle period summaries. These are consistent with how CICFlowMeter constructs a bidirectional (bi-flow) record by using the first packet to define forward and backward directions and then computing summary statistics over the flow [3]. Because the dataset is already distributed as aggregated flow features, our evaluation does not parse packets, reconstruct TCP streams, or decrypt any traffic.

2.3 Preprocessing. Flow-feature distributions typically exhibit heavy tails (e.g., byte rates) and can include zeros and occasional negative values depending on tool conventions for missing values. We apply a signed logarithmic transform to reduce heavy tails and stabilize optimization: $x' = \mathrm{sign}(x)\cdot\log(1+|x|)$. After this transform, we apply z-score standardization (zero mean, unit variance) using the training split statistics. This standardization is applied consistently to validation and test sets. The same preprocessing is used for both the flat vector baseline and the tokenized representation.

2.4 Data splits and reproducibility protocol. To ensure reproducibility and avoid optimistic leakage, we use a fixed stratified split of 70%/15%/15% for train/validation/test with random seed 42 for both A1 and A2. Stratification preserves the label distribution in each split (Table IV). All reported metrics are computed on the held-out test split after selecting the best model checkpoint by validation Macro-F1 (early stopping). We keep the protocol identical across models to enable fair comparison.

2.5 Tokenization of flow statistics. Sequence architectures require an ordered set of tokens. We design a tokenization that is grounded in network timing semantics rather than arbitrary feature ordering. Specifically, we map the 23 flow features into a 6×4 token matrix T with six tokens: RATE, FIAT, BIAT, FLOWIAT, ACTIVE, and IDLE (Table II). Each token contains four related statistics. RATE captures volume and throughput signals (duration, packets/s, bytes/s). FIAT and BIAT capture forward and backward inter-arrival summaries that can reflect interactive vs bulk transfer patterns and VPN encapsulation overhead. FLOWIAT captures overall flow inter-arrival statistics, while ACTIVE and IDLE capture burst/idle structure. For Scenario A1, the dataset provides std_fiat and std_biat, so FIAT and BIAT use {min, max, mean, std}. For Scenario A2, the dataset provides total_fiat and total_biat, so we use {min, max, mean, total}. This choice keeps the 6×4 shape constant across scenarios while using the available fields exactly as distributed.

2.6 Models. We compare four classifiers, summarized in Table V.

(1) MLP baseline (flat features): A two-hidden-layer multilayer perceptron with 256 hidden units per layer, ReLU activations, and dropout (p=0.3). It operates directly on the standardized 23-feature vector and serves as a strong baseline for engineered flow features.

(2) 1D-CNN (token features): A 1D convolutional network applied along the token dimension. The input is the 6×4 token matrix, interpreted as a length-6 sequence with 4 channels. We apply two Conv1D layers with kernel size 3 and output channels 32 and 64 respectively, followed by global max pooling and dropout. This architecture is lightweight and emphasizes local token interactions and feature selection.

(3) BiLSTM (token features): A 1-layer bidirectional LSTM [15] with hidden size 16, followed by mean pooling across the token sequence and a linear classifier. BiLSTMs can capture order-sensitive relationships, although in our setting the token order is fixed by semantics rather than time.

(4) Transformer encoder (token features): A 1-layer Transformer encoder [14] with d_model=16, two attention heads, a 32-unit feed-forward sublayer, and sinusoidal positional encoding. The output sequence is mean pooled and mapped to class logits. This model captures token-to-token interactions via self-attention.

2.7 Training configuration. We train with Adam [16] using learning rate 1×10−3 and weight decay 1×10−4. Loss is class-weighted cross-entropy to account for class imbalance, especially in A2 where some classes (e.g., VPN-STREAMING) have fewer samples (Table III). We use early stopping based on validation Macro-F1 with patience between 3 and 5 epochs, and we cap training to 12–15 epochs depending on the task to ensure consistent, reproducible CPU runtime. Table VI summarizes the configuration.

2.8 Metrics and diagnostics. For all tasks, we report Accuracy and Macro-F1. Macro-F1 is emphasized because it weights each class equally and is therefore sensitive to minority-class performance. For A1 (binary), we compute ROC-AUC and PR-AUC using the predicted probability of the VPN class. For A2 (multi-class), we compute macro-averaged one-vs-rest ROC-AUC and macro-averaged Average Precision (PR-AUC). In addition to scalar metrics, we report ROC/PR curves (Figures 2–3), a normalized confusion matrix for A2 (Figure 4), and ablation and robustness results.

2.9 Robustness protocol (VPN vs non-VPN domain shift). Randomly shuffled train/test splits can overestimate real-world performance because capture conditions and tunneling settings may change at deployment time. To simulate a common shift, we define a 7-class application task by stripping the "VPN-" prefix from A2 labels and then evaluate two cross-domain directions: train on non-VPN flows and test on VPN flows (Non-VPN→VPN), and train on VPN flows and test on non-VPN flows (VPN→Non-VPN). Training uses an 80/20 split of the training domain into train/validation with stratification; testing is performed on the entire target domain. This protocol measures generalization when the tunneling condition differs between training and deployment.

2.10 Implementation details. Experiments are implemented in Python (3.11) using PyTorch [18] for neural models and scikit-learn [17] for metrics and preprocessing. Figures are generated with Matplotlib [22]. We report parameter counts for transparency, and all results in the Results section are computed from the models trained under the described protocol.

2.11 Detailed feature inventory. The 23 released features fall into several families: (i) duration and throughput (duration, flowPktsPerSecond, flowBytesPerSecond), (ii) forward and backward inter-arrival time summaries (min_fiat, max_fiat, mean_fiat, std_fiat or total_fiat; and min_biat, max_biat, mean_biat, std_biat or total_biat), (iii) flow inter-arrival time summaries (min_flowiat, max_flowiat, mean_flowiat, std_flowiat), (iv) active period summaries (min_active, mean_active, max_active, std_active), (v) idle period summaries (min_idle, mean_idle, max_idle, std_idle). These features intentionally omit payload content and are therefore compatible with encrypted traffic monitoring.

2.12 Token order and invariances. The token sequence order is fixed as [RATE, FIAT, BIAT, FLOWIAT, ACTIVE, IDLE]. This order is not temporal; rather, it encodes a conceptual ordering from volume/rate to direction-specific timing to burst structure. Because the order is fixed and semantically defined, convolution and attention operations can be interpreted as modeling interactions between timing families. We emphasize that this differs from packet-sequence modeling, where the ordering is derived from time. Accordingly, our tokenization should be viewed as a structured feature grouping rather than a proxy for packet chronology.

2.13 Class-weighted loss. For a K-class problem with class counts n_k in the training split, we use weights w_k = N/(K·n_k), where N is the number of training samples. The cross-entropy loss is then L = −Σ_i w_{y_i} log p_{i,y_i}. This choice balances gradient contributions across classes and improves Macro-F1 stability when labels are imbalanced.

2.14 Metric definitions. Let P_k and R_k denote precision and recall for class k. The class-wise F1 is F1_k = 2 P_k R_k /(P_k+R_k). Macro-F1 is the unweighted average over classes: Macro-F1 = (1/K) Σ_k F1_k. ROC-AUC for A1 is computed from the rank ordering of the VPN class probability; PR-AUC is computed as average precision, which summarizes the precision–recall curve. For multi-class A2, ROC-AUC is computed as a macro-average of one-vs-rest AUCs and PR-AUC is computed as macro-average average precision.

2.15 Computational constraints and model sizing. The experiments are designed to be reproducible on CPU in a constrained environment. Therefore, the BiLSTM and Transformer are intentionally small (Table V). This sizing choice reduces wall-clock time and makes the experiments repeatable, but it may under-represent the potential of larger sequence models. Our primary goal is controlled comparison under a consistent protocol rather than maximizing benchmark scores.

2.16 Reproducibility artifacts. In addition to reporting metrics, we provide all generated tables and figures (Tables I–XIII and Figures 1–6) and specify software components. The code uses standard library implementations for preprocessing and metrics (scikit-learn) and for neural training (PyTorch). Because the dataset is publicly downloadable and the feature extraction is not re-run in our pipeline, the experiments can be reproduced by loading the same ARFF files and applying the described preprocessing, splits, and hyperparameters.

TABLE I. Datasets and tasks used in this work (A1+A2, 60 s).

Dataset	Source	Flow timeout	# flows	# features	Label space	Task
ISCX VPN-nonVPN 2016 (Scenario A1, 60s)	CIC/UNB	60 s	15515	23	VPN vs Non-VPN	Binary VPN detection
ISCX VPN-nonVPN 2016 (Scenario A2, 60s)	CIC/UNB	60 s	15515	23	14 classes (7 apps × {VPN, Non-VPN})	VPN-service classification

TABLE II. Tokenization mapping from released flow statistics to a 6×4 token matrix.

Token	Dim	Mapped flow features
RATE	4	duration, flowPktsPerSecond, flowBytesPerSecond, pad(0)
FIAT	4	min_fiat, max_fiat, mean_fiat, std_fiat (A1) / total_fiat (A2)
BIAT	4	min_biat, max_biat, mean_biat, std_biat (A1) / total_biat (A2)
FLOWIAT	4	min_flowiat, max_flowiat, mean_flowiat, std_flowiat
ACTIVE	4	min_active, mean_active, max_active, std_active
IDLE	4	min_idle, mean_idle, max_idle, std_idle

TABLE III. Scenario A2 class distribution (60 s).

Class	Count
BROWSING	2500
VPN-BROWSING	2500
VOIP	1829
VPN-VOIP	1620
FT	1372
P2P	1000
VPN-FT	898
MAIL	849
VPN-P2P	823
CHAT	778
VPN-CHAT	514
VPN-MAIL	383
STREAMING	252
VPN-STREAMING	197

TABLE IV. Train/validation/test split sizes for all experiments (seed=42).

Experiment	Split	# samples
A1 VPN detection	Train	10859
A1 VPN detection	Validation	2328
A1 VPN detection	Test	2328
A2 VPN-service	Train	10859
A2 VPN-service	Validation	2328
A2 VPN-service	Test	2328
Robustness NV→VPN (7 apps)	Train	6864
Robustness NV→VPN (7 apps)	Validation	1716
Robustness NV→VPN (7 apps)	Test	6935
Robustness VPN→NV (7 apps)	Train	5548
Robustness VPN→NV (7 apps)	Validation	1387

Robustness VPN→NV (7 apps)	Test	8580

TABLE V. Model architectures and parameter counts.

Model	Input	Architecture (summary)	Trainable parameters
MLP	Flat 23 features	2×FC (256 hidden), ReLU + Dropout	72450
1D-CNN	Tokenized 6×4	Conv1D(4→32,k=3) +Conv1D(32→64,k=3)+GlobalMaxPool	6754
BiLSTM	Tokenized 6×4	1-layer BiLSTM (hidden=16) + MeanPool	2882
Transformer	Tokenized 6×4	1-layer Encoder (d=16, heads=2, FFN=32) + MeanPool	2338

TABLE VI. Training configuration used across experiments.

Scope	Preprocessing	Optimizer	Loss	Stopping rule	Batch size
All experiments	sign(x)·log(1+|x|) + z-score	Adam (lr=1e-3, wd=1e-4)	Class-weighted CE	Early stopping (patience=3–5)	2048 (MLP) / 1024 (token)

3. Results

3.1 Main performance on Scenario A1 and A2. Table VII reports test performance for Scenario A1 (VPN vs Non-VPN). The MLP baseline achieves the highest Macro-F1 (0.716) and the highest ROC-AUC (0.797), indicating that the engineered time-based feature vector is highly informative for VPN detection in this dataset. The 1D-CNN trained on the tokenized representation achieves Macro-F1=0.689 with ROC-AUC=0.751, demonstrating that the tokenization retains most of the discriminative signal. The BiLSTM and Transformer achieve Macro-F1 values around 0.61. Because the token sequence length is only six, these sequence architectures have limited opportunity to exploit long-range structure beyond what is already encoded in the aggregated statistics.

Table VIII reports performance for Scenario A2 (14 classes). The task is harder because the model must distinguish both application category and VPN status across seven services. The MLP baseline achieves Macro-F1=0.389, while the 1D-CNN achieves Macro-F1=0.346. BiLSTM and Transformer perform worse in this configuration, which is consistent with the observation that their advantages become more pronounced on longer sequences (e.g., packet-length sequences) rather than compact feature-group tokens.

3.2 ROC/PR analysis for A1. Figure 2 plots ROC curves for A1. The MLP curve dominates others across most thresholds and achieves the highest ROC-AUC. Figure 3 shows precision–recall curves. PR curves are often more informative when the operational goal is to maintain high precision (low false-positive rate) while still capturing a large fraction of VPN flows. The MLP achieves PR-AUC=0.750 and maintains higher precision at moderate recall compared to the tokenized sequence models.

3.3 Confusion analysis for A2. Figure 4 shows the normalized confusion matrix for the best A2 model (MLP). Several patterns emerge. First, classes with larger support achieve higher recall because the classifier has more examples during training. Second, confusion frequently occurs between VPN and non-VPN variants of the same application, suggesting that timing signatures are dominated by application behavior rather than tunneling. Third, minority classes exhibit lower recall and are often confused with other high-throughput classes. Table IX provides per-class precision, recall, and F1 for the best A2 model.

3.4 Feature-group masking ablation. To understand which timing groups drive performance, we perform a post-hoc masking ablation on the trained 1D-CNN. For each ablation setting, we set one or more tokens in the input matrix to zero at test time and recompute metrics. Table X reports Macro-F1 for A1 and A2 under each setting. On A2, masking ACTIVE produces the largest drop in Macro-F1 (0.093), followed by FLOWIAT and FIAT. Single-group settings perform poorly, indicating that multiple timing groups must be combined to separate VPN-service labels.

3.5 Token length versus performance. Using the ablation-derived importance ranking, we keep the top-k tokens and mask the rest. Table XIII and Figure 6 show monotonic improvement as k increases from 1 to 6, reaching Macro-F1=0.346 only when all six tokens are retained.

3.6 Robustness under VPN/non-VPN domain shift. To quantify generalization when tunneling conditions change, we construct a 7-class application task by stripping the VPN prefix from A2 labels and evaluate cross-domain transfer. Table XI and Figure 5 summarize results. In the Non-VPN→VPN direction, the best model is the 1D-CNN with Macro-F1=0.340. In the VPN→Non-VPN direction, the best model is again the 1D-CNN with Macro-F1=0.271. Overall, cross-domain Macro-F1 is substantially lower than in-distribution evaluation, demonstrating that tunneling changes the distribution of flow statistics in a way that impairs supervised generalization.

3.7 Quantitative comparison across metrics. While Accuracy provides a simple summary, Macro-F1 is more informative for imbalanced settings such as A2. For example, the A2 MLP achieves Accuracy=0.465 and Macro-F1=0.389; the gap indicates that minority classes suffer more than majority classes. Ranking metrics provide complementary information: on A2, the MLP obtains ROC-AUC=0.878 and PR-AUC=0.423, indicating that probability estimates still contain useful signal even when the top-1 label is imperfect. Table XII reports token importance as Macro-F1 drop when removing each token, providing effect sizes that connect model behavior to timing semantics.

3.8 Convergence and parameter efficiency. All models converged within a small number of epochs under early stopping. On A1, the MLP trained for 15 epochs, the 1D-CNN for 13 epochs, the BiLSTM for 9 epochs, and the Transformer for 8 epochs before early stopping selected the best checkpoint. On A2, the MLP and 1D-CNN trained for 15 epochs, while the BiLSTM and Transformer trained for 8 and 9 epochs respectively. This pattern is consistent with validation Macro-F1 saturating quickly when inputs are engineered flow summaries rather than long raw sequences.

Table V also shows a large spread in parameter counts. The MLP contains 72,450 trainable parameters, while the tokenized 1D-CNN contains 6,754 parameters and the Transformer contains 2,338 parameters. Despite being much smaller, the 1D-CNN achieves competitive performance relative to the MLP on A1 and retains a substantial fraction of the A2 signal. These results suggest that, for compact tokenized flow features, model capacity beyond a certain point may provide diminishing returns, and that inductive bias (how the model composes feature groups) can matter as much as parameter count.

3.9 Practical takeaway from ablations. The token importance estimates (Table XII) are useful for feature-set design. If an operator must compute a reduced feature set for high-speed monitoring, ACTIVE/IDLE and FLOWIAT statistics should be prioritized because their removal causes the largest Macro-F1 drops on A2. Conversely, RATE alone is insufficient for identifying VPN-service labels, even though it captures throughput. These findings match intuitive expectations: throughput is informative for coarse categories, but burst/idle and inter-arrival structure are more informative for separating application behaviors under encryption.

TABLE VII. Scenario A1 (60 s) VPN detection performance on the test split.

Model	Accuracy	Macro-F1	ROC-AUC	PR-AUC
MLP	0.718	0.716	0.797	0.75
1D-CNN	0.694	0.689	0.751	0.701
BiLSTM	0.618	0.613	0.599	0.554
Transformer	0.617	0.609	0.592	0.56

TABLE VIII. Scenario A2 (60 s) VPN-service classification performance on the test split.

Model	Accuracy	Macro-F1	ROC-AUC	PR-AUC
MLP	0.465	0.389	0.878	0.423
1D-CNN	0.437	0.346	0.86	0.385
BiLSTM	0.231	0.121	0.669	0.153
Transformer	0.265	0.188	0.702	0.192

TABLE IX. Per-class precision/recall/F1 for Scenario A2 (best model: MLP).

Class	Precision	Recall	F1	Support
BROWSING	0.534	0.651	0.587	375.0
CHAT	0.18	0.308	0.227	117.0
FT	0.864	0.495	0.63	206.0
MAIL	0.306	0.26	0.281	127.0
P2P	0.507	0.513	0.51	150.0
STREAMING	0.102	0.132	0.115	38.0
VOIP	0.616	0.803	0.697	274.0
VPN-BROWSING	0.629	0.293	0.4	375.0
VPN-CHAT	0.361	0.286	0.319	77.0
VPN-FT	0.226	0.052	0.084	135.0
VPN-MAIL	0.198	0.702	0.309	57.0
VPN-P2P	0.5	0.54	0.519	124.0
VPN-STREAMING	0.196	0.6	0.295	30.0
VPN-VOIP	0.531	0.42	0.469	243.0

TABLE X. Token masking ablation on the 1D-CNN (Macro-F1).

Ablation setting	A1 Macro-F1	A2 Macro-F1
All tokens	0.689	0.346
w/o RATE	0.633	0.296
w/o FIAT	0.59	0.269
w/o BIAT	0.589	0.31
w/o FLOWIAT	0.627	0.267

w/o ACTIVE	0.631	0.252
w/o IDLE	0.656	0.295
Only IAT (FIAT+BIAT+FLOWIAT)	0.602	0.205
Only Burst (ACTIVE+IDLE)	0.356	0.065
Only RATE	0.394	0.031

TABLE XI. Cross-domain robustness (7 applications): train on one domain and test on the other.

Train→Test domain	Model	Accuracy	Macro-F1
Non-VPN→VPN	MLP	0.378	0.332
Non-VPN→VPN	1D-CNN	0.372	0.34
Non-VPN→VPN	BiLSTM	0.329	0.188
Non-VPN→VPN	Transformer	0.26	0.192
VPN→Non-VPN	MLP	0.254	0.217
VPN→Non-VPN	1D-CNN	0.314	0.271
VPN→Non-VPN	BiLSTM	0.408	0.199
VPN→Non-VPN	Transformer	0.154	0.124

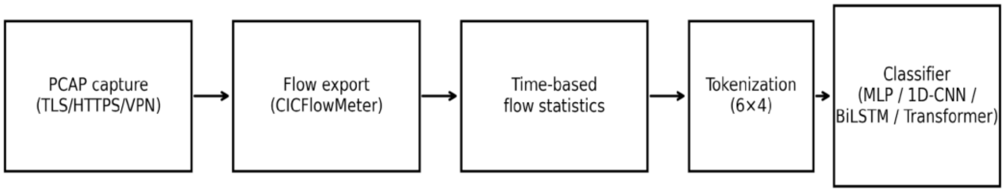
TABLE XII. Token importance estimated by Macro-F1 drop on A2 when removed (1D-CNN).

Token	Macro-F1 drop on A2 when removed	Importance rank
ACTIVE	0.093	1.0
FLOWIAT	0.078	2.0
FIAT	0.077	3.0
IDLE	0.051	4.0
RATE	0.049	5.0
BIAT	0.036	6.0

TABLE XIII. Token length vs performance for A2 (1D-CNN).

k tokens kept	Macro-F1 (A2, CNN)	Accuracy (A2, CNN)
1.0	0.026	0.048
2.0	0.137	0.225
3.0	0.22	0.308

4.0	0.26	0.341
5.0	0.31	0.381
6.0	0.346	0.437



Tasks: A1 VPN detection (binary) and A2 VPN-service classification (14 classes)

Fig. 1. Payload-agnostic ETA pipeline used in this study.

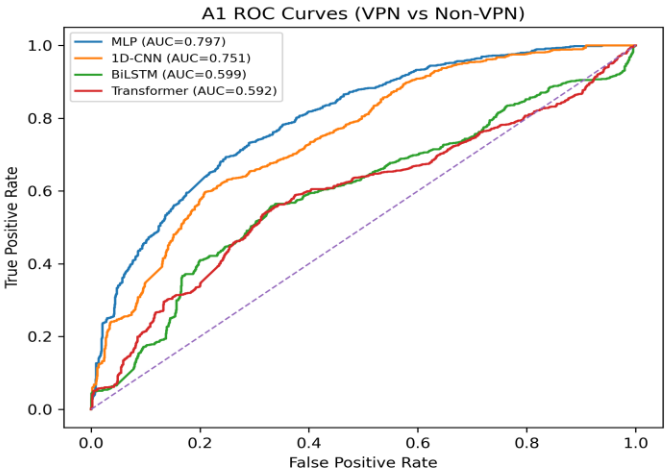


Fig. 2. ROC curves for A1 VPN detection.

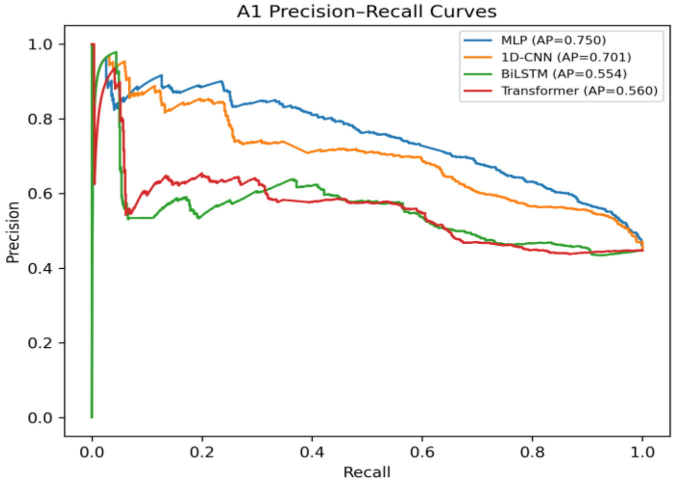


Fig. 3. Precision–Recall curves for A1 VPN detection.

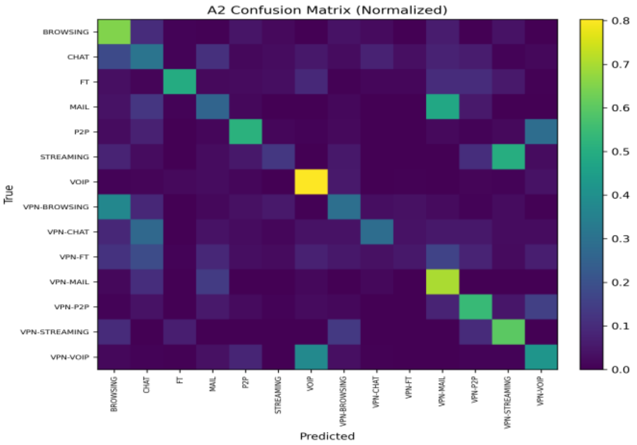


Fig. 4. Normalized confusion matrix for A2 (best model: MLP).

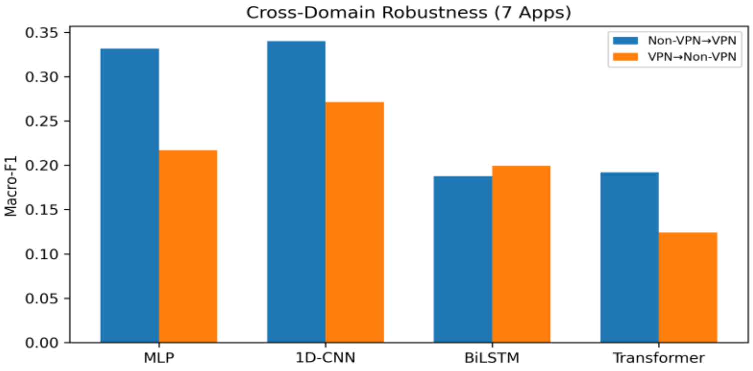


Fig. 5. Cross-domain robustness for application classification under VPN/non-VPN shift (Macro-F1).

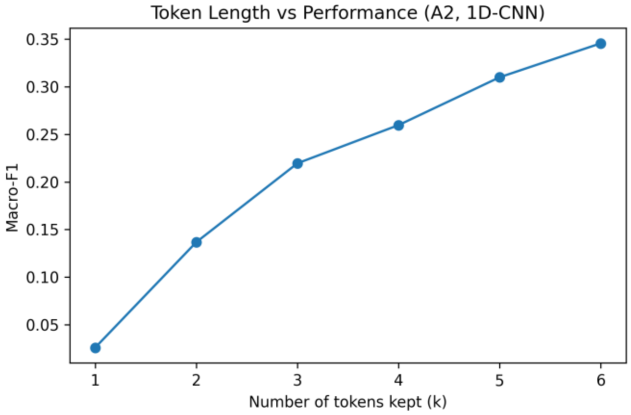


Fig. 6. Token length vs A2 Macro-F1 for the 1D-CNN.

4. Discussion

These experiments lead to several observations relevant to payload-agnostic ETA.

First, strong baselines matter. The MLP baseline dominates in-distribution performance on both A1 and A2. This indicates that on ISCX VPN-nonVPN 2016, time-based flow statistics are already engineered in a way that aligns closely with the classification tasks described by Draper-Gil et al. [1]. For researchers and practitioners, this is a reminder that deep architectures should be compared against well-tuned shallow models; otherwise, improvements may be overstated.

Second, tokenization enables controlled sequence modeling on flow features, but sequence length limits the benefit of recurrence and attention. Unlike packet-level approaches that operate on dozens to hundreds of packets per flow (e.g., Deep Packet [10]) or that pretrain contextual datagram representations (ET-BERT [12]), our input contains only six tokens derived from aggregated statistics. With such short sequences, the representational power of BiLSTM and Transformer is constrained, and these models can be harder to optimize than a simple MLP. The 1D-CNN performs competitively because convolution plus global pooling provides a strong inductive bias for selecting informative token groups and interactions.

Third, robustness evaluation is essential for ETA claims. Anderson and McGrew show that encrypted traffic classification is sensitive to non-stationarity and label noise in operational settings [13]. Our cross-domain results demonstrate a related phenomenon: VPN tunneling shifts the distribution of flow timing features enough to significantly reduce Macro-F1 even when the underlying application label set is unchanged. This is important because many ETA studies report only random-split evaluation, which can overestimate real-world performance when deployment conditions differ.

Fourth, ablation analysis provides an interpretable bridge between model outputs and network semantics. ACTIVE and FLOWIAT statistics are the strongest contributors to A2 performance in our masking study. Intuitively, VPN tunnels introduce additional encapsulation, scheduling, and potential buffering that can alter burstiness and inter-arrival variability. However, we also observe redundancy: multiple tokens contribute overlapping information, and performance does not collapse when any single token is removed. This suggests that robust models should leverage multiple timing signals rather than relying on a single feature family.

Fifth, limitations must be stated precisely. This paper evaluates only engineered flow features and therefore cannot capture packet-level leakage such as packet size sequences, directional burst patterns, or TLS/QUIC handshake fields. Furthermore, A2 exhibits class imbalance (Table III), which affects Macro-F1 and leads to minority-class confusions (Figure 4). Our model configurations are intentionally small to support CPU reproducibility; larger models or self-supervised pretraining may improve results but would require larger datasets or additional unlabeled traffic.

Threats to validity. Internal validity is supported by a fixed split protocol and consistent preprocessing. External validity is limited by the dataset capture environment: ISCX VPN-nonVPN 2016 reflects a particular set of applications and VPN settings, and real enterprise networks may differ in latency, congestion, and multiplexing behavior. Construct validity is limited by the flow-feature representation: different flow exporters or timeout settings can change feature distributions. For operational deployment, models should be validated across multiple capture days, networks, and VPN implementations.

Practical recommendations. Based on our results and prior surveys [8], [9], we recommend that ETA studies using flow features should: (i) report strong tabular baselines; (ii) include robustness tests that simulate realistic deployment shifts (VPN vs non-VPN, different capture conditions); (iii) provide feature-group ablations to support interpretability; and (iv) clearly separate what is inferred from payload-free metadata versus what would require decryption or deep protocol parsing.

Adversarial considerations. ETA systems can be subject to adaptive behavior. Attackers may attempt to pad packet sizes, add timing jitter, or multiplex flows to mimic benign patterns. At the flow-feature level, such actions may alter throughput and inter-arrival statistics. While our experiments do not include active adversaries, the robustness results suggest that even naturally occurring shifts can degrade performance; deliberate obfuscation may further reduce reliability. Future work should evaluate robustness to controlled perturbations of timing and burst structure and should consider detection frameworks that provide calibrated uncertainty estimates.

Relation to protocol evolution. The Internet's move toward TLS 1.3, DoH, and QUIC reduces available plaintext metadata [4]–[7]. Flow-statistic methods remain applicable under these protocols because they rely on timing and volume rather than decrypted content. However, protocol evolution can change traffic patterns (e.g., 0-RTT, connection migration in QUIC), which may shift distributions. This reinforces the need for continuous evaluation and model updating when ETA is used operationally.

Ethical and privacy implications. Although our pipeline uses payload-free features, ETA can still enable inference about user activities. Organizations should ensure that ETA deployments comply with applicable laws and policies and should consider transparency measures. From a research perspective, releasing aggregated flow-feature datasets (as in ISCX VPN-nonVPN 2016) provides a useful compromise between reproducibility and privacy. Nevertheless, researchers should be cautious when claiming general-purpose surveillance capabilities from benchmark performance.

Future directions. Three directions are especially promising. First, combining flow statistics with limited, non-sensitive handshake metadata (when available) may improve robustness. Second, self-supervised pretraining on large unlabeled traffic corpora, as explored by ET-BERT [12], could improve generalization on small labeled datasets. Third, evaluation protocols should broaden beyond random splits to include cross-day, cross-network, and cross-protocol settings, enabling more realistic estimates of deployment performance.

Comparison with prior benchmarks. The ISCX VPN-nonVPN 2016 dataset has been used in numerous studies with different feature designs and inputs. The original dataset paper focuses on time-related features and shows that they can support both VPN detection and service characterization [1]. Later deep learning work typically operates on packet sequences or byte streams, sometimes achieving very high F1 on related tasks when enough training data and strong inputs are available [10], [12]. Our evaluation differs in two ways: (i) we restrict ourselves to the released 60 s ARFF features (compact flow statistics) and (ii) we emphasize robustness under VPN/non-VPN domain shift. As a result, our absolute scores are not directly comparable to packet-sequence models; instead, the results should be interpreted as a careful baseline for feature-table ETA.

Why scores differ across representations. Packet-sequence models have access to fine-grained burst structure—e.g., patterns of alternating small and large packets—that can be highly discriminative even under encryption. Aggregated flow statistics compress these patterns into a small number of summaries, potentially discarding information. Therefore, it is expected that models trained on feature tables may underperform models trained on rich sequences, particularly for multi-class problems such as A2. Nevertheless, feature tables remain attractive in operational environments because they are cheap to compute and store.

Data quality and exporter effects. Flow-feature extraction can be sensitive to implementation details, such as how active and idle periods are defined and how inter-arrival times are computed under retransmissions or out-of-order delivery. CICFlowMeter provides a commonly used implementation for CIC datasets [3], but different exporters may yield slightly different statistics. This is one reason we treat robustness evaluation as a first-class requirement: even if a model performs well on one feature extractor, it may degrade when deployed with a different exporter configuration.

Interpretation of robustness gaps. The cross-domain results demonstrate that VPN tunneling can change the distribution of timing features enough to degrade application recognition. This has two practical implications. First, if the deployment network includes both VPN and non-VPN traffic, a single global model may not be sufficient; conditional models or domain adaptation may be required. Second, reporting only in-distribution performance can lead to overly optimistic conclusions about real-world effectiveness. These implications are consistent with broader calls in the ETA literature for careful, deployment-aligned evaluation protocols [9], [13], [20].

Additional observations on metric choice. Macro-F1 is intentionally strict: a model that performs well on large classes but poorly on small classes will have a low Macro-F1 even if Accuracy is moderate. This is desirable for security and policy settings where missing a minority behavior can be costly. At the same time, in some deployment contexts operators may prefer thresholded detectors or top-N recommendations rather than hard labels. In those cases, ranking metrics such as ROC-AUC and PR-AUC provide complementary information. For example, on A2 the MLP's ROC-AUC (0.878) is substantially higher than its Macro-F1 (0.389), implying that the model's probability estimates contain useful signal beyond the top-1 decision. A practical system could exploit this by flagging uncertain flows for secondary analysis.

Handling missing or tool-specific values. Flow exporters sometimes emit sentinel values when a statistic is undefined (e.g., when no backward packets are present). In the released ARFF files we treat missing numeric fields as −1 before applying the signed log transform and standardization. This approach preserves the information that a statistic is undefined while keeping the preprocessing pipeline deterministic. In deployments, it may be preferable to add explicit missingness indicators or to impute such fields based on exporter semantics.

Model selection perspective. The results suggest a simple decision rule for similar flow-feature datasets: begin with a strong tabular baseline, then introduce structured representations only if they improve robustness or interpretability. In our experiments, the 1D-CNN is the most competitive of the sequence models and is also relatively lightweight (6,754 parameters). This makes it a reasonable candidate when a compact model is required or when token-level ablations are desired for explainability.

5. Conclusions

This paper presented a payload-agnostic encrypted traffic analysis pipeline based on time-based bidirectional flow statistics and evaluated four models on the ISCX VPN-nonVPN 2016 dataset (Scenario A1 and A2, 60 s). A semantically grounded 6×4 tokenization enabled sequence models (1D-CNN, BiLSTM, Transformer) to operate on engineered flow features. Across both tasks, the MLP baseline was strongest, while the 1D-CNN provided competitive performance on the tokenized representation. Masking ablations and token-length analysis showed that multiple timing groups are necessary for A2 performance. Cross-domain evaluation under VPN/non-VPN shifts produced substantial degradation, emphasizing the importance of robustness tests for ETA claims. Future work will extend this pipeline to packet-level sequences and TLS/QUIC metadata to better reflect modern encrypted protocols.

References

[1] 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 Proc. Int. Conf. Inf. Syst. Secur. Privacy (ICISSP), 2016, pp. 407–414.

[2] Canadian Institute for Cybersecurity, “VPN-nonVPN dataset (ISCXVPN2016),” Univ. New Brunswick, accessed Jan. 2026. [Online]. Available: https://www.unb.ca/cic/datasets/vpn.html

[3] Canadian Institute for Cybersecurity, “CICFlowMeter (formerly ISCXFlowMeter),” accessed Jan. 2026. [Online]. Available: https://www.unb.ca/cic/research/applications.html

[4] E. Rescorla, “The Transport Layer Security (TLS) Protocol Version 1.3,” RFC 8446, Aug. 2018.

[5] P. Hoffman and P. McManus, “DNS Queries over HTTPS (DoH),” RFC 8484, Oct. 2018.

[6] J. Iyengar and M. Thomson, “QUIC: A UDP-Based Multiplexed and Secure Transport,” RFC 9000, May 2021.

[7] M. Thomson and S. Turner, “Using TLS to Secure QUIC,” RFC 9001, May 2021.

[8] T. T. T. Nguyen and G. Armitage, “A Survey of Techniques for Internet Traffic Classification Using Machine Learning,” IEEE Commun. Surveys Tuts., vol. 10, no. 4, pp. 56–76, 2008.

[9] S. Rezaei and X. Liu, “Deep Learning for Encrypted Traffic Classification: An Overview,” IEEE Commun. Mag., vol. 57, no. 5, pp. 76–81, 2019.

[10] M. Lotfollahi, R. Shirali Hossein Zade, M. J. Siavoshani, and M. Saberian, “Deep Packet: A Novel Approach for Encrypted Traffic Classification Using Deep Learning,” arXiv:1709.02656, 2017.

[11] M. Lotfollahi, R. S. H. Zade, M. J. Siavoshani, and M. Saberian, “Deep Packet: A Novel Approach for Encrypted Traffic Classification Using Deep Learning,” Soft Comput., vol. 24, pp. 1999–2012, 2020.

[12] 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 Proc. ACM Web Conf. (WWW), 2022, pp. 633–642.

[13] B. Anderson and D. McGrew, “Machine Learning for Encrypted Malware Traffic Classification: Accounting for Noisy Labels and Non-Stationarity,” in Proc. ACM SIGKDD, 2017, pp. 1723–1732.

[14] A. Vaswani et al., “Attention Is All You Need,” in Proc. NeurIPS, 2017, pp. 5998–6008.

[15] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” Neural Comput., vol. 9, no. 8, pp. 1735–1780, 1997.

[16] D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” arXiv:1412.6980, 2014.

[17] F. Pedregosa et al., “Scikit-learn: Machine Learning in Python,” J. Mach. Learn. Res., vol. 12, pp. 2825–2830, 2011.

[18] A. Paszke et al., “PyTorch: An Imperative Style, High-Performance Deep Learning Library,” in Proc. NeurIPS, 2019, pp. 8024–8035.

[19] L. Buitinck et al., “API Design for Machine Learning Software: Experiences from the scikit-learn Project,” arXiv:1309.0238, 2013.

[20] Z. Wang, K.-W. Fok, and V. L. L. Thing, “Machine Learning for Encrypted Malicious Traffic Detection: Approaches, Datasets and Comparative Study,” arXiv:2203.09332, 2022.

[21] A. Habibi Lashkari, “CICFlowMeter-V4.0 (formerly ISCXFlowMeter),” GitHub repository, accessed Jan. 2026. [Online]. Available: https://github.com/ahlashkari/CICFlowMeter

[22] J. D. Hunter, “Matplotlib: A 2D Graphics Environment,” Comput. Sci. Eng., vol. 9, no. 3, pp. 90–95, 2007.