Building Trust in Machine Learning-Powered Networking: The Network Explainer Framework

Riya Ponraj * Ram Durairajan* Yu Wang*

Abstract.

Recent advancements in deep learning (DL) have automated tremendous time-series applications in networking systems. However, the lack of explainability in these advancements has made network operators reluctant to deploy DL-based solutions in production networks, as they cannot fully understand or verify the reasoning behind critical decisions taken by those solutions. To address this problem, we introduce the Traffic-Explainer, an explainability framework to uncover influential features driving DL predictions over time-series networking applications. By maximizing the mutual information between predictions and explanations, Traffic-Explainer provides deeper insights into DL decision-making, fostering greater trust in DL-driven network management. We demonstrate the efficacy of Traffic-Explainer through two critical applications: traffic classification and network cartography. In traffic sequence classification, the Traffic-Explainer identifies the most impactful bytes and byte-byte interactions that shape ML-based predictions for application classification and country localization. In network cartography, it pinpoints the most critical physical fiber cables responsible for carrying critical Internet traffic. Through extensive real-world evaluations, we validate Traffic-Explainer's ability to generate high-fidelity explanations, strengthening the trustworthiness of DL-driven networking solutions. Our implementation is publicly available at https://anonymous.4open.science/r/Byte-Explainer-F7C0/README.md.

1 Introduction

Networking systems are the backbone of modern digital infrastructures [41, 18] and recent advancements in deep learning (DL) have automated tremendous applications in networking systems, and many of them could be formulated as time-series tasks, such as traffic classification, anomaly detection, routing, and resource allocation. However, despite their efficiency, decisions made by DL models are based on learned features and lack interpretability. Taking traffic classification as an example, earlier traffic classifiers rely on manually extracted features (e.g., device and certificate) [35, 36] given by domain expertise, making their decision-making inherently explainable. In contrast, DL-based classifiers, such as ET-Bert and TFE-GNN [41, 18], operate within a latent embedding space, where classified labels have no direct correspondence with input features from domain perspectives. This lack of explainability has made network operators reluctant to deploy DL-based solutions in production networks [15, 8], as shown in Figure 1.1.

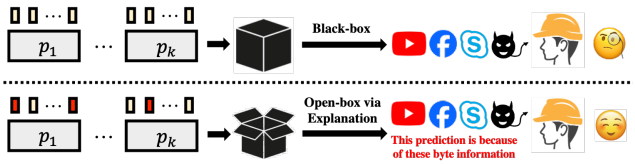


Figure 1.1: (a) **Untrustworthy scenario:** DL-based solutions that solely present predictions with no explanation. (b) **Trustworthy scenario:** DL solutions that offer explanations gain the trust of network operators.

To improve the explainability of DL-based networking systems and facilitate trustworthy network management, conventional explanation methods can be naturally employed [2, 37, 43]. Gradient-based methods like Saliency Maps [33, 30, 39] can be used to unveil the most influential parts of networking systems that contribute to the DL-models decision-making. Furthermore, the identified explanation can facilitate knowledge discovery, such as identifying novel traffic anomalies or emerging application categories in traffic sequence classification. Despite the well-established explanation methods [27, 20, 40, 3, 42], there is a notable absence of explanation studies on the networking domain. For example, in traffic classification, Although rule-based traffic classification is self-explainable [15], their explanations are hard-coded into their handcrafted features, making them difficult to adapt to other traffic applications. NetXplain [23] and Hybrid Explainability [8] are the only notable efforts bridging explainability and networking. However, NetXplain focuses exclusively on explaining traffic delays, while Hybrid Explainability is limited to post-hoc interpretations of non-DL models such as decision trees. No frameworks have systematically explained deep learning behaviors in different time-series networking applications.

*University of Oregon, {ripo, ramd, yuwang}@uoregon.edu

To develop explainable DL-based networking applications in building the trust of network operators, this paper designs an explanation framework for deep learning-based networking applications. Achieving this goal requires addressing three key challenges. First, we need to determine the explanation object that balances specificity (i.e., encoding sufficient class-desired information to enable the explanation) and generalizability (i.e., being applicable across various networking tasks). Since many networking problems can be formulated as sequential predictions, we choose each sequence as the basic explanation object and aim to identify the most important units within the sequence for its prediction. For instance, in traffic classification, where sequences consist of bytes, explanations should highlight the critical bytes/byte-byte interactions. In network cartography, where sequences comprise round-trip times (RTTs) across hops, explanations should identify the specific RTT hop that characterizes the physical fiber links. Second, the DL-based traffic classifier should handle sequential data and deliver high accuracy. To meet these requirements, we adopt the transformer as our default classifier when evaluating our explanation framework. Finally, the generated explanations should be interpretable for network operators. We formulate explanation generation as a mutual information maximization problem [40], ensuring that the most fundamental networking units (e.g., bytes and RTTs) are identified to maximize domain interpretability.

Addressing these challenges, we propose Traffic-Explainer, a novel explanation framework that identifies the most critical units/unit-unit interactions within a sequence for explaining DL-based decision-making:

- **Novel Explanation Problem:** We are the first to systematically investigate the explanation of DL-based networking applications.
- **A Systematic Explanation Framework:** We introduce a novel explanation framework, Traffic-Explainer, for deep learning-based sequential networking applications, bridging the gap between deep learning automation and network operator trust.
- **Illustration with Real-world Applications:** We demonstrate the efficacy of Traffic-Explainer via three use cases: traffic classification, country localization, and network cartography. For traffic classification, our explanations can be generated at both the local instance and global class level. Additionally, we showcase the practical impact of Traffic-Explainer by demonstrating its utility in manipulating country locations of traffic flows and mapping traceroutes to critical submarine cables, highlighting its potential for enhancing trustworthiness in network management.

2 Related Work

Explainable DL-based Networking Systems: DL-based models in networking applications often operate as black boxes, with parameters trained purely in a data-driven manner without domain expertise [44, 16, 15, 5, 8, 14, 1]. This raises concerns about the trustworthiness of their predictions, motivating the need for explanation techniques to build trust in automated network management. Explanation methods can be generally categorized into gradient, perturbation, surrogate, and decomposition-based methods [31, 28, 27, 9, 8]. Despite their widespread use, these techniques have rarely been applied in networking tasks [8], motivating us to explore their adaptations for networking applications. Specifically, our proposed explanation framework leverages a perturbation-based masking strategy to identify crucial units within a sequence for explaining traffic classification and detecting critical round-trip times for enhancing network cartography mapping.

Traffic Classification: Network traffic classification has been applied in application classification and anomaly detection [6]. Traditional methods rely on hand-crafted features like traffic fingerprints, which require extensive domain expertise and are limited to specific traffic scenarios [34, 19]. Recent approaches use deep learning (e.g., Transformers and Graph Neural Networks) to automatically learn representations for traffic classification [18, 41]. However, their black-box nature presents challenges in understanding their predictions, which necessitates the explanation techniques to delve deep into their decision-making process.

Network Cartography: Network cartography seeks to identify dependencies between the traffic along IP routes (i.e., layer 3) and the routes that the traffic takes within the underlying physical infrastructure (i.e., layer 1). Unfortunately, understanding the dependencies between IP routes and the underlying physical infrastructure is challenging. For one, an IP route between two endpoints on the Internet (such as those captured by `traceroute`) typically provides a list of IP addresses associated with the routers traversed by traffic between those endpoints. By itself, this *logical* representation of a route reveals little information about the underlying *physical* infrastructure. This decoupling between the network and physical layers makes it very difficult to reliably determine the (cross-layer) dependency. Therefore, trendy efforts focus on mapping specific portions of the traffic to the physical infrastructure. For example, Nautilus [25] focuses on mapping traceroutes to submarine infrastructure, whereas iGDB [4] focuses on charting traffic to terrestrial fiber-optic pathways. What is critically lacking is a holistic way to map traceroutes to both terrestrial and submarine fiber optic paths.

3 Preliminary

Given that many networking applications can be formulated as sequence-based tasks, such as traffic application classification and mapping logical traceroute to physical fiber mapping, we select sequence as our basic unit to conduct DL-based prediction, which generally includes value forecasting, category classification, and decision-making. Correspondingly, the explanation problem would be formulated as identifying the most important units within a sequence that maximally explain its DL-based prediction. Next, we introduce the notations and problem statement.

3.1 Mathematical Notations

Let $\mathcal{X} = \{(\mathcal{X}^i, \mathcal{Y}^i)\}_{i=1}^N$ represent a set of N sequences, where the i^{th} sequence consists of a series of units $\mathcal{X}^i = \{\mathcal{X}_j^i\}_{j=1}^{|\mathcal{X}^i|}$ and its corresponding one-hot label $\mathcal{Y}^i \in \{0,1\}^C$. In the context of **network traffic classification**, each sequence $\mathcal{X}^i$ corresponds to a sequence of bytes, where each byte $\mathcal{X}_j^i$ takes values in the range $[0, 255]$, and the label $\mathcal{Y}^i$ represents the downstream application (e.g., YouTube, Skype, and Facebook). In **network cartography**, each sequence $\mathcal{X}^i$ consists of logical-layer measurements, where each measurement $\mathcal{X}_j^i$ is a real-valued round-trip time (RTT), and the label $\mathcal{Y}^i$ corresponds to the physical infrastructure (e.g., fiber links). Assuming the traffic classifier as $g_{\mathbf{\Theta}_g}$ and our proposed Traffic-Explainer as $h_{\mathbf{\Theta}_h}$ parameterized respectively with $\mathbf{\Theta}_g, \mathbf{\Theta}_h$, we formulate the problem of sequence-level Explanation for Traffic Classification as:

3.2 Problem Statement

Given a set of sequences $\mathcal{X}$ in networking applications that are either bytes or logical-layer RTT, we aim to:

- **Learn an optimal classifier** $g_{\mathbf{\Theta}_g^*}$: $\mathcal{X} \to \widetilde{\mathcal{Y}}$ so that predicted class $\widetilde{\mathcal{Y}}$ matches ground-truth class $\mathcal{Y}$.
- **Learn an optimal explainer** $h_{\mathbf{\Theta}_h^*}$: $(\mathcal{X}, \widetilde{\mathcal{Y}}, g_{\mathbf{\Theta}_g^*}) \to \mathbf{M}$ so that the learned explanation $\mathbf{M}$ could maximally explain the prediction $\widetilde{\mathcal{Y}}$ made by the optimal classifier $g_{\mathbf{\Theta}_g^*}$ over the sequence $\mathbf{X}$.

Notably, the above notation and problem setup can be generalized to many sequence-based networking applications such as anomaly detection, network capacity planning, among others, in addition to traffic classification and networking cartography investigated in this paper. Building upon this formulation, we next introduce our basic transformer-based sequence classification and then present our Traffic-Explainer which identifies the most influential units among the sequences that drive classification decisions.

4 Framework

To ensure broad applicability across diverse networking applications, our DL-powered traffic prediction and explanation framework operates at the sequence level. Before introducing the sequence-based explainer, we first introduce the DL-powered traffic classifier.

4.1 Traffic Classifier

Since networking data inherently exhibit a sequential nature, such as traceroute data and traffic packet sequences, we construct a transformer-based model $g_{\mathbf{\Theta}_g}$ to obtain sequence embedding for classification.

4.1.1 Unit Tokenization Each sequence $\mathcal{X}^i$ consists of a set of units, where each unit $\mathcal{X}_j^i$ is mapped to a d-dimensional embedding via a learnable embedding matrix $\mathbf{E} \in \mathbb{R}^{|\mathcal{V}| \times d}$, where $|\mathcal{V}|$ denotes the vocabulary size. For categorical unit values, we initialize a learnable embedding matrix and use it to obtain the corresponding unit embeddings. For instance, in traffic classification, each byte b_k in a packet takes a value in $[0, 256]$, where 256 serves as a padding token to standardize packet lengths. Accordingly, we retrieve the embedding of the j^{th} unit as $\mathbf{e}_j = \mathbf{E}[\mathcal{X}_j^i, :], \forall\ \mathcal{X}_j^i \in \mathcal{X}^i$. For real-valued sequences (e.g., RTT in network cartography), the units are projected into a continuous embedding space via a learnable transformation matrix.

4.1.2 Self-Attention To capture dependencies among units within a sequence, we apply self-attention. Since ordering information is essential (e.g., byte order in packets or temporal order in RTTs), we incorporate positional encoding $\mathbf{\Phi}_j$ into each unit embedding. The transformer then applies self-attention over the sequence to produce context-aware representations:

$$\mathbf{h}_j^i = \text{Self-ATT}(\{(\mathbf{e}_j, \mathbf{\Phi}_j)\}_{j=1}^{|\mathcal{X}^i|}). \tag{4.1}$$

4.1.3 Pooling for Sequence Representation After obtaining unit-wise representations $\{\mathbf{h}_j^i\}_{j=1}^{|\mathcal{X}^i|}$, we aggregate them into a fixed-size sequence embedding via pooling operation:

$$\mathbf{F}^i = \text{Pooling}(\{\mathbf{h}_j^i\}_{j=1}^{|\mathcal{X}^i|}), \tag{4.2}$$

where the typical pooling operation could be mean-pooling, and the obtained sequence embedding $\mathbf{F}^i$ serves as the final representation for classification. Given the ground-truth class $\mathcal{Y}^i$, we optimize the classifier parameters $\mathbf{\Theta}_g$ by minimizing the cross-entropy loss:

$$\mathbf{\Theta}_g^* = \arg\min_{\mathbf{\Theta}_g} \sum_{i=1}^{N} \sum_{c=1}^{C} \mathcal{Y}_c^i \log \widehat{\mathbf{F}}_c^i, \tag{4.3}$$

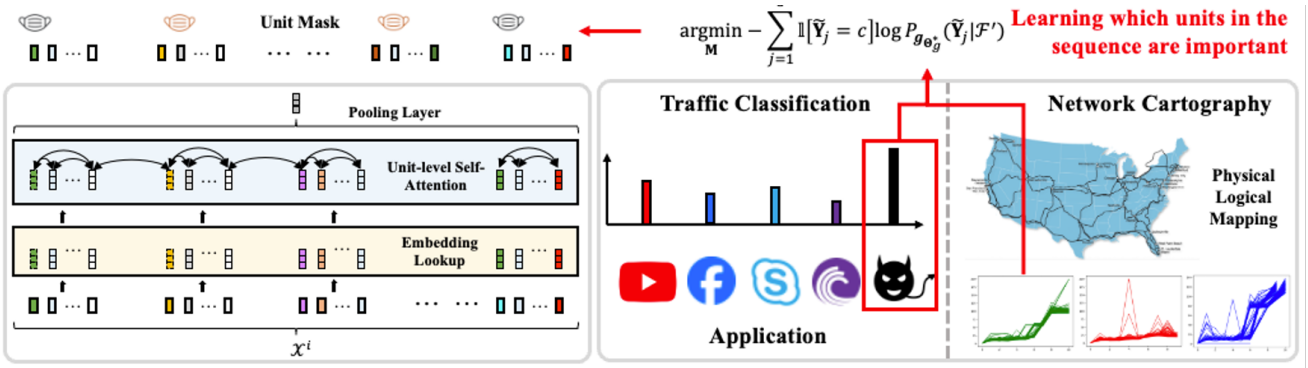


Figure 3.1: Given a traffic sequence $\mathcal{X}$ consisting of k units $\{\mathcal{X}_i\}_{i=1}^{k}$, our proposed transformer first predicts the sequence label, such as the application type in traffic classification or the optical fiber cable used in a specific traceroute. To generate explanations, we identify the most important units and unit-unit interactions by optimizing input masks via mutual information maximization.

where $\widehat{\mathbf{F}}^i \in \mathbb{R}^C$ is the predicted class distribution after applying a softmax layer. This formulation ensures that our transformer-based classifier remains flexible, supporting both traffic classification (with byte sequences) and network cartography (with RTT sequences). In the next, we introduce our Traffic-Explainer.

4.2 Traffic-Explainer

After introducing a transformer-based classifier, this section focuses on developing our proposed explanation framework, Traffic-Explainer. Since each sequence consists of fundamental units that compose the data structure across different applications and domains, we aim to identify the most important units responsible for classification decisions. Formally, given a sequence $\mathcal{X} = \{\mathcal{X}_j\}_{j=1}^{|\mathcal{X}|}$, we seek to extract a subset of units $\mathcal{X}' \subseteq \mathcal{X}$ that are most responsible for its prediction $\widetilde{\mathcal{Y}}$. Since the notion of a fundamental unit varies across applications (e.g., bytes in traffic classification, RTTs in network cartography), our Traffic-Explainer framework provides a generalizable approach by centering explanations on these core sequence components. For instance, bytes form the fundamental building blocks of traffic flows across diverse protocols, while RTTs offer ubiquitous network performance measurements. By identifying the most critical units for classification, our method enhances interpretability across heterogeneous domains. Moreover, our framework extends beyond individual units to capture unit-unit interactions by refining the masking mechanism to operate at the self-attention layer rather than the input level.

4.3 Explanation as Unit-level Mask

Given a sequence of units $\mathcal{X}^i$, to identify the most important units, we initialize a learnable unit masking matrix $\mathbf{M}$ with $\mathbf{M}_j$ denoting the importance of unit j in classifying the sequence. Subsequently, our classifier takes the masked sequence and makes the prediction, i.e., its first self-attention layer becomes:

$$\mathbf{h}_j^i = \text{Self-ATT}(\{(\mathbf{e}_j * \sigma(\mathbf{M}_j), \mathbf{\Phi}_j)\}_{j=1}^{|\mathcal{X}^i|}),\ \forall\ \mathcal{X}_k^i \in \mathcal{X}^i, \tag{4.4}$$

where σ is the sigmoid function mapping the mask score to a value between 0 and 1. The transformed units $\mathbf{h}_j^i$ are then aggregated via pooling for classification. A higher value of $\mathbf{M}_j$ indicates the higher importance of unit $\mathcal{X}_k^i$ in predicting the sequence label. We next focus on optimizing the unit-level masking matrix $\mathbf{M}$.

4.4 Explanation at the Local-Instance Level

For each sequence $\mathcal{X}^i$ with predicted class $\widetilde{\mathcal{Y}}^i = g_{\mathbf{\Theta}_g^*}(\mathcal{X}^i)$, we optimize explanation by maximizing mutual information between the prediction $\widetilde{\mathcal{Y}}^i$ and its explanation $\widetilde{\mathcal{X}}^i$ (i.e., a subset of units):

$$\max_{\widetilde{\mathcal{X}}^i} \text{MI}(\widetilde{\mathcal{Y}}^i, \widetilde{\mathcal{X}}^i) = H(\widetilde{\mathcal{Y}}^i) - H(\widetilde{\mathcal{Y}}^i|\widetilde{\mathcal{X}}^i) \tag{4.5}$$

For traffic sequence $\mathcal{X}^i$, MI quantifies the change in the prediction $\widetilde{\mathcal{Y}}^i$ when the sequence $\mathcal{X}^i$ is limited to the explained sub-sequence $\widetilde{\mathcal{X}}^i$. For example, consider the situation where removing the unit $\widetilde{\mathcal{X}}_j^i$ strongly decreases the probability of the prediction $\max_{c\in\mathcal{C}} \widetilde{\mathcal{Y}}_c^i$, the unit $\widetilde{\mathcal{X}}_j^i$ is naturally a good counterfactual explanation for the prediction of sequence $\mathcal{X}^i$. Maximizing the mutual information between the predicted label distribution $\widetilde{\mathcal{Y}}$ and explanation $\widetilde{\mathcal{X}}$ equals minimizing the conditional entropy $H(\widetilde{\mathcal{Y}}|\widetilde{\mathcal{X}})$:

$$\mathbf{M}^* = \arg\min_{\mathbf{M}} H(\widetilde{\mathcal{Y}}|\widetilde{\mathcal{X}}) = -\mathbb{E}_{\widetilde{\mathcal{Y}}|\mathcal{X}}[\log P_{g_{\mathbf{\Theta}_g^*}}(\widetilde{\mathcal{Y}}|\widetilde{\mathcal{X}})]. \tag{4.6}$$

The explanation for prediction $\widetilde{\mathcal{Y}}$ is thus a subsequence of units $\mathcal{X}$ that minimizes uncertainty of $g_{\mathbf{\Theta}_g}$, essentially

following the intuition that *the explanation should be the ones that if only keep the explanation and remove all other non-explanation parts, the model would become more confident about its original prediction.*

For networking management, rather than explain through the model confidence, the network operators sometimes care about "why does the trained traffic classifier predict a certain class label?". Therefore, we modify the conditional entropy objective in Eq. (4.6) with a cross-entropy objective between class and prediction:

$$\mathbf{M}^*(c) = \arg\min_{\mathbf{M}} -\sum_{j=1}^{C} \mathbb{1}[\widetilde{\mathcal{Y}}_j^i = c] \log P_{g_{\mathbf{\Theta}_g^*}}(\widetilde{\mathcal{Y}}_j^i | \widetilde{\mathcal{X}}). \tag{4.7}$$

The explanation for prediction $\widetilde{\mathcal{Y}}$ is thus a subsequence of units $\widetilde{\mathcal{X}}$ that, if only keep them and remove all other units in the sequence, would maximize the prediction score of model's original predicted class. We empirically find this objective slightly outperforms the previous confidence objective in explanation.

4.5 Explanation at the Global-Class Level

Beyond instance-level explanations, what is more interesting in real-world applications is why the model always makes certain predictions about a group of instances, e.g., the ones belonging to the same class. This motivates us to explain the predictions at the global class level. For example, identifying the most important units driving predictions towards certain classes.

$$\mathbf{M}^* = \arg\min_{\mathbf{M}} -\sum_{i=1}^{N}\sum_{j=1}^{C} \mathbb{1}[\widetilde{\mathcal{Y}}_j^i = c] \log P_{g_{\mathbf{\Theta}_g^*}}(\widetilde{\mathcal{Y}}_j^i | \widetilde{\mathcal{X}}) \tag{4.8}$$

4.6 Mask Regularization

Blindly optimizing the mask $\mathbf{M}$ according to the above explanation-based objectives Eq. (4.6)-(4.8) may lead to trivial explanations, e.g., $\widetilde{\mathcal{X}} = \mathcal{X}$, where all units in the original sequence are tagged important, as it would naturally encompass all the information necessary to explain the model prediction. Moreover, a widely adopted assumption in feature selection and sparse representation learning is that model prediction should primarily be attributed to a subset of the inputs [17]. To prevent the trivial use of the entire sequence as the explanation and also consider the principle of sparsity, we impose a predefined budget B to limit the set size of the explanation units. Under this constraint, we formulate the final loss with mask regularization as:

$$\mathbf{M}^* = \arg\min_{\mathbf{M}} \mathcal{L} = \mathrm{ReLU}(||\mathbf{M}||_1 - B) + \mathcal{L}^{\mathrm{Explain}}, \tag{4.9}$$

where $\mathcal{L}^{\mathrm{Explain}}$ could refer to any of the previous three explanation loss defined in Eq (4.6)-(4.8). Intuitively, minimizing $\mathcal{L}$ identifies the most informative units while *maximally excluding units irrelevant to the explanation, thereby refining the final explanation.* Although the presented framework only focuses on identifying the most important units, it can be easily tailored to identify the most important unit-unit level interactions by masking the self-attention among units. Therefore, we also include this level of explanation in Table 5.1.

5 Experiment Setting

We demonstrate the practical usage of Traffic-Explainer via three applications, detailing their objectives, dataset collection, baseline methods, evaluation strategies, and hyperparameter settings. Details are in Appendix A.

- **Application 1 - Traffic Classification [18, 41]**: Downstream applications are classified by traffic flow where the sequences represent the set of traffic flow consisting of sequential bytes. Here, the byte is the basic unique. The explanation is to identify the most important bytes for predicted application.
- **Application 2 - Country Localization [24, 38]**: aims to localize the country where the traffic happens and can be similarly formulated as traffic classification to Application 1. The explanation is to identify the most important bytes for localized countries.
- **Application 3 - Network Cartography [26]**: aims to map logical traceroutes (a sequence of RTT) to physical fiber-optic cables. The explanation is to identify the most important RTT hops for the mapped physical cables.

5.1 Dataset Collection

For the application 1, Following [41], we validate the proposed Traffic-Explainer on four datasets: **ISCX-VPN**, **ISCX-NonVPN**, **ISCX-Tor**, and **ISCX-NonTor** [41]. we use SplitCap to obtain bidirectional flows and increase the training samples in ISCX-Tor by dividing each flow into 60-second non-overlapping blocks. **For the application 2**, two datasets **IOS-Cross Platform** and **Android-Cross Platform** [36, 24] are used to showcase Traffic-Explainer in identifying traffic country localization. These two datasets comprise user-generated data for 215 Android and 196 iOS apps in the US, China, and India. **For application 3**, we collect traceroutes from predefined source locations to destinations. Using RIPE Atlas probes [13], we select target countries and direct probes to a set of international servers. Over a three-day period, we collected 5000 unique source-to-destination traceroutes, including Palo Alto, US to Yokohama, Japan; Rockville, Maine, US to Amsterdam, Netherlands; and Vancouver, Canada to Washington, US.

Object	Explainer		ISCX-VPN Fid	ISCX-VPN Acc	ISCX-VPN C-Fid	ISCX-VPN C-Acc	ISCX-nonVPN Fid	ISCX-nonVPN Acc	ISCX-nonVPN C-Fid	ISCX-nonVPN C-Acc	ISCX-Tor Fid	ISCX-Tor Acc	ISCX-Tor C-Fid	ISCX-Tor C-Acc	ISCX-nonTor Fid	ISCX-nonTor Acc	ISCX-nonTor C-Fid	ISCX-nonTor C-Acc
Byte Level	Random	1%	31.9	31.2	1.27	4.46	22.5	22.8	1.5	10.6	13.2	9.2	0.57	17.8	43.6	43.9	0.29	2.88
		5%	35.7	35.7	4.50	8.30	25.6	25.8	4.60	13.4	14.4	12.6	3.50	21.3	42.4	42.7	1.80	3.40
		10%	39.5	40.8	9.60	12.1	28.6	28.9	7.90	12.9	16.1	12.6	5.80	23.0	42.9	43.2	4.50	6.10
	Saliency Map	1%	33.8	35.0	6.40	9.60	23.3	23.5	0.76	10.1	31.6	22.4	29.3	42.5	37.9	38.1	3.00	4.90
		5%	69.4	72.0	50.3	52.2	40.0	39.5	14.9	18.7	32.8	23.0	28.7	36.2	31.8	31.8	12.5	13.2
		10%	71.3	72.0	53.5	52.9	44.6	43.3	22.0	24.1	46.0	36.8	43.7	52.9	51.8	51.5	21.7	21.8
	Byte-Explainer	1%	**82.8**	**81.5**	**58.6**	**56.7**	**65.3**	**65.1**	**29.1**	**33.2**	**44.3**	**39.1**	**49.4**	**50.6**	**74.5**	**74.8**	**20.2**	**20.9**
		5%	**97.5**	**92.4**	**82.8**	**79.6**	**92.4**	**84.6**	**78.5**	**74.9**	**92.0**	**77.6**	**86.8**	**88.5**	**96.1**	**95.4**	**71.1**	**70.5**
		10%	**98.7**	**93.0**	**84.1**	**80.9**	**96.2**	**87.3**	**79.8**	**76.5**	**97.7**	**81.0**	**86.8**	**90.2**	**97.9**	**96.0**	**77.2**	**76.8**
Byte-Byte Level	Random	1%	28.7	27.4	0.00	5.70	34.9	35.7	0.51	10.4	16.7	16.7	0.57	18.4	17.7	17.4	0.83	2.70
		5%	35.0	33.8	0.64	5.10	40.5	40.3	1.30	10.1	27.6	31.6	3.50	19.5	26.0	25.8	0.29	2.80
		10%	38.9	36.9	1.90	5.10	52.2	51.9	2.00	10.4	31.6	36.2	4.00	20.7	55.3	55.0	0.42	2.70
	Self-Attention	1%	93.6	**93.6**	8.90	10.2	93.9	**89.1**	8.90	12.7	53.5	56.9	9.20	14.4	37.9	38.1	3.00	4.90
		5%	93.6	**93.0**	**74.5**	**76.4**	97.0	**89.1**	**43.5**	**44.1**	69.5	61.5	46.0	52.3	31.8	31.8	12.5	13.2
		10%	**95.5**	**93.6**	**79.0**	**80.9**	98.2	**89.6**	**56.5**	**56.7**	71.3	61.5	59.8	64.4	51.8	51.5	21.7	21.8
	Byte-Explainer	1%	**97.5**	92.4	**56.1**	**52.2**	**96.0**	87.1	**38.7**	**34.4**	**74.7**	**63.8**	**73.0**	**69.0**	**96.8**	**94.6**	**48.5**	**47.6**
		5%	**98.1**	92.4	58.6	58.0	**99.0**	88.6	37.0	36.2	**93.7**	**75.3**	**74.7**	**70.1**	**97.1**	**94.6**	**60.4**	**60.0**
		10%	94.9	89.2	56.7	58.0	**98.7**	88.4	38.5	39.8	**98.3**	**79.9**	**74.7**	**73.0**	**96.9**	**94.4**	**61.3**	**61.1**

Table 5.1: Comparing local instance-level explanation. The best and runner-up results under the same byte budget are in **bold** and underlined. Our Traffic-Explainer exhibits significantly higher explanation quality.

5.2 Baselines/Evaluation/Hyperparameters

We compare our traffic classifier with the following baselines: GRU-based FS-Net [19], Transformer-based ET Bert [18], and GNN-based TFE-GNN [41]. To benchmark the explanation by the proposed Traffic-Explainer, in the first traffic application, we compare it with three explanation baselines: Random [22], Saliency Map [33], Self-Attention [11], details of which are attached in Appendix B. We use fidelity/accuracy and counterfactual-fidelity/accuracy as key metrics. To calculate the above four metrics, we first obtain the explanation as the Top-K important bytes and either keep or remove them to check the performance variation. (Counterfactual-)Fidelity is the ratio of predictions that (differ from) remain the same as the predicted class after (removing) retaining only the obtained explanation. (Counterfactual-)Accuracy is the ratio of predictions that (differ from) remain the same as the ground-truth class after (removing) retaining only the obtained explanation. More detailed computation is attached in the Appendix. To prepare the traffic flow data, we set the maximum packet number of one sample to 50. The max payload byte length and the max header byte length are set to 150 and 40, respectively. In the training stage, we set the max training epoch to 1000. We tune the following hyperparameters: the batch size in $[64, 512, 4096]$, the learning ratio in $[0.001, 0.01]$, and the dropout rate in $[0.2, 0.5]$.

To validate the Traffic-Explainer, we use our proposed transformer as the default traffic classifier for its optimal trade-off between performance and efficiency (Table D.1). The classifier is trained and then used to generate explanations at both the unit and byte-byte interaction levels. At the byte level, explanations are byte scores, i.e., $\mathbf{M} \in \mathbb{R}^{257}$, while byte-byte interactions are captured as $\mathbf{M} \in \mathbb{R}^{257\times257}$. We rank importance scores, select Top-K bytes, and compute (counterfactual)-fidelity/accuracy.

6 Application1 - Traffic Classification

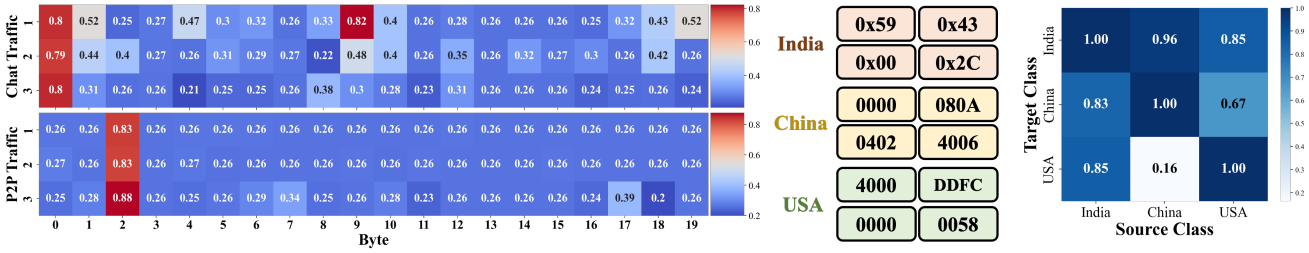
6.1 Local-Instance Level Explanation

In Table 5.1, the local-instance level explanations by our proposed Traffic-Explainer generally exhibit higher quality than those produced by other baselines. For byte-level explanations, the performance margin is even more significant. Impressively, the Traffic-Explainer requires only 5% of bytes (approximately $256 \times 5\% \approx 10$ bytes) to explain half of the traffic flow predictions for all traffic flow datasets. This finding suggests that certain bytes are particularly salient and closely associated with each traffic class. Among the two baselines, the Saliency Map outperforms the Random method due to its gradient-based approach to weighting the importance of bytes. The Saliency Map is inferior to Traffic-Explainer because it only considers the individual importance of bytes, overlooking their cooperation effect in contributing to the prediction. In contrast, Traffic-Explainer, by jointly optimizing to identify the Top-K important bytes, inherently accounts for these byte interactions. Moreover, as the explanation size increases (i.e., the amount of selected bytes increases), the performance of all three baselines increases.

For byte-byte interaction level explanations, we replace the Saliency Map with the Self-Attention Baseline, as self-attention mechanisms naturally capture byte-byte interactions, whereas the Saliency Map lacks a clear definition of such interactions. Traffic-Explainer achieves high explanation quality across most cases, although it performs slightly lower than the Self-Attention Baseline on the ISCX-VPN and ISCX-NonVPN datasets when evaluated with Acc and C-Fid metrics. We hypothesize that this is because the selected Top-K byte-byte interactions in Self-Attention are applied across all self-attention layers of the transformer, causing more influence on the model decision process.

Object	Explainer		ISCX-VPN Fid	ISCX-VPN Acc	ISCX-VPN C-Fid	ISCX-VPN C-Acc	ISCX-nonVPN Fid	ISCX-nonVPN Acc	ISCX-nonVPN C-Fid	ISCX-nonVPN C-Acc	ISCX-Tor Fid	ISCX-Tor Acc	ISCX-Tor C-Fid	ISCX-Tor C-Acc	ISCX-nonTor Fid	ISCX-nonTor Acc	ISCX-nonTor C-Fid	ISCX-nonTor C-Acc
Byte Level	Saliency Map	1%	35.7	35.7	14.0	15.9	**25.9**	**19.5**	**27.6**	**40.2**	**27.9**	**26.3**	**8.10**	**13.7**	24.0	24.3	3.46	4.71
		5%	66.9	69.4	51.6	54.1	30.5	**24.1**	**36.2**	**42.5**	41.5	41.5	**23.3**	**26.8**	58.5	58.8	16.5	17.2
		10%	75.2	75.2	56.1	57.3	51.2	48.9	53.5	57.5	43.0	41.3	33.4	34.9	83.4	84.0	29.0	29.3
	Byte-Explainer	1%	**76.4**	**76.4**	**40.8**	**41.4**	16.7	12.6	11.5	21.8	25.3	24.3	4.56	11.1	**45.6**	**46.1**	**3.92**	**5.04**
		5%	**94.9**	**93.0**	**68.2**	**68.2**	**32.2**	**24.1**	33.9	40.8	**51.7**	**47.1**	**23.3**	26.3	**81.2**	**81.6**	**29.3**	**29.3**
		10%	**98.7**	**94.9**	**75.2**	**75.2**	**64.9**	**55.8**	**74.7**	**78.2**	**83.0**	**78.0**	**47.3**	**47.1**	**91.5**	**91.2**	**52.6**	**52.6**

Table 5.2: Comparison of global class-level explanation. In general, our proposed Traffic-Explainer exhibits higher quality in their generated explanation. We omit the results for **Random** due to their inferior performance.



(a) (b) (c)

Figure 5.1: **(a)** By visualizing the byte importance scores for each traffic flow instance in Chat and P2P applications, network operators can verify the decision-making process of the traffic classifier with domain expertise. **(b)** We visualize Top-4 important bytes among the global-class level explanation for each country. **(c)** We successfully manipulate the country prediction by switching important bytes between traffic sequences in different countries. More explanation results are in the Table A.1 in Appendix.

6.2 Global-Class Level Explanation

While local-instance explanations are valuable for understanding the model's decision-making process for each traffic flow, certain applications demand understanding a group of instances. In these cases, identifying common patterns that influence a group can reveal insights going beyond individual cases. Therefore, we extend our analysis to include global-class level explanations shown in Table 5.2. In general, our Traffic-Explainer still achieves the best explanation performance. Compared with the instance-level explanation in Table 5.1, the lower quality of the class-level explanation is due to the increased variability in aggregating explanations across a broader set of traffic flows.

6.3 Explanation Visualization

After quantitatively validating the quality of explanations generated by our proposed Traffic-Explainer, we qualitatively visualize the byte importance scores for three traffic flow sequences from the Chat and P2P classes. Figure 5.1(a) [1] illustrates a distinct byte importance pattern, where bytes 0, 1, and 9 are crucial for identifying traffic of application Chat, while byte 2 plays the most significant role in distinguishing traffic of class P2P. Furthermore, the explanation also differs across different flow sequences even though they belong to the same traffic application class.

[1] Here, we omit the real byte value on the x-axis to save space.

7 Application2 - Country Localization

Here, we present another real-world case study of our proposed Traffic-Explainer on country localization, highlighting both its adversarial and trustworthy implications. When users access the internet in specific geographical areas, such as a particular country, their traffic patterns reflect regional infrastructures and traffic locality [21]. Traffic-Explainer can analyze these patterns to infer the country of origin and identify the bytes contributing to the location prediction. In Figure 5.1(b), we visualize the most important bytes for countries like India, China, and the USA generated by using our Traffic-Explainer framework, which raises privacy concerns. Moreover, malicious actors could exploit this framework to manipulate traffic patterns and alter country-level predictions. By intentionally swapping certain crucial patterns of information in the format of bytes between traffic flows from different classes/countries, attackers could conceal harmful activities by making traffic appear as if it originated from a different country. We demonstrate this by swapping the Top-10% important bytes generated by our Traffic-Explainer between classes on the IOS/Android Cross-Platform datasets, resulting in high transformation rates—e.g., 83% of Indian traffic is manipulated to appear as in China (Figure 5.1(c)). Note that our byte-swapping operation involves exchanging 16-bit hex values between two valid sequences, so the checksums remain correct, and the network traffic is still valid after swapping bytes.

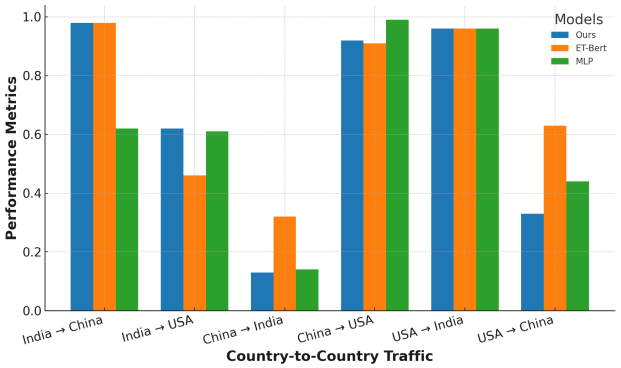


Figure 7.1: Transformation rates among traffic sequences from different classes and countries remain consistently high across various traffic classifiers. This consistency suggests that our Traffic-Explainer provides generalizable explanations applicable to multiple traffic classification models. The results indicate that Traffic-Explainer identifies causal bytes specific to each country, rather than overfitting to spurious correlations.

To further demonstrate that the derived explanation (i.e., the most important bytes) can be generalized to traffic sequences from different classes/countries, we calculate the transformation rates in Figure 7.1 by swapping the explanation—specifically, the most important bytes identified by Traffic-Explainer between traffic sequences from different classes or countries and then applying traffic classifiers. These high transformation rates observed consistently across diverse classifiers, demonstrate the generalizability of the identified explanations (key bytes) across different traffic classification models. This finding suggests that Byte-Explainer effectively identifies causal bytes associated with each country rather than overfitting to spurious byte patterns correlated with predictions from our hierarchical transformer model. These results highlight the robustness and reliability of Byte-Explainer in pinpointing critical bytes for traffic classification. Beyond adversarial applications, the analysis in Figure 5.1(b) allows us to visualize the most important bytes for countries like India, China, and the USA, offering valuable insights into regional traffic characteristics. This capability can enhance network management by enabling more precise geo-based services for a wide variety of applications (e.g., regional-specific content delivery, targeted online advertising, accurate mapping services, geofencing and location-based authentication, regional matchmaking in online gaming, etc.) [29]. Byte-Explainer also offers benefits from a social-good perspective. This tool could protect users by obfuscating or anonymizing byte patterns linked to specific locations in regions with government surveillance or censorship.

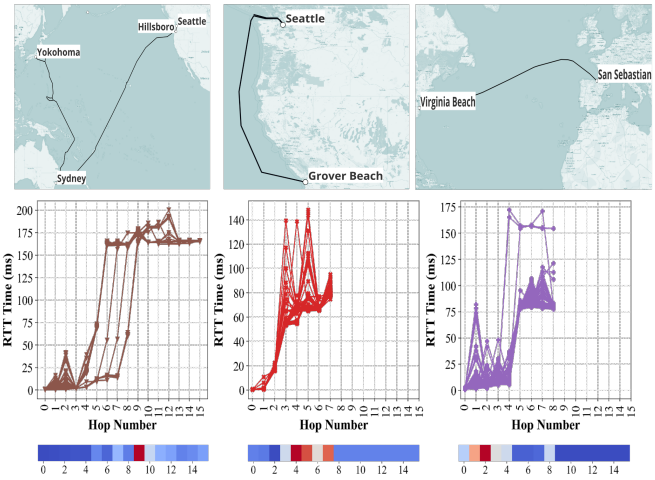


Figure 7.2: Top-Three distinct submarine cables used for transmitting traceroute data. Middle-Sequences of RTT measurements for different traceroute paths: (a) US → Australia → Japan (b) Seattle → San Jose (c) US → France. Bottom-Explanation of how specific RTT hops reveal insights into submarine cable routes and their cartographic significance.

7.1 Application3 - Network Cartography

The Traffic-Explainer is further used to address the traffic mapping challenge in network cartography, where the objective is to establish cross-layer dependencies between logical IP routes and the underlying physical infrastructure. We collaborate with domain experts to collect traceroute data using tools such as RIPE Atlas and CAIDA Ark [10]. After preprocessing, we employ IP geolocation services to associate IP addresses with precise physical locations, incorporating latitude and longitude coordinates [12]. Additionally, we analyze traceroutes that traverse both terrestrial and submarine fiber-optic networks, classifying them based on RTT characteristics. The key insight is that traffic flowing through the same physical link—such as a specific submarine cable—should exhibit similar RTT patterns. Traffic-Explainer helps uncover these patterns, providing a holistic view of how different fiber-optic pathways influence traffic behavior and performance. We detail in four steps below.

Step 1: Data Collection. The data collection process involves gathering traceroutes from predefined source locations to designated destinations. Using RIPE Atlas probes [13], we select target countries and direct probes to a consistent set of international servers. For example, in a case study analyzing transpacific traffic, we selected the United States as the source and Japan as the destination. Over a three-day period, we collected nearly 5000 traceroute data, including Palo

Alto, US to Yokohama, Japan; Rockville, Maine, US to Amsterdam, Netherlands; and Vancouver, Canada to Washington, US. These traceroutes traverse both submarine and terrestrial fiber-optic networks, enabling cross-layer analysis of network infrastructure.

Step 2: Data Preprocessing and Geolocation. Once collected, the traceroute data is preprocessed to extract key hop-level details, including hop number, IP address, and source-destination pair. To assess the consistency of multiple traceroutes between the same endpoints, we visualize RTT variations across different probe times (see middle line charts in Figure 7.2). These visualizations help identify potential deviations in traffic flow and latency variations that indicate the use of different physical network paths – especially submarine cables, as indicated by the sharp increase in RTTs.

For geolocation, we use services such as IPGeolocation, IPLocation.net, and IPinfo to determine the approximate physical location of each hop [7]. The extracted geographic coordinates are then mapped using QGIS, providing a spatial representation of network paths. This geolocation data is crucial for identifying submarine and terrestrial infrastructure used by the traffic, helping establish physical dependencies in network routing.

Step 3: Mapping Traceroutes to Submarine Infrastructure. To map traceroutes to submarine cable infrastructure, we analyze RTT characteristics and geolocation data. First, we identify RTT spikes along traceroutes, as submarine cables typically introduce higher latency due to increased physical distance and transmission delay. Next, we overlay the preprocessed traceroute paths onto QGIS, mapping their geographic coordinates to known submarine cable locations. Additionally, we perform DNS reverse lookups to correlate IP addresses with submarine cable domains, further validating the inferred physical pathways. By linking domain names to submarine cable infrastructure, we identify the most probable submarine routes taken by the traffic, which are collected as a few labeled training sets for instantiating the semi-supervised submarine cable mapping/classification based on RTT characteristics. The hypothesis is that traffic flowing through the same submarine cable exhibits similar RTT patterns [32]. This classification enables a more refined understanding of how submarine cables influence network performance, congestion, and routing decisions.

Step 4: Traffic-Explainer for Route Interpretation. The mapped traceroute data is further analyzed using Traffic-Explainer, which enhances the interpretation of traffic flow through submarine and terrestrial networks. By overlaying traceroutes onto submarine cable maps, we verify their alignment with known physical routes. Visualizing RTT spikes along the path provides deeper insights into where traffic transitions onto submarine cables, highlighting latency variations across terrestrial vs. submarine network paths.

More broadly, by leveraging Traffic-Explainer, network operators improve network observability and better understand how submarine cables impact traffic latency, routing behaviors, and performance. This approach significantly enhances their ability to map logical-layer traffic onto the underlying physical infrastructure, advancing the field of network cartography.

8 Conclusion

Despite the successful adoption of deep learning-powered models in networking applications, most of them focus on boosting performance without considering the underlying reasons behind their decision-making process, which jeopardizes the trustworthy adoption of DL solutions by network operators. This motivates us to introduce an explanation framework, Traffic-Explainer, to explain traffic predictions by identifying the important units and unit-unit interactions in given sequences. Extensive real-world evaluations validate the effectiveness of Traffic-Explainer in generating both local-instance and global-class explanations, demonstrating its potential to advance trustworthy network management.

References

[1] M. Abdullahi, Y. Baashar, H. Alhussian, A. Alwadain, N. Aziz, L. F. Capretz, and S. J. Abdulkadir, *Detecting cybersecurity attacks in internet of things using artificial intelligence methods: A systematic literature review*, Electronics, 11 (2022), p. 198.

[2] A. Adadi and M. Berrada, *Peeking inside the black-box: a survey on explainable artificial intelligence (xai)*, IEEE access, 6 (2018), pp. 52138–52160.

[3] J. Amann, A. Blasimme, E. Vayena, D. Frey, V. I. Madai, and P. Consortium, *Explainability for artificial intelligence in healthcare: a multidisciplinary perspective*, BMC medical informatics and decision making, 20 (2020), pp. 1–9.

[4] S. Anderson, L. Salamatian, Z. S. Bischof, A. Dainotti, and P. Barford, *igdb: connecting the physical and logical layers of the internet*, ACM Meas, (2022), p. 433–448, https://doi.org/10.1145/3517745.3561443, https://doi.org/10.1145/3517745.3561443.

[5] C. Ardi, C. Aubry, B. Kocoloski, D. DeAngelis, A. Hussain, M. Troglia, and S. Schwab, *The darpa searchlight dataset of application network traffic*, in Proceedings of the 15th Workshop on Cyber Security Experimentation and Test, CSET '22, New York, NY, USA, Aug. 2022, Association for Computing Machinery, https://doi.org/10.1145/3546096.3546103, https://doi.org/10.1145/3546096.3546103.

[6] A. Azab, M. Khasawneh, S. Alrabaee, K.-K. R. Choo, and M. Sarsour, *Network traffic classification: Techniques, datasets, and challenges*, Digital Communications and Networks, 10 (2024), pp. 676–692.

[7] O. Dan, V. Parikh, and B. D. Davison, *Ip geolocation through geographic clicks*, ACM Trans. Spatial Algorithms Syst., 8 (2022), https://doi.org/10.1145/3476774, https://doi.org/10.1145/3476774.

[8] A. Elfandi, H. Sagalyn, R. Durairajan, and W. Willinger, *Bootstrapping Trust in ML4Nets Solutions with Hybrid Explainability*, in 2024 3rd ACM Workshop on Practical Adoption Challenges of ML for Systems, ACM, 2024.

[9] Q. Feng, N. Liu, F. Yang, R. Tang, M. Du, and X. Hu, *Degree: Decomposition based explanation for graph neural networks*, arXiv preprint arXiv:2305.12895, (2023).

[10] M. Gharaibeh, A. Shah, B. Huffaker, H. Zhang, R. Ensafi, and C. Papadopoulos, *A look at router geolocation in public and commercial databases*, in Proceedings of the 2017 Internet Measurement Conference, 2017, pp. 463–469.

[11] Y. Hao, L. Dong, F. Wei, and K. Xu, *Self-attention attribution: Interpreting information interactions inside transformer*, in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, 2021, pp. 12963–12971.

[12] C. Hauff and G.-J. Houben, *Placing images on the world map: a microblog-based enrichment approach*, ACM Meas, (2012), p. 691–700, https://doi.org/10.1145/2348283.2348376, https://doi.org/10.1145/2348283.2348376.

[13] T. Holterbach, C. Pelsser, R. Bush, and L. Vanbever, *Quantifying interference between measurements on the ripe atlas platform*, ACM Meas, (2015), p. 437–443, https://doi.org/10.1145/2815675.2815710, https://doi.org/10.1145/2815675.2815710.

[14] N. Kaloudi and J. Li, *The ai-based cyber threat landscape: A survey*, ACM Computing Surveys (CSUR), 53 (2020), pp. 1–34.

[15] J. Knofczynski, R. Durairajan, and W. Willinger, *Arise: A multitask weak supervision framework for network measurements*, IEEE Journal on Selected Areas in Communications, 40 (2022).

[16] Y. Lavinia, R. Durairajan, R. Rejaie, and W. Willinger, *Challenges in using ml for networking research: How to label if you must*, in Proceedings of the Workshop on Network Meets AI & ML, 2020, pp. 21–27.

[17] J. Li, K. Cheng, S. Wang, F. Morstatter, R. P. Trevino, J. Tang, and H. Liu, *Feature selection: A data perspective*, ACM computing surveys (CSUR), 50 (2017), pp. 1–45.

[18] X. Lin, G. Xiong, G. Gou, Z. Li, J. Shi, and J. Yu, *Et-bert: A contextualized datagram representation with pre-training transformers for encrypted traffic classification*, in Proceedings of the ACM Web Conference 2022, 2022, pp. 633–642.

[19] C. Liu, L. He, G. Xiong, Z. Cao, and Z. Li, *Fs-net: A flow sequence network for encrypted traffic classification*, in IEEE INFOCOM 2019-IEEE Conference On Computer Communications, IEEE, 2019, pp. 1171–1179.

[20] S. M. Lundberg and S.-I. Lee, *A unified approach to interpreting model predictions*, Advances in neural information processing systems, 30 (2017).

[21] E. J. Malecki, *The economic geography of the internet's infrastructure*, Economic geography, 78 (2002), pp. 399–424.

[22] Y. Nian, Y. Chang, W. Jin, and L. Lin, *Globally interpretable graph learning via distribution matching*, in Proceedings of the ACM on Web Conference 2024, 2024, pp. 992–1002.

[23] D. Pujol Perich, J. R. Suárez-Varela Maciá, S. Xiao, B. Wu, A. Cabellos Aparicio, and P. Barlet Ros, *Netxplain: Real-time explainability of graph neural networks applied to networking*, ITU Journal on future and evolving technologies, 2 (2021), pp. 57–66.

[24] C. Qian, X. Li, Q. Wang, G. Zhou, and H. Shao, *Netbench: A large-scale and comprehensive network traffic benchmark dataset for foundation models*, arXiv preprint arXiv:2403.10319, (2024).

[25] A. Ramanathan and S. Abdu Jyothi, *Nautilus: A framework for cross-layer cartography of submarine cables and ip links*, Proc. ACM Meas. Anal. Comput. Syst., 7 (2023), https://doi.org/10.1145/3626777, https://doi.org/10.1145/3626777.

[26] A. Ramanathan and S. Abdu Jyothi, *Towards efficient and scalable internet cross-layer mapping*, in Proceedings of the CoNEXT on Student Workshop 2024, CoNEXT-SW '24, New York, NY, USA, 2024, Association for Computing Machinery, p. 11–12, https://doi.org/10.1145/3694812.3699931, https://doi.org/10.1145/3694812.3699931.

[27] M. T. Ribeiro, S. Singh, and C. Guestrin, *" why should i trust you?" explaining the predictions of any classifier*, in Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, 2016, pp. 1135–1144.

[28] M. Robnik-Šikonja and M. Bohanec, *Perturbation-based explanations of prediction models*, Human and Machine Learning: Visible, Explainable, Trustworthy and Transparent, (2018), pp. 159–175.

[29] K. Rusek, J. Suárez-Varela, P. Almasan, P. Barlet-Ros, and A. Cabellos-Aparicio, *Routenet: Leveraging graph neural networks for network modeling and optimization in sdn*, IEEE Journal on Selected Areas in Communications, 38 (2020), pp. 2260–2270.

[30] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, *Grad-cam: Visual explanations from deep networks via gradient-based localization*, in Proceedings of the IEEE international conference on computer vision, 2017, pp. 618–626.

[31] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, *Grad-cam: visual explanations from deep networks via gradient-based localization*, International journal of computer vision, 128 (2020), pp. 336–359.

[32] S. Sengupta, H. Kim, and J. Rexford, *Continuous in-network round-trip time monitoring*, ACM Meas, (2022), https://doi.org/10.1145/3544216.3544222, https://doi.org/10.1145/3544216.3544222.

[33] K. Simonyan, A. Vedaldi, and A. Zisserman, *Deep inside convolutional networks: Visualising image classification models and saliency maps*, arXiv preprint arXiv:1312.6034, (2013).

[34] P. Sirinam, M. Imani, M. Juarez, and M. Wright, *Deep fingerprinting: Undermining website fingerprinting defenses with deep learning*, in Proceedings of the 2018 ACM SIGSAC conference on computer and communications security, 2018, pp. 1928–1943.

[35] V. F. Taylor, R. Spolaor, M. Conti, and I. Martinovic, *Robust smartphone app identification via encrypted network traffic analysis*, IEEE Transactions on Information Forensics and Security, 13 (2017), pp. 63–78.

[36] T. Van Ede, R. Bortolameotti, A. Continella, J. Ren, D. J. Dubois, M. Lindorfer, D. Choffnes, M. Van Steen, and A. Peter, *Flowprint: Semi-supervised mobile-app fingerprinting on encrypted network traffic*, in Network and distributed system security symposium (NDSS), vol. 27, 2020.

[37] S. Verma, J. Dickerson, and K. Hines, *Counterfactual explanations for machine learning: A review*, arXiv preprint arXiv:2010.10596, 2 (2020), p. 1.

[38] Q. Wang, C. Qian, X. Li, Z. Yao, and H. Shao, *Lens: A foundation model for network traffic*, arXiv preprint arXiv:2402.03646, (2024).

[39] Y. Wang, T. Zhang, X. Guo, and Z. Shen, *Gradient based feature attribution in explainable ai: A technical review*, arXiv preprint arXiv:2403.10415, (2024).

[40] Z. Ying, D. Bourgeois, J. You, M. Zitnik, and J. Leskovec, *Gnnexplainer: Generating explanations for graph neural networks*, Advances in neural information processing systems, 32 (2019).

[41] H. Zhang, L. Yu, X. Xiao, Q. Li, F. Mercaldo, X. Luo, and Q. Liu, *Tfe-gnn: A temporal fusion encoder using graph neural networks for fine-grained encrypted traffic classification*, in Proceedings of the ACM Web Conference 2023, 2023, pp. 2066–2075.

[42] H. Zhao, H. Chen, F. Yang, N. Liu, H. Deng, H. Cai, S. Wang, D. Yin, and M. Du, *Explainability for large language models: A survey*, ACM Transactions on Intelligent Systems and Technology, 15 (2024), pp. 1–38.

[43] Y. Zhao, Y. Wang, and T. Derr, *Fairness and explainability: Bridging the gap towards fair model explanations*, in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, 2023, pp. 11363–11371.

[44] W. Zheng, C. Gou, L. Yan, and S. Mo, *Learning to classify: A flow-based relation network for encrypted traffic classification*, in Proceedings of The Web Conference 2020, 2020, pp. 13–22.

A Datasets

We use six datasets to verify the effectiveness of our proposed hierarchical self-attention and Traffic-Explainer framework. The details of each dataset are introduced as follows:

- **Application 1 - ISCX VPN/nonVPN**: A public traffic dataset including ISCX-VPN and ISCX-nonVPN datasets. The ISCX-VPN dataset is collected over virtual private networks (VPNs), used for accessing some blocked websites or services, and is difficult to recognize due to the obfuscation technology. Conversely, the traffic in ISCX-nonVPN is regular and not collected over VPNs.
- **Application 1 - ISCX Tor/NonTor**: ISCX Tor-nonTor is a public dataset, and the ISCX-Tor dataset is collected over the onion router, whose traffic can be difficult to trace. Besides, ISCX-nonTor is also regular and not collected over the onion router.
- **Application 2 - IOS/Android**: The Cross-Platform dataset comprises user-generated data for 215 Android and 196 iOS apps. The iOS apps were gathered from the top 100 apps in the US, China, and India App Store. The Android apps originate from the top 100 apps in the Google Play Store in the US and India, plus from the top 100 apps of the Tencent MyApps and 360 Mobile Assistant stores, as Google Play is unavailable in China. Each app was executed between three and ten minutes while receiving real user inputs. Procedures to install, interact, and uninstall the apps were given to student researchers who followed them to complete the experiments while collecting data. We use this dataset to evaluate our method's performance with user-generated data and the performance between different operating systems.
- **For application 3 - Network Traceroute Data**, we collect traceroutes from predefined source locations to destinations. Using RIPE Atlas probes [13], we select target countries and direct probes to a set of international servers. Over a three-day period, we collected 5000 unique source-to-destination traceroutes, including Palo Alto, US to Yokohama, Japan; Rockville, Maine, US to Amsterdam, Netherlands; and Vancouver, Canada to Washington, US.

We follow [41] to preprocess the first four synthetic traffic flow datasets and follow [24] to preprocess the last two real-world traffic flow datasets. The comprehensive statistics of each dataset is presented in Table .1. Note that to avoid any overfitting bias, unlike [41], we also add the validation split following ratio: 80%/10%/10%.

B Baselines

Here we thoroughly review the baselines used in this paper. Since we have two main tasks, one for traffic classification and the other one for explanation, we also categorize our baselines into two categories. For traffic classification, we have the following three baselines:

- **FS-Net** [19]: Recurrent neural networks (RNNs) to automatically extract representations from raw packet size sequences of encrypted traffic.
- **ET-Bert** [18]: Transformer-based traffic classifier where self-attention is applied on bytes to allow byte-byte interactions. Furthermore, they consider large-scale pre-training.
- **TFE-GNN** [41]: GNN-based deep learning model that firstly constructs byte-level graph for each packet, then applies GNNs to obtain each packet graph embedding and finally aggregates them together through the sequential-based model such as GRU.

For traffic explanation, we have following three baselines:

- **Random**: we randomly select the Top-K bytes or byte-byte interactions and treat them as the explanation.
- **Saliency Map** [33]: we calculate the gradient of the output prediction with respect to either each byte or each byte-byte interaction and then select the ones with top-K gradient magnitude as the explanation.
- **Self-Attention** [11]: we use the transformer attention as the importance score for each byte-byte interaction and select the top-K ones as the explanation.

Dataset	Type	# Train/Val/Test Seq.	# Packet	# Byte	Task	# Label
ISCX-VPN	Header Payload	1,231/154/157	34.20±15.23 24.62±18.99	1,351±612.4 2,828±2,742	App	6
ISCX-NonVPN	Header Payload	3,140/392/395	25.23±15.66 14.88±16.05	1,009±626.3 1,641±2,057	App	6
ISCX-Tor	Header Payload	1,354/169/174	43.03±14.92 40.94±17.02	1,721±596.6 6,093±2,543	App	8
ISCX-NonTor	Header Payload	19,179/2,398/2,400	26.00±16.77 13.60±17.06	1,040±670.7 1,740±2,411	App	8
IOS	Header Payload	776,156/97,803/97,803	/ /	25.74±2.740 28.27±29.93	Country	3
Android	Header Payload	1,086,909/135,691/135,692	/ /	25.66±2.670 28.84±31.04	Country	3

Table .1: Statistics of Traffic Flow Datasets. The first four are synthetic traffic flow datasets, while the last two are real-world user-generated traffic flow datasets. **We can clearly see that some traffic flow sequences consist of over a thousand bytes, which would incur a significantly large computational load when using a conventional transformer with quadratic time/space complexity.** This motivates us to design the hierarchical self-attention to overcome the computation issue.

Explainer	Budget	IOS				Android			
		Fid	Acc	C-Fid	C-Acc	Fid	Acc	C-Fid	C-Acc
Random	1%	39.63%	39.46%	2.60%	3.93%	18.12%	18.05%	1.18%	1.62%
	5%	40.71%	40.32%	2.88%	4.07%	24.43%	24.39%	1.81%	2.28%
	10%	50.62%	50.33%	3.79%	5.04%	31.38%	31.09%	2.20%	2.66%
Saliency Map	1%	72.51%	71.83%	15.91%	16.52%	75.42%	75.34%	13.43%	14.09%
	5%	92.64%	91.68%	39.97%	40.40%	90.17%	89.87%	22.30%	22.67%
	10%	95.44%	94.39%	46.64%	46.95%	94.45%	94.03%	25.75%	25.96%
Traffic-Explainer	1%	45.43%	44.47%	5.33%	5.70%	76.85%	76.13%	11.37%	11.46%
	5%	94.61%	92.93%	39.05%	38.96%	96.55%	95.73%	33.39%	33.26%
	10%	97.92%	96.43%	62.28%	62.34%	99.16%	98.57%	44.91%	44.80%

Table A.1: Traffic Explanation Performance for the Country Detection Task. Generally, both Traffic-Explainer and Saliency Map achieve high-quality explanations.

C Computation of Evaluation Metric

As our contributions involve both a scalable traffic classifier and a trustworthy traffic explainer, our evaluation metrics should also consider these two aspects: one for evaluating the traffic classification performance and the other for evaluating the explanation quality. We use Accuracy (Acc) and F1-macro to evaluate the traffic classification performance to avoid the bias caused by the imbalance of traffic flow instances. To evaluate the explanation quality, we first apply Traffic-Explainer to select the top-K important bytes, and then we recalculate the model predictions with the updated flow sequences by either removing or keeping those selected important bytes. For i^{th} instance, assuming $\mathcal{Y}^i$ represents its ground-truth label, $\widetilde{\mathcal{Y}}^i$ represents its predicted label with the original traffic flow sequence, $\widetilde{\mathcal{Y}}^{F,i}$ is its updated prediction after keeping only the Top-K important bytes and masking all others, and $\widetilde{\mathcal{Y}}^{CF,i}$ is its updated prediction after removing the Top-K important bytes and keeping all others. Then, the following four explanation evaluation metrics can be calculated as:

- **Fidelity (Fid)**: The percentage of updated predictions that equal the original model predictions after keeping only the Top-K important bytes.

$$\text{Fid} = \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}(\widetilde{\mathcal{Y}}^{F,i} = \widetilde{\mathcal{Y}}^{i})$$

- **Accuracy (Acc)**: The percentage of updated predictions that equal the ground-truth labels after keeping

Dataset	Metric	FS -Net	ET -Bert	TFE -GNN	Ours wo Packet	Ours wo Byte	Ours
ISCX-VPN	Accuracy	0.841	0.949	0.905	0.949	0.841	0.943
	F1-Macro	0.836	0.942	0.893	0.942	0.829	0.935
ISCX-nonVPN	Accuracy	0.704	0.901	0.856	0.901	0.848	0.896
	F1-Macro	0.681	0.916	0.861	0.916	0.857	0.907
ISCX-Tor	Accuracy	0.782	0.535	0.776	0.535	0.713	0.816
	F1-Macro	0.732	0.515	0.690	0.515	0.637	0.743
ISCX-nonTor	Accuracy	0.910	0.963	0.961	0.963	0.966	0.974
	F1-Macro	0.557	0.822	0.741	0.822	0.810	0.864
Average	Accuracy	0.809	0.837	<u>0.875</u>	0.837	0.842	**0.907**
	F1-Macro	0.702	<u>0.799</u>	0.796	<u>0.799</u>	0.783	**0.862**

Table D.1: Application 1 - Performance comparison among baselines. The best and runner-up results are in **bold** and <u>underlined</u>.

only the Top-K important bytes.

$$\text{Acc} = \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}(\widetilde{\mathcal{Y}}^{F,i} = \mathcal{Y}, \rangle)$$

- **Counterfactual-Fidelity (C-Fid)**: The percentage of updated predictions that equal the original model predictions after removing the Top-K important bytes.

$$\text{C-Fid} = \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}(\widetilde{\mathcal{Y}}^{CF,i} = \widetilde{\mathcal{Y}}^{i})$$

- **Counterfactual-Accuracy (C-Acc)**: The percentage of updated predictions that equal the ground-truth labels after removing the Top-K important bytes.

$$\text{C-Acc} = \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}(\widetilde{\mathcal{Y}}^{CF,i} = \mathcal{Y}^{i})$$

D Additional Results

D.1 Traffic Classification Performance

In addition to the superior explanation performance of our proposed Traffic-Explainer, our transformer-based traffic classifier also achieves superior classification performance. In Table D.1, ours achieves the best overall traffic classification performance in the task of Application 1. Moreover, to overcome the complexity of quadratic complexity of self-attention over bytes in the extensive, lengthy traffic sequences, our proposed hierarchical transformer, by first conducting byte-level self-attention within one packet, followed by the package-level self-attention within one flow, successfully maintains the optical performance as well as significantly reduce the computation complexity

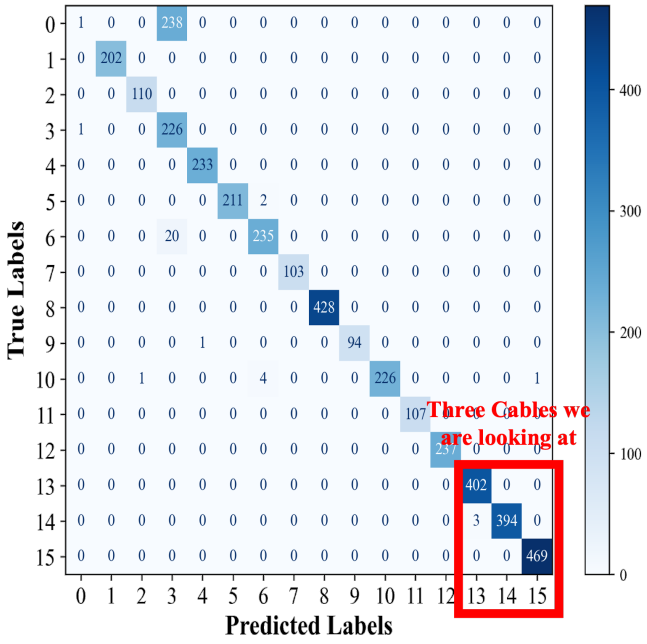


Figure D.1: Application 3 - The highlighted diagonal region in the classification confusion matrix effectively demonstrates the strong mapping capability of DL-powered network cartography in identifying associations based on RTT patterns. The last three groups of traceroutes are the ones we analyzed in Figure 7.2, including Palo Alto, US to Yokohama, Japan; Rockville, Maine, US to Amsterdam, Netherlands; and Vancouver, Canada to Washington, US.

D.2 Networking Cartography Performance

We collected approximately 5,000 traceroute measurements over a three-day period, covering 15 unique source-destination Paris traceroutes. To verify link homophily—where traceroutes utilizing the same cable exhibit similar RTT patterns—we labeled 5% of the traceroutes within each source-destination group and conducted a classification experiment. The results, visualized below, demonstrate high classification accuracy, confirming the correlation between cable usage and RTT values. Furthermore, the trained mapping model is applied to identify explanations in Figure D.1, offering insights that can inspire future research in deep learning-powered network cartography.