TrafCL: Robust Encrypted Malicious Traffic Detection via Contrastive Learning

Xiaodu Yang
Institute of Information Engineering, Chinese Academy of Sciences
School of Cyber Security, University of Chinese Academy of Sciences
Beijing, China
yangxiaodu@iie.ac.cn

Sijie Ruan*
School of Computer Science and Technology, Beijing Institute of Technology
Beijing, China
sjruan@bit.edu.cn

Jinyu Li
School of Computer Science and Technology, Beijing Institute of Technology
Beijing, China
1120201230@bit.edu.cn

Yinliang Yue
Zhongguancun Laboratory
Beijing, China
yueyl@zgclab.edu.cn

Bo Sun*
National Computer Network Emergency Response Technical Team
Beijing, China
sunbo@cert.org.cn

Abstract

Remote control malwares enable cyber attackers to achieve command and control over victim hosts, which are widely employed in ransomware attacks and espionage operations, jeopardizing personal privacy and state security. To effectively detect such malicious traffics holds high practical value. However, prior works have not adequately addressed the task due to challenges of encrypted traffics with misleading contents, incomplete sessions, and limited labels. To overcome these limitations, in this paper, we propose TrafCL, a contrastive learning framework for robust encrypted malicious traffic detection. In TrafCL, we first generate incomplete variants for the input session by *Session Augmentation*, then extract explicit session features with excluding misleading traffic contents by *Triple-aspect Session Feature Extraction*, and obtain session representations by *Co-attention Session Encoder* which fuses triple-aspect session features with capturing their interdependence. After that, we use a projection head to obtain final representations. TrafCL is pre-trained using unlabeled data to learn close representations for complete sessions and their incomplete variants, then fine-tuned on labeled data to detect encrypted malicious traffics. Experiment results show that TrafCL outperforms the best baseline by 11.35% and 6.71% in F1-scores on two datasets respectively.

CCS Concepts

• **Security and privacy** → **Network security**; • **Computing methodologies** → **Artificial intelligence**.

*Corresponding authors.

CIKM '24, October 21–25, 2024, Boise, ID, USA

ACM ISBN 979-8-4007-0436-9/24/10
https://doi.org/10.1145/3627673.3679839

Keywords

Encrypted Malicious Traffic Detection, Contrastive Learning

ACM Reference Format:
Xiaodu Yang, Sijie Ruan, Jinyu Li, Yinliang Yue, and Bo Sun. 2024. TrafCL: Robust Encrypted Malicious Traffic Detection via Contrastive Learning. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24), October 21–25, 2024, Boise, ID, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3627673.3679839

1 Introduction

Cyber attackers develop malware to fulfill their intrusive intentions, among which the most detrimental ones are those enabling attackers to gain remote control over victim users, such as remote access trojans [11]. Their attack workflow is illustrated in Fig. 1. Attackers first implant remote control malware on victim hosts via methods like phishing attacks, then realize command and control (C&C) on compromised victim hosts, where victims request and execute malicious tasks from attacker servers such as keyboard logging, screenshot, file transfer, etc., causing data exfiltration. These malwares are involved in not only individual-targeted ransomware attacks, but also state-sponsored espionage operations [16], threatening public privacy and national security. For instance, in 2022, the UAC-0041 group used FormBook to target the Ukrainian government for stealing sensitive data [33].

Moreover, traffic encryption has become the recent mainstream for secure data transmission over networks. Till 2024, over 95% of traffics across Google have been encrypted [15]. Nevertheless, it is double-edged since attackers also follow the trend, encrypting their malicious communication contents to conceal identities [4, 19, 43].

Considering the high-risk and stealthy nature of remote control malwares, it is of significant practical value to detect their encrypted traffics. However, it is non-trivial due to the following **challenges**, which have not been adequately addressed by previous methods:

- **Encrypted traffics with misleading contents**. Encrypted traffics contain multi-aspect information of payloads and interaction processes. Existing methods of encrypted traffic classification

either focus on a single aspect of information [25, 26, 45], or ignore the interdependence among multiple aspects [24]. Moreover, prior methods can be misled by disguised certificates [31, 32, 37] and uninformative payloads, e.g., random bytes or fast-expired sessions IDs [25, 45]. Because advanced remote control malware supports self-defined certificates [18], allowing attackers to disguise as legal servers with valid certificates applied from open platforms like *Let's Encrypt* [43].

- **Incomplete sessions**. Due to various reasons like software or device failures, the collected sessions are not always complete in the real world [24], especially at large-scale traffic gateways. In the most severe cases, nearly 50% packets in a session can not be captured. This causes information loss and impairs the performance of existing methods, as they are designed based on complete sessions. For example, ETBERT [25] models payloads of the first five packets in a session. However, the initial packets of complete and incomplete sessions may vary greatly, leading to significant discrepancies in the semantics of model inputs.
- **Limited labels**. Previous deep learning methods of encrypted malicious traffic detection usually require a large amount of labelled training data [25]. However, due to resource limitations and the rapid evolution of attack techniques, the availability of labelled malicious traffics is limited.

To tackle *the first challenge*, we find that even in encrypted traffics, there are still plaintext contents that can be utilized, i.e., payloads in the handshake stage which reflect the security parameters negotiated by two sides, and headers in the entire session which reveal the interaction processes between attackers and victims. To tackle *the second and the third challenges*, we find that though incomplete sessions may break the semantics of actual sessions, they are essentially subsets of complete sessions. Moreover, there are a vast volume of unlabeled encrypted traffics over networks, and it would be highly advantageous if we could transfer their knowledge to aid in detecting encrypted malicious traffics. Contrastive learning [3] is a recent unsupervised representation learning technique to pull similar samples closer and push dissimilar samples far apart in the representation space. Inspired by it, we regard a session and its incomplete variants to be similar, and can learn robust traffic representations insensitive to incomplete sessions using abundant unlabeled encrypted traffics.

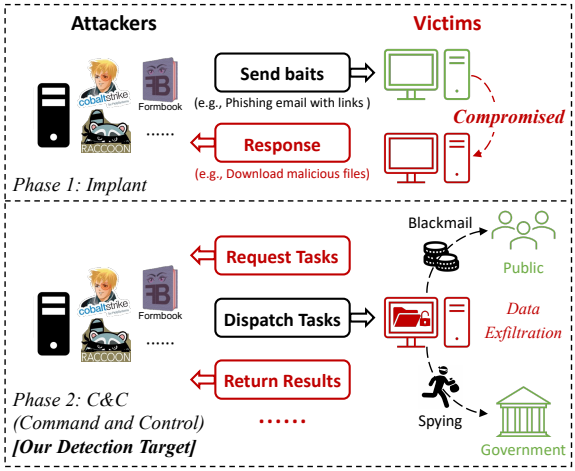


Figure 1: Attack Workflow of Remote Control Malware.

To this end, in this paper, we propose TrafCL, a contrastive learning framework for robust detection of encrypted malicious traffics. We pre-train the TrafCL framework on unlabeled data, guiding it to learn close representations for complete sessions and their incomplete variants, then detach and fine-tune the encoder using labeled data to detect encrypted malicious traffics where sessions can be incomplete. In TrafCL, we first generate incomplete variants for the input session by a *Session Augmentation* strategy. Then, we extract session features by *Triple-aspect Session Feature Extraction*, which explicitly characterize attackers and victims by their preferred security parameters and interaction patterns with excluding uninformative and disguisable contents. Next, we obtain session representations by a *Co-attention Session Encoder*, which fuses triple-aspect session features with capturing their interdependence. After that, we use a projection layer to obtain the final representations.

Our **contributions** are summarized below:

- We propose TrafCL, a contrastive learning framework for robust encrypted malicious traffic detection on incomplete sessions, which learns close representations for complete sessions and their incomplete variants by pre-training on unlabeled data.
- We propose a *Session Augmentation* strategy adapting to the nature of network traffic data, which generates incomplete variants as different views of a complete session, helping TrafCL learn robust session representations in a self-supervised way.
- We propose a *Co-attention Session Encoder* which captures the interdependence among triple-aspect session features by the alternating Co-attention mechanism.
- We evaluate the performance of TrafCL on two datasets. Experiment results show that TrafCL outperforms the best baseline by 11.35% and 6.71% in F1-scores respectively.

2 Preliminaries

2.1 Definitions

As shown in Fig 2, TLS-encrypted traffics exhibit a nested three-layer structure of TLS records, packets and sessions.

Definition 1 (TLS Record). *TLS record r is the unit of TLS traffics, which encapsulates one TLS message and is composed of two parts: (1) the header, containing attributes of the content type, the TLS version and the record length, which form a sequence and reflect the interaction process; and (2) the payload. For handshake messages like Client Hello, record payloads are plaintext and can be explicitly parsed into numeric, categorical or string fields, which reflect the negotiated security parameters and the identity information of two sides. For data transfer messages like Application Data, record payloads are unparsable ciphertext. Multiple TLS records can be encapsulated into a single packet to improve network transmission efficiency.*

Definition 2 (TLS Session). *A TLS session is actually a sequence of TLS records in chronological order, denoted as $S =< r_1, ..., r_n >$, which is split by the 5-tuple $< IP_{source}, IP_{destination}, Port_{source}, Port_{destination}, TLS\ protocol >$. It consists of two stages: the plaintext handshake for negotiating secure parameters, and the ciphertext data transfer for exchanging contents encrypted under the negotiated parameters [20].*

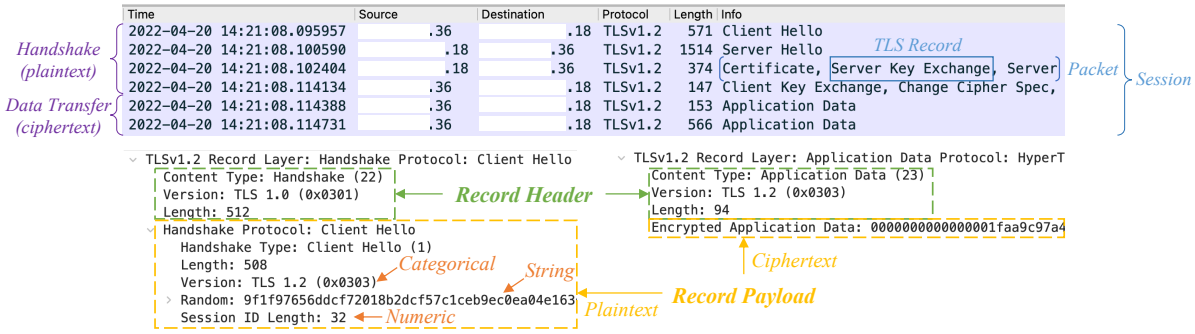


Figure 2: Preliminaries about TLS-encrypted Traffics.

2.2 Problem Formulation

To detect encrypted malicious traffic generated by various remote control malwares is actually a multi-classification problem. Given the input set $\mathcal{S}$ of TLS sessions and the label set $\mathcal{Y}$, our goal is to obtain an encrypted traffic classification model, predicting the label for each TLS session which can be incomplete.

3 Framework Overview

3.1 Model Overview

Insights. As stated in Sec. 1, contrastive learning is a promising technique that can be used to detect encrypted malicious traffics with incomplete sessions. Since a standard contrastive learning framework consists of a comprehensive input encoder and a rational input variant generation strategy, the following two key points should be carefully considered:

- **Fully exploit available semantics**: A session contains interdependent multi-aspect information, e.g., the negotiated security parameters, the identity attributes and the interaction processes (Sec. 2). Therefore, to model encrypted traffics comprehensively, we need to capture the multi-aspect information with considering their mutual dependence. Besides, disguisable certificates and some uninformative contents, like randoms and fast-expired sessions IDs, should be excluded to avoid misleading the model.
- **Simulate incomplete sessions**: To make the representations learned by contrastive learning robust on incomplete sessions, we should have two versions of a session, i.e., the incomplete version and its original complete version. However, it is impossible to obtain both versions of a session in the real world. For generating the input variant, we need to randomly drop some packets in the original session to simulate the incomplete sessions.

Main Idea. To realize the insights above, we propose TrafCL, a contrastive learning framework for robust encrypted malicious traffic detection, as shown in Fig. 3. TrafCL follows the classical dual-branch structure of contrastive learning methods, e.g., SimCLR[3] and MoCo[17]. We pre-train TrafCL with unlabeled data to enhance the robustness of encoder for incomplete sessions, then fine-tune it with labeled data to detect encrypted malicious traffics. First, given an input session S_i, it is fed into the *Session Augmentation* module to generate an incomplete variant S_j to simulate packet-loss scenarios (Sec. 4.1). Then, S_i and S_j go through the *Triple-aspect Session Feature Extraction* module to extract their features $X_i^{meta}, X_i^{id}, X_i^{seq}$ and $X_j^{meta}, X_j^{id}, X_j^{seq}$ based on the content structure of TLS-encrypted traffics (Sec. 4.2), where uninformative fields and disguisable certificates are excluded. Next, the triple-aspect features are fed into the *Co-attention Session Encoder* to obtain session embeddings h_i and h_j of S_i and S_j respectively, which are fused with interdependence being captured. The same encoder is shared in two branches (Sec. 4.3). Finally, the session embeddings h_i and h_j go through the *Projection Head*, mapped into lower-dimensional z_i and z_j as the final representations of sessions. The projection head is shared in two branches (Sec. 4.4).

3.2 Model Training

Following previous contrastive learning methods, we use the NT-Xent loss [3] $\mathcal{L}$ (Eq. 1) to pre-train our TrafCL framework, which aims to maximize the agreement between representations of similar positive samples and to minimize those of dissimilar negative samples. We randomly sample a batch of N sessions and define the contrastive task on the raw sessions and their derived augmentations in the batch, resulting in $2N$ data points. As for positive sample pairs, we use the raw S_i and its incomplete augmentation S_j, which are supposed to be similar in the latent space. As for negative samples, we use other $2(N-1)$ sessions in the batch. The loss function for a positive session pair (S_i, S_j) is defined as $\ell_{i,j}$, and the final loss $\mathcal{L}$ is computed across all positive pairs in the batch.

$$\begin{aligned} \operatorname{sim}(\boldsymbol{u}, \boldsymbol{v}) &= \frac{\boldsymbol{u}^{\top}\boldsymbol{v}}{\|\boldsymbol{u}\|\|\boldsymbol{v}\|} \\ \ell_{i,j} &= -\log \frac{\exp\left(\operatorname{sim}\left(\boldsymbol{z}_i, \boldsymbol{z}_j\right)/\tau\right)}{\sum_{k=1}^{2N} \mathbb{1}_{[k\neq i]} \exp\left(\operatorname{sim}\left(\boldsymbol{z}_i, \boldsymbol{z}_k\right)/\tau\right)} \\ \mathcal{L} &= \frac{1}{2N} \sum_{k=1}^{N} [\ell_{2k-1,2k} + \ell_{2k,2k-1}] \end{aligned} \tag{1}$$

Here, sim denotes the cosine similarity, τ is a temperature parameter and $\mathbb{1}_{[k\neq i]} \in \{0, 1\}$ is an indicator function [3]. Once pre-trained, the *Co-attention Session Encoder* will be detached and serve as an encoder to generate embeddings for sessions, which is then connected to a two-layer MLP and fine-tuned for detecting encrypted malicious traffics.

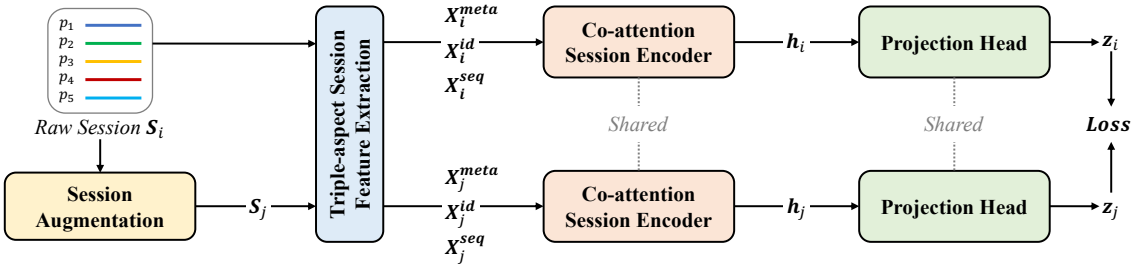


Figure 3: Framework of TrafCL.

4 Framework Details

4.1 Session Augmentation

Data augmentation generates variants of input samples, e.g., cropping images in CV tasks, so that the encoder can learn to capture the shared patterns of the raw input and its variants [3]. For network traffics, incomplete sessions are essentially subsets of the complete session and can be regarded as its different views. Inspired by it, we propose a session augmentation strategy as shown in Fig. 4, which assists the encoder in learning close representations for a complete session and its incomplete variants.

According to expertise, in large-scale network traffic gateways, the ratio of packets not being captured in sessions can range from 10% to 50%. Thus, given a session $S_i =< p_1, \cdots, p_n >$ where p_i is a packet and n is the packet number, we first randomly sample a packet loss ratio $r_{pl} \in [10\%, 50\%]$, then randomly drop r_{pl} packets in S_i to generate an incomplete variant $S_j =< p_{a_1}, ..., p_{a_{|S_j|}} >$, where $S_j \subset S_i$, $\mid S_j \mid = \lfloor (1 - r_{pl}) \cdot n \rfloor$ and $a_1, \cdots, a_{|S_j|}$ is a strictly increasing sequence. We use an independent and identically uniform distribution for the dropped probability of each packet. Under the same r_{pl}, we can obtain different incomplete variants for S_i.

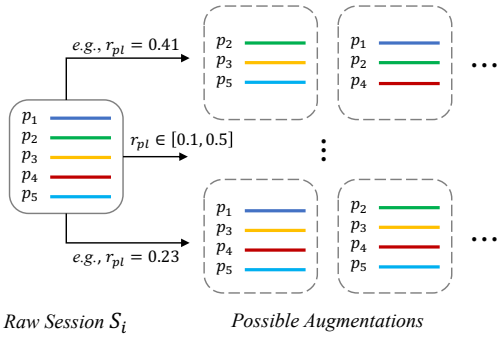


Figure 4: Session Augmentation.

4.2 Triple-aspect Session Feature Extraction

With the raw session S_i and its incomplete variant S_j, we next enrich them beyond payload bytes by extracting triple-aspect session features, i.e., *handshake meta features*, *handshake identity features* and *record sequential features*, which reflect attributes as well as interaction processes of two parties, and will be later fed into *Co-attention Session Encoder* to generate session representations.

Handshake Meta Features. As shown in Fig. 2, handshake record payloads are highly-structured plaintext, which can be directly parsed into semantically explicit fields. We consider four types of handshake messages for parsing, i.e., *Client Hello*, *Server Hello*, *Server Key Exchange* and *Client Key Exchange*. We discard uninformative messages like *Change Cipher Spec* and *Server Hello Done*, as well as messages about certificates, since some advanced remote control malware supports using legal certificates as disguises [18]. In *Client Hello* and *Server Hello*, we extract features of secure parameters that clients support and servers select, e.g., TLS versions, cipher suites, compression methods and extensions like application layer protocol negotiation, signature algorithms, supported groups, ec-point formats, etc. In *Server Key Exchange* and *Client Key Exchange*, we extract features of cryptographic parameters for the (Elliptic Curve) Diffie-Hellman algorithm, e.g., primes, generators, named curves, client versions of pre-master secret, etc.

The fields parsed from handshake record payloads are numerical, categorical or string-like. We concatenate numerical fields and one-hot encoded categorical fields as 716-dimensional *handshake meta features* $\mathbf{X}_{meta}$, which reflect the secure parameters chosen by victims and attackers.

Handshake identity features. Among the string fields parsed above, we select *Cookie* and *SNI* (Server Name Indication) as *handshake identity features* $\mathbf{X}_{id}$. Since advanced remote control malware, e.g., Cobalt Strike, can encode malicious contents into *Cookie* [28]. Moreover, attackers can dynamically retrieve the information of their C&C servers from legal platforms to conceal activities, e.g., Vidar storing its C&C server URLs on Telegram [7]. Therefore, the combinations of *Cookie* and *SNI* reveal the operational habits of attackers. Other string fields are not adopted, as they either expire fast, like session IDs and session tickets, or are randoms. In Sec. 6.5, we will study the impact of different combinations of string fields.

Record Sequential Features. We model the sequential interaction patterns at the TLS record level, unlike prior works at the packet level [24], since TLS records are finer-grained than packets and can better reveal the structure of encrypted traffics (Sec. 2.1). Record headers are informative plaintext and have the same structure in the entire session, forming a sequence that reflects the interaction process between victims and attackers. We extract header attributes from each record across the session, e.g., the record content type, the record handshake type (padding zero for *Application Data* records), the TLS version and the record length. We also extract features from lower layers which encapsulate TLS records, e.g., the lengths of TCP payload and packet. We further make length features directed by multiplying +1 (outgoing) or −1 (incoming). Finally, we denote header features of each TLS record as $\mathbf{hd}_i$, and *record sequential features* of the entire session as $\mathbf{X}_{seq} =< \mathbf{hd}_1, \cdots, \mathbf{hd}_r >$. Considering the tradeoff between information gain and model efficiency, we limit the length of record sequential features r to L_{seq}.

4.3 Co-attention Session Encoder

With the triple-aspect session features, we then generate session representations by *Co-attention Session Encoder* in Fig. 5. First, we embed *handshake meta features* $\mathbf{X}_{meta}$, *handshake identity features* $\mathbf{X}_{id}$ and *record sequential features* $\mathbf{X}_{seq}$ into $\mathbf{H}_{meta}$, $\mathbf{H}_{id}$ and $\mathbf{H}_{seq}$ through a fully-connected (FC) layer, a *handshake identity feature representation* layer and a *record sequential feature representation* layer respectively. $\mathbf{X}_{meta}$ after the FC layer is denoted as $\mathbf{H}_{meta} \in \mathbb{R}^{H}$. We detail the later two embedding processes and the alternating Co-attention mechanism to capture the mutual dependence among the triple parts as follows.

Handshake Identity Feature Representation. To embed $\mathbf{X}_{id}$ into $\mathbf{H}_{id}$, for each of F_{id} handshake identity features, we first tokenize it into a token sequence, where every two payload bytes are treated as a token and its decimal representation is regarded as the token ID (ranging from 0 to 65,535) following prior works [25]. Given both information gain and model efficiency, we retain the first L_{id} tokens for each identity feature. Then, we use an embedding layer to convert the token ID sequence into a dense vector sequence of dimension d. Next, we use a Transformer encoder [38] to model the token sequence, which directly calculates pairwise correlations and has shown clear advantages over RNN-based models. $\mathbf{X}_{id}$ after the Transformer encoder is denoted as $\mathbf{H}_{id} \in \mathbb{R}^{F_{id} \times L_{id} \times H_{id}}$.

Record Sequential Feature Representation. We adopt another Transformer encoder to embed *record sequential features* $\mathbf{X}_{seq}$. First, we input $\mathbf{X}_{seq}$ into a fully connected layer to fuse the header knowledge. Then, we feed the outputs to a positional encoding layer followed by another Transformer encoder. We denote the outputs of Transformer encoder as $\mathbf{H}_{seq} \in \mathbb{R}^{L_{seq} \times H_{seq}}$.

Alternating Co-attention. To capture the interdependence among triple-aspect session features, we generate the session representation $\mathbf{h}$ by the Co-attention mechanism, which integrates $\mathbf{H}_{meta}$, $\mathbf{H}_{id}$ and $\mathbf{H}_{seq}$ by jointly performing the *meta-seq-guided attention* on handshake identity feature representations and the *meta-id-guided attention* on record sequential feature representations.

Handshake identity features characterize the operation habits of attackers, which are tied to record sequential features that reflect interaction patterns of certain malware. Hence, to attend $\mathbf{H}_{id}$, we first calculate the pairwise importance between each identity feature token and each record header feature, and fuse the importance matrix into the process of attention calculation as shown in Eq. 2. The

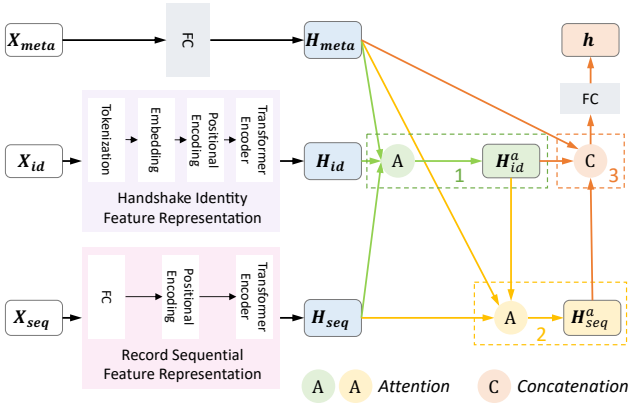


Figure 5: Co-attention Session Encoder.

pairwise importance matrix $\mathbf{Q} \in \mathbb{R}^{F_{id} \times L_{seq} \times L_{id}}$ is obtained though matrix multiplication, then the context matrix $\mathbf{C}_{id} \in \mathbb{R}^{F_{id} \times k \times L_{id}}$ is created by fusing different aspects of information and $\mathbf{Q}$. After that, we use $\mathbf{C}_{id}$ to produce the attention weights $\mathbf{a}_{id} \in \mathbb{R}^{L_{id}}$. Finally, $\mathbf{a}_{id}$ is applied to $\mathbf{H}_{id}$ to obtain the attended handshake identity feature representations $\mathbf{H}^a_{id} \in \mathbb{R}^{F_{id} \times H_{id}}$. Here, $\mathbf{W} \in \mathbb{R}^{H_{seq} \times H_{id}}$, $\mathbf{W}_{id} \in \mathbb{R}^{k \times H_{id}}$, $\mathbf{W}_{seq} \in \mathbb{R}^{k \times H_{seq}}$, $\mathbf{W}_{meta} \in \mathbb{R}^{k \times H}$ and $\mathbf{v} \in \mathbb{R}^{k}$ are learnable parameters.

$$
\begin{aligned}
\mathbf{Q} &= \mathbf{H}_{seq}(\mathbf{W}\mathbf{H}^{\top}_{id}) \\
\mathbf{C}_{id} &= \tanh(\mathbf{W}_{id}\mathbf{H}^{\top}_{id} + (\mathbf{W}_{seq}\mathbf{H}^{\top}_{seq})\mathbf{Q} + \mathbf{W}_{meta}\mathbf{H}_{meta}) \\
\mathbf{a}_{id} &= \mathrm{softmax}(\mathbf{v}^{\top}\mathbf{C}_{id}) \\
\mathbf{H}^a_{id} &= \sum_i a^i_{id}\mathbf{H}^i_{id}
\end{aligned}
\tag{2}
$$

Since the pairwise importance has been modeled in $\mathbf{H}^a_{id}$, to attend $\mathbf{H}_{seq}$, we simply use the classic additive attention in Eq. 3. We first create a context matrix $\mathbf{C}_{seq} \in \mathbb{R}^{k \times L_{seq}}$ by fusing different aspects of information, then use $\mathbf{C}_{seq}$ to produce the attention weights $\mathbf{a}_{seq} \in \mathbb{R}^{L_{seq}}$. Finally, $\mathbf{a}_{seq}$ is applied to $\mathbf{H}_{seq}$ to obtain the attended record sequential feature representations $\mathbf{H}^a_{seq} \in \mathbb{R}^{H_{seq}}$. Here, $\mathbf{W}'_{seq} \in \mathbb{R}^{k \times H_{seq}}$, $\mathbf{W}'_{id} \in \mathbb{R}^{k \times H_{id}F_{id}}$, $\mathbf{W}'_{meta} \in \mathbb{R}^{k \times H}$ and $\mathbf{w} \in \mathbb{R}^{k}$ are learnable parameters, $\mathrm{flat}(\cdot)$ is the matrix flatten operation.

$$
\begin{aligned}
\mathbf{C}_{seq} &= \tanh(\mathbf{W}'_{seq}\mathbf{H}^{\top}_{seq} + \mathbf{W}'_{id}\mathrm{flat}(\mathbf{H}^a_{id})^{\top} + \mathbf{W}'_{meta}\mathbf{H}_{meta}) \\
\mathbf{a}_{seq} &= \mathrm{softmax}(\mathbf{w}^{\top}\mathbf{C}_{seq}) \\
\mathbf{H}^a_{seq} &= \sum_i a^i_{seq}\mathbf{H}^i_{seq}
\end{aligned}
\tag{3}
$$

Finally, we fuse the attended $\mathbf{H}^a_{id}$ and $\mathbf{H}^a_{seq}$ with $\mathbf{H}_{meta}$ to obtain the final session representations. We first flatten $\mathbf{H}^a_{id}$, and employ the concatenation-based fusion strategy to combine the triple-aspect information, then use a fully connected (FC) layer to transform them into the output $\mathbf{h}$ as the final session representation.

$$
\mathbf{h} = \mathrm{FC}([\mathbf{H}_{meta}; \mathrm{flat}(\mathbf{H}^a_{id}); \mathbf{H}^a_{seq}]) \tag{4}
$$

4.4 Projection Head

We transform the session representation $\mathbf{h}$ into the branch output $\mathbf{z}$ via a projection head as defined in Eq. 5, which is a feed-forward network, shared by the two branches of TrafCL framework. Prior studies of contrastive learning have proven that introducing such projection operation improves the overall performance [3].

$$
\mathbf{z} = \mathrm{FFN}(\mathbf{h}) \tag{5}
$$

where $FFN(\cdot)$ is composed of two fully connected layers interleaved with a ReLU activation function. The output $\mathbf{z}$ will be used to calculate the NT-Xent loss defined in Eq. 1 during pre-training.

5 Downstream Classification

Having pre-trained *Co-attention Session Encoder* via the TrafCL framework, we then fine-tune it using labelled datasets by connecting the encoder to a fully connected (FC) layer followed by a

sigmoid activation function σ which normalizes the output into $[0, 1]$, transforming the session representation into the output as defined below:

$$\hat{y} = \sigma(\mathrm{FC}(\mathbf{h})) \tag{6}$$

Given that to detect encrypted malicious traffics is a multi-class classification task, during fine-tuning, the classic cross-entropy loss is adopted to train the model:

$$\mathcal{L}(\boldsymbol{\theta}) = -\sum_{s_k \in \mathcal{S}} \sum_{k=1}^{K} y_k \log(\hat{y}_k) \tag{7}$$

where $\boldsymbol{\theta}$ denotes all learnerable parameters in the model, $\mathcal{S}$ is the set of sessions used for training, k is the number of classes, and y_k is the ground-truth of session s_k.

6 Experiments

6.1 Datasets

Pre-training Dataset. Despite the scarcity of labeled data for encrypted malicious traffics, there are abundant unlabeled encrypted traffics available on networks which can be leveraged to learn robust session representations. To facilitate reproducibility, we pre-train TrafCL using encrypted traffics from the public dataset CIC-DoHBrw-2020 [30] with neglecting labels.

Evaluation Datasets. As detailed in Table 1, we use two datasets from different network environments for evaluation, i.e., the private RAT-PN dataset and the public USTC-VPN dataset.

- RAT-PN, which is collected from real-world scenarios and contains incomplete sessions. Its malicious encrypted traffics come from six representative remote control malwares which are currently prevalent, provided by cybersecurity companies. Its benign encrypted traffics are collected from a company over two months, where the employees were aware of the data collection and carried out diverse normal network activities randomly.
- USTC-VPN, which comes from the widely-adopted public datasets USTC-TFC [39] and ISCX-VPN [10], consisting of complete sessions. USTC-TFC originally comprises ten types of benign traffics and ten types of malicious traffics, in which we retain only the encrypted traffics, resulting in four types of malicious traffics and hardly any benign traffics remaining. Thus, we supplement the dataset with benign encrypted traffics from ISCX-VPN, following prior works [9]. To simulate the scenarios where the captured sessions are incomplete, we randomly drop packets for sessions in USTC-VPN at ratios ranging from 10% to 50%.

Fig. 6 illustrates the existing ratios of common TLS messages (encapsulated in packets) in our datasets, with complete sessions sourced from the CIC-DoHBrw-2020 dataset and incomplete sessions from the real-world RAT-PN dataset.

6.2 Experimental Settings

Evaluation Metrics. We use the macro averages of four typical metrics for performance evaluation, i.e., Accuracy (AC), Precision (PR), Recall (RC), and F1-score.

Baselines. We compare our TrafCL with the following state-of-the-art baselines of encrypted traffic classification, including:

Table 1: Details of Evaluation Datasets.

Dataset	Traffic Type	#. Session
RAT-PN	Benign	10,418
	Cobalt Strike	1,076
	Sliver	508
	Vidar	182
	AgentTesla	83
	FormBook	55
	Raccoon	52
USTC-VPN	Benign	2,042
	Htbot	1,110
	Neris	159
	Shifu	356
	Virut	450

- **Statistical methods**: (1) FlowPrint [37], which extracts features from devices, certificates, packet sizes and timestamps, then clusters to build fingerprints, and classifies traffics based on cross-correlation. and (2) BSRF [44], which extracts statistical features of packet size and time, then trains balanced stacked Random Forest for classification;
- **Single-modal deep learning methods**: (1) FS-Net [26], which represents packet length sequences based on the encoder-decoder structure and stacked bi-GRUs, (2) EC-GCN [8], which builds graphs from packet length intervals and models them by GCN, (3) TSCRNN [22], which represents payload bytes by CNN and bi-LSTM, (4) ETBERT [25], which represents stitched packet payloads by pre-training Transformers, and (5) TFE-GNN [45], which builds byte-level graphs of packet headers and payloads, then represents them based on GNN;
- **Multi-modal deep learning method**: PEAN [24], which models payload bytes by pre-training Transformers and packet length sequences by LSTM, then concatenates the two parts of representations for classification.

Note that, ETBERT and PEAN are first pre-trained on unlabelled data, and then fine-tuned on labeled data as our TrafCL, while other baselines are trained from scratch.

Variants. We compare TrafCL with its six variants. *TrafCL-noMeta* (without handshake meta features), *TrafCL-noID* (without handshake identity features) and *TrafCL-noRec* (without record sequential features) exclude one aspect of session features and concatenate embeddings of the left two aspects as session representations. *TrafCL-noCoAt* fuses the triple feature embeddings by direct concatenation instead of the Co-attention mechanism. *TrafCL-noCL* directly trains *Co-attention Session Encoder* on labeled datasets without contrastive-learning-based pre-training. *TrafCL-aug-aug* builds

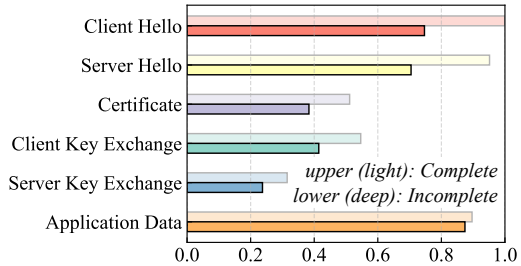


Figure 6: Existing Ratios of TLS Messages in Our Datasets.

Table 2: Performance Comparison with Baselines.

Dataset	RAT-PN				USTC-VPN			
Baseline	**AC**	**PR**	**RC**	**F1**	**AC**	**PR**	**RC**	**F1**
FlowPrint [37]	0.8680	0.6815	0.5002	0.5400	0.9291	0.8777	0.8896	0.8541
BSRF [44]	0.8627	0.7088	0.4933	0.5182	0.9289	0.8515	0.8118	0.8241
FS-Net [26]	0.8592	0.4000	0.2874	0.2731	0.5222	0.5222	0.3203	0.3031
EC-GCN [8]	0.8136	0.7213	0.8136	0.7483	0.8740	0.8411	0.8740	0.8564
TSCRNN [22]	0.7770	0.6038	0.7770	0.6795	0.5801	0.4405	0.5801	0.4965
TFE-GNN [45]	0.8397	0.4630	0.3828	0.3393	0.9291	0.9019	0.7792	0.7715
ETBERT [25]	0.8307	0.2560	0.3493	0.2589	0.9554	0.9537	0.8521	0.8677
PEAN [24]	0.7770	0.1110	0.1429	0.1249	0.2913	0.0583	0.2000	0.0902
TrafCL (Ours)	**0.9433**	**0.9014**	**0.8528**	**0.8618**	**0.9816**	**0.9463**	**0.9277**	**0.9348**

positive pairs with two different augmentations, instead of the raw session and its one augmentation.

Training Details & Hyperparameters. Each evaluation dataset is split by 8:1:1 for training, validation and test. For both pre-training and fine-tuning, the batch size N is 128 and the learning rate is $3e$-4. We fine-tune with Adam [21] for 10 epochs, where $\beta_1 = 0.9$, $\beta_2 = 0.999$, the dropout rate is set to 0.1. The training is stopped once the validation loss no longer decreases. L_{id} and L_{seq} are set to 256 and 20 respectively. All transformer encoders contain one layer. In handshake identity feature representation, the dense sub-layer contains 4 neurons, the token is embedded into 16 dimensions, and the hidden dimension for the attention calculation is 16. In record sequential feature representation, the dense sub-layer and the FC layer contain 768 neurons. Hyperparameters are set by grid search on validation set, e.g., the hidden dimension of attention is searched in $\{8, 16, 32\}$, L_{id} in $\{64, 128, 256\}$, and L_{seq} in $\{10, 20, 50\}$.

Implementation. Our method is implemented with PyTorch. Experiments are conducted on a workstation with an Intel(R) Core(TM) CPU i7-8700K @ 3.7GHz, 32GB memory, Nvidia 1080ti GPU, and Windows 10 OS.

6.3 Overall Performance Evaluation

We compare TrafCL with eight baselines of encrypted traffic classification on two datasets, of which the results are detailed in Table 2. Generally, TrafCL outperforms the best baselines by 11.35% and 6.71% in F1-scores on datasets RAT-PN and USTC-VPN respectively. Notably, the RAT-PN dataset comes from more advanced remote control malwares (Sec. 6.1) that are harder to detect, while the USTC-VPN dataset was collected in 2016, of which the attack techniques are comparatively out-dated and easier to be exposed.

Previous methods are designed based on complete sessions and do not perform well when input sessions are incomplete. Specifically, FS-Net directly models the raw packet length sequences and shows F1-scores 58.87% and 63.17% lower than TrafCL on datasets RAT-PN and USTC-VPN, as the sequential information is corrupted for incomplete sessions. FlowPrint, BSRF and EC-GCN show F1-scores 26.69%, 24.51% and 47.52% higher than FS-Net on RAT-PN, as well as 55.10%, 52.10% and 55.33% higher on USTC-VPN respectively. Since they extract coarser features, e.g., session-level statistical features and graphs built from packet length intervals, which are less fluctuated than raw sequences for incomplete sessions. However, the distributions of these features for incomplete sessions can still be biased greatly compared to those for complete sessions.

Payload-based methods TSCRNN, ETBERT and TFE-GNN show F1-scores 18.23%, 60.29% and 52.25% lower than TrafCL on RAT-PN, as well as 43.83%, 6.71% and 16.33% lower on USTC-VPN, which has two aspects of reasons. First, they model payloads of the initial packets in sessions, of which the semantics may appear significant variations for incomplete sessions. Second, they learn superficial semantics of raw payload bytes, and can be misled by uninformative or disguised payload contents like certificates. The multi-modal method PEAN performs the worst among baselines, showing F1-scores 73.69% and 84.46% lower than our TrafCL. Though PEAN incorporates more information than single-modal methods, it can introduce additional disturbances for incomplete sessions.

It is worth noting that, ETBERT and PEAN also adopt the paradigm of pre-training and fine-tuning. However, their pre-training tasks are designed based on complete sessions, focusing on capturing the contextual relationships between adjacent packets. In cases where sessions are incomplete, such contextual information will be severely disrupted, leading to poor performance.

Differently, our TrafCL enhances the robustness of learned traffic representations for incomplete sessions through a contrastive learning framework with designing augmentation strategies. Moreover, we thoroughly model the multi-aspect information of encrypted traffics with excluding misleading contents by *Triple-aspect Session Feature Extraction*, and fuse the features with capturing their interdependence by the *Co-attention Session Encoder*.

6.4 Ablation Study

In this section, we verify the effectiveness and necessity of each module in TrafCL, of which the results are shown in Table 3, where the upper part compares encoder components and the lower part concerns the contrastive learning strategies.

Impact of Encoder Components. Comparing TrafCL with TrafCL-noMeta, TrafCL-noRec, TrafCL-noID and TrafCL-noCoAt, we find that: (1) The triple-aspect session features all contribute to the overall performance, while handshake meta features and record sequential features play a more important role. Since removing any of the three parts leads to dropped performance, and it is more evident for TrafCL-noMeta and TrafCL-noRec of which F1-scores drops 26.03% and 38.58% on RAT-PN, as well as 13.34% and 12.87% on USTC-VPN. (2) The Co-attention mechanism better captures the

Table 3: Ablation Study of TrafCL.

Dataset							RAT-PN				USTC-VPN			
Variant	**M**	**I**	**R**	**Co**	**CL**	**r-a**	**AC**	**PR**	**RC**	**F1**	**AC**	**PR**	**RC**	**F1**
TrafCL-noMeta	×	✓	✓	×	✓	✓	0.8807	0.5796	0.6419	0.6015	0.9134	0.8377	0.7877	0.8014
TrafCL-noID	✓	×	✓	×	✓	✓	0.9314	0.9125	0.7973	0.8264	0.9764	0.9356	0.9100	0.9189
TrafCL-noRec	✓	✓	×	×	✓	✓	0.8501	0.6127	0.4665	0.4760	0.8451	0.8229	0.8321	0.8061
TrafCL-noCoAt	✓	✓	✓	×	✓	✓	0.9336	0.9179	0.8260	0.8405	0.9764	0.9452	0.9143	0.9278
TrafCL-noCL	✓	✓	✓	✓	×	×	0.9210	0.8899	0.7624	0.7765	0.9685	0.8998	0.8937	0.8947
TrafCL-aug-aug	✓	✓	✓	✓	✓	×	0.9351	0.9141	0.8307	0.8436	0.9685	0.9170	0.8910	0.9009
TrafCL	✓	✓	✓	✓	✓	✓	**0.9433**	**0.9014**	**0.8528**	**0.8618**	**0.9816**	**0.9463**	**0.9277**	**0.9348**

[1] 'M': handshake meta features, 'I': handshake identity features, 'R': record sequential features, 'Co': the Co-Attention mechanism.
[2] 'CL': the contrastive learning framework, 'r-a': the strategy of constructing positive pairs with the raw session and its one augmentation.

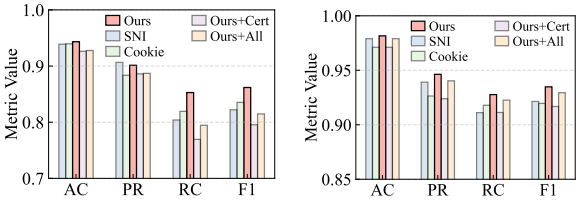
interdependence among triple-aspect features than simple concatenation, of which F1-scores drop 2.13% and 0.7% on RAT-PN and USTC-VPN.

Impact of Contrastive Learning Framework. Compared with TrafCL-noCL, we find that: (1) The contrastive-learning-based pre-training improves our performance on incomplete sessions. Without pre-training, TrafCL-noCL shows F1-scores 8.53% and 4.01% lower than TrafCL on RAT-PN and USTC-VPN. (2) Despite it, TrafCL-noCL still surpasses the best baseline with 2.82% and 2.7% higher F1-scores on RAT-PN and USTC-VPN, proving the effectiveness of our triple-aspect session feature extraction and Co-attention session encoder.

Impact of Positive Pair Selection. As for positive pairs, using the raw session and its one augmentation (i.e., our TrafCL) performs better than using two different augmentations of the raw session (i.e., TrafCL-aug-aug). Since augmented sessions are subsets of the raw session, and they are more inclined to achieve close representations. While two augmentations may have few overlaps, making their representations harder to converge.

6.5 Handshake Identity Feature Analysis

In Sec. 4.2, we select *Cookie* and *SNI* from the string fields parsed from handshake payloads as handshake identity features, since other string fields contain two types of disturbances: (1) uninformative contents, like randoms and fast-expired Session IDs, and (2) disguisable contents, like certificates. In Fig. 7, we compare different combinations of string fields parsed from handshake payloads to verify the superiority of our choice. Specifically, the combination of Cookie and SNI ('Ours') performs the best. Using only 'Cookie' or 'SNI' causes performance decreases, while adding certificates on top of Cookie and SNI ('Ours+Cert') performs even worse than without certificates, especially on the RAT-PN dataset in which recent remote control malwares can adopt disguised certificates. Using all string fields ('Ours+All'), including randoms, keys, certificates, etc., does not improve the performance and even performs worse than using only 'Cookie' or 'SNI' on the RAT-PN dataset. The comparison among the combinations of 'Ours', 'Ours+Cert' and 'Ours+All' proves the necessity of excluding uninformative and disguisable string fields when selecting handshake identity features.

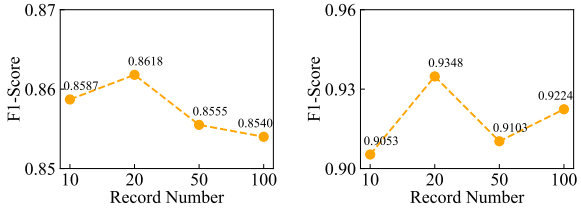


(a) Dataset: RAT-PN (b) Dataset: USTC-VPN

Figure 7: Handshake Identity Feature Analysis.

6.6 Sensitivity Analysis

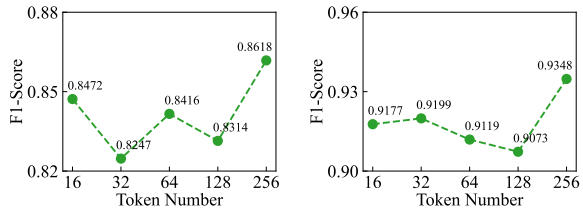
Impact of Record Sequential Feature Length L_{seq}. It can be seen from Fig. 8 that, including more TLS records do not necessarily result in better detection performance. The model achieves the best performance when the record number is at most 20 on both datasets. Using less than 20 records can cause information loss for most sessions, while using more records may lower the computational efficiency and include redundant information from ciphertext *Application Data*, making the model performance decrease.



(a) Dataset: RAT-PN (b) Dataset: USTC-VPN

Figure 8: Impact of Record Sequential Feature Length.

Impact of Handshake Identity Feature Length L_{id}. Following previous works [29], we vary the number of tokens from 16 to 256 to evaluate the impact of handshake identity feature length. From Fig. 9 we can observe that, the best performance is achieved with the handshake identity feature length equal to 256. On the one hand, longer identity features can cover more information of string fields and thus improve the discriminative ability of model. When the number of tokens is increased to 256, the model performance shows a relatively evident improvement. On the other hand, using more than 256 tokens will further increase the computing complexity. Considering such trade-off, we set the handshake identity feature length to 256.



(a) Dataset: RAT-PN (b) Dataset: USTC-VPN

Figure 9: Impact of Handshake Identity Feature Length.

7 Related Work

7.1 Encrypted Traffic Classification

To detect encrypted malicious traffics belongs to the general field of encrypted traffic classification, of which existing methods can be categorized into three types: (1) *Rule-based methods* expire fast, which build blocklists of IPs [2], certificates [32], or port numbers [12] for identifying attackers. (2) *Statistical methods* usually extract statistical features of packet size and time, then classify using machine learning algorithms, e.g., AppScanner [36] and BSRF [44]. FlowPrint [37] uses more information of devices and certificates to build fingerprints by clustering. IARF [31] further extracts handshake features like certificate validation results. However, statistical features can be greatly biased for incomplete sessions, and handshake features can be deceptive due to disguised certificates. (3) *Deep learning methods* are state-of-the-art in this task, which are single-modal or multi-modal. Some single-modal deep learning methods only input packet time and length, e.g., Deep Fingerprinting [35] and FS-Net [26] directly representing the raw sequences, GraphDApp [34], TrafficGCN [42] and EC-GCN [8] transforming the sequences into more intricate graphs. Other single-modal methods only input payloads, e.g., EETC [27] transforming payloads into images, TSCRNN [22] extracting spatio-temporal features from payloads by CNN and bi-LSTM, TFE-GNN [45] encoding byte-level packet graphs by GNN. ETBERT [25] also adopts the pre-training and fine-tuning paradigm as us, which pre-trains Transformers to represent the first 512 payload tokens of sessions. Multi-modal methods integrate the two aspects of information. They usually model payloads based on CNN, combined with statistical features about packet time and length [6, 23]. PEAN [24] represents payloads by pre-training Transformers and packet length sequences by LSTM, then concatenates the two parts.

However, existing methods are all designed based on complete sessions. When it comes to incomplete sessions, the sequential information of packet time and size can be disrupted, and payload semantics of initial packets can vary greatly. Such disturbances can be even additional for multi-modal methods, causing their performance worse than single-modal methods. Moreover, prior multi-modal methods ignore the interdependence among the multi-aspect information, and payload-dependent methods (either single-modal or multi-modal) can be misled by uninformative or disguised contents in encrypted malicious traffics.

Differently, our TrafCL learns robust representations for incomplete sessions via contrastive learning with specific session augmentations. Furthermore, to model encrypted traffics comprehensively, we fuse triple-aspect session features by Co-attention to capture their mutual dependence, exclude uninformative and disguisable contents to avoid misleading the model.

7.2 Contrastive Learning

Recently, contrastive learning has shown promising performance in CV [3, 17, 40] and NLP [5, 13, 14] tasks. However, common data augmentation policies, e.g., rotation and color jitter for images, do not adapt to the nature of network traffic data, and few studies have introduced contrastive learning into encrypted malicious traffic detection. Though PacRep [29] adopts contrastive learning for traffic analysis, it focuses on multi-label packet-level classification where positive pairs are packets sharing the same labels across all tasks and negative pairs are those with different labels, which differs from our task. NetAugment [1] and Rosetta [41] are both single-modal and only model packet time and length. NetAugment proposes Tor-tailored augmentations on traffic bursts for website fingerprinting, which modify incoming burst sizes, insert outgoing bursts and merge incoming bursts. However, such augmentations could substantially distort the semantics and the labels of traffics in our task. Similarly, Rosetta also augments traffics by merging bursts, plus shifting and duplicating subsequences in traffics. These method are still based on the premise of complete sessions, similar with prior methods discussed in Sec. 7.1.

To adapt contrastive learning for encrypted malicious traffic detection robust to incomplete sessions, our TrafCL introduces a session augmentation strategy to help the encoder learn close representations for complete sessions and their incomplete variants, and a Co-attention session encoder with triple-aspect session features extracted to capture the multi-modal information in traffics comprehensively.

8 Conclusion

In this paper, we propose TrafCL, a contrastive learning framework for robust encrypted malicious traffic detection, overcoming the challenges of encrypted traffics with misleading contents, incomplete sessions and limited labels. TrafCL is pre-trained on unlabeled data to learn close representations for complete sessions and their incomplete variants, then fine-tuned on labeled data to detect encrypted malicious traffics which can be incomplete. In TrafCL, we first generate incomplete variants for input sessions by the *Session Augmentation* module, then extract features from them with excluding uninformative and disguisable contents by the *Triple-aspect Session Feature Extraction* module, next fuse the triple-aspect features with capturing their interdependence to generate session representations by the *Co-attention Session Encoder*, and obtain the final representations by the *Projection Head*. Experiments on two datasets prove the effectiveness of our TrafCL framework, which outperforms the best baseline by 11.35% and 6.71% respectively in F1-scores. In future work, we will investigate leveraging generative models to construct more samples, in order to enhance the robustness of our model in detecting evolving remote control malwares.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 62202462) and Beijing Institute of Technology Research Fund Program for Young Scholars (No. 6120220113).

References

[1] Alireza Bahramali, Ardavan Bozorgi, and Amir Houmansadr. 2023. Realistic Website Fingerprinting By Augmenting Network Traces. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*. 1035–1049.

[2] Gabriel Bastos, Artur Marzano, Osvaldo Fonseca, Elverton Fazzion, Cristine Hoepers, Klaus Steding-Jessen, Ítalo Cunha, Dorgival Guedes, and Wagner Meira. 2019. Identifying and Characterizing bashlite and mirai C&C servers. In *2019 IEEE Symposium on Computers and Communications (ISCC)*. IEEE, 1–6.

[3] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*. PMLR, 1597–1607.

[4] Cisco. 2021. Cisco Encrypted Traffic Analytics White Paper. https://www.cisco.com/c/en/us/solutions/collateral/enterprise-networks/enterprise-network-security/nb-09-encrytd-traf-anlytcs-wp-cte-en.html.

[5] Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555* (2020).

[6] Jianbang Dai, Xiaolong Xu, and Fu Xiao. 2023. GLADS: A Global-Local Attention Data Selection Model for Multimodal Multitask Encrypted Traffic Classification of IoT. *Computer Networks* (2023), 109652.

[7] Darktrace. 2023. A Surge of Vidar: Network-based details of a prolific info-stealer. https://darktrace.com/blog/a-surge-of-vidar-network-based-details-of-a-prolific-info-stealer.

[8] Zulong Diao, Gaogang Xie, Xin Wang, Rui Ren, Xuying Meng, Guangxing Zhang, Kun Xie, and Mingyu Qiao. 2023. EC-GCN: A encrypted traffic classification framework based on multi-scale graph convolution networks. *Computer Networks* 224 (2023), 109614.

[9] Cong Dong, Zhigang Lu, Zelin Cui, Baoxu Liu, and Kai Chen. 2021. MBTree: detecting encryption RATs communication using malicious behavior tree. *IEEE Transactions on Information Forensics and Security* 16 (2021), 3589–3603.

[10] Gerard Draper-Gil, Arash Habibi Lashkari, Mohammad Saiful Islam Mamun, and Ali A Ghorbani. 2016. Characterization of encrypted and vpn traffic using time-related. In *Proceedings of the 2nd international conference on information systems security and privacy (ICISSP)*. 407–414.

[11] Zhuoqun Fu, Mingxuan Liu, Yue Qin, Jia Zhang, Yuan Zou, Qilei Yin, Qi Li, and Haixin Duan. 2022. Encrypted Malware Traffic Detection via Graph-based Network Analysis. In *Proceedings of the 25th International Symposium on Research in Attacks, Intrusions and Defenses*. 495–509.

[12] Recorded Future. 2021. Report: Full-Spectrum Cobalt Strike Detection. https://go.recordedfuture.com/hubfs/reports/mtp-2021-0914.pdf.

[13] Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple Contrastive Learning of Sentence Embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. 6894–6910.

[14] John Giorgi, Osvald Nitski, Bo Wang, and Gary Bader. 2021. DeCLUTR: Deep Contrastive Learning for Unsupervised Textual Representations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. 879–895.

[15] Google. 2024. Google Transparency Report: HTTPS encryption on the web. https://transparencyreport.google.com/https/overview.

[16] Group-IB. 2023. Cyber threats wrapped: rampant ransomware, inglorious initial access brokers, sneaky stealers top threat trends to watch. https://www.group-ib.com/media-center/press-releases/hi-tech-crime-trends-2022-2023/.

[17] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 9729–9738.

[18] HelpSystems. 2022. Cobalt Strike: Malleable Command and Control. https://trial.cobaltstrike.com/help-malleable-c2.

[19] HelpSystems. 2023. Cobalt Strike: HTTP Beacon and HTTPS Beacon. https://hstechdocs.helpsystems.com/manuals/cobaltstrike.

[20] Internet Engineering Task Force (IETF). 2008. RFC 5246: The Transport Layer Security (TLS) Protocol Version 1.2. https://www.rfc-editor.org/rfc/rfc5246.

[21] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[22] Kunda Lin, Xiaolong Xu, and Honghao Gao. 2021. TSCRNN: A novel classification scheme of encrypted traffic based on flow spatiotemporal features for efficient management of IIoT. *Computer Networks* 190 (2021), 107974.

[23] Kunda Lin, Xiaolong Xu, and Fu Xiao. 2022. MFFusion: A multi-level features fusion model for malicious traffic detection based on deep learning. *Computer Networks* 202 (2022), 108658.

[24] Peng Lin, Kejiang Ye, Yishen Hu, Yanying Lin, and Cheng-Zhong Xu. 2022. A Novel Multimodal Deep Learning Framework for Encrypted Traffic Classification. *IEEE/ACM Transactions on Networking* (2022).

[25] Xinjie Lin, Gang Xiong, Gaopeng Gou, Zhen Li, Junzheng Shi, and Jing Yu. 2022. ET-BERT: A Contextualized Datagram Representation with Pre-training Transformers for Encrypted Traffic Classification. In *Proceedings of the ACM Web Conference 2022*. 633–642.

[26] Chang Liu, Longtao He, Gang Xiong, Zigang Cao, and Zhen Li. 2019. FS-Net: A flow sequence network for encrypted traffic classification. In *IEEE INFOCOM 2019-IEEE Conference On Computer Communications*. IEEE, 1171–1179.

[27] Xiuli Ma, Wenbin Zhu, Jieling Wei, Yanliang Jin, Dongsheng Gu, and Rui Wang. 2023. EETC: An extended encrypted traffic classification algorithm based on variant resnet network. *Computers & Security* 128 (2023), 103175.

[28] Durgesh Sangvikar Yanhui Jia Yu Fu Matthew Tennis, Chris Navarrete and Siddhart Shibiraj. 2023. Cobalt Strike Attack Detection Defense Technology Overview. https://live.paloaltonetworks.com/t5/blogs/cobalt-strike-attack-detection-amp-defense-technology-overview/ba-p/533753.

[29] Xuying Meng, Yequan Wang, Runxin Ma, Haitong Luo, Xiang Li, and Yujun Zhang. 2022. Packet representation learning for traffic classification. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 3546–3554.

[30] Mohammadreza MontazeriShatoori, Logan Davidson, Gurdip Kaur, and Arash Habibi Lashkari. 2020. Detection of doh tunnels using time-series classification of encrypted traffic. In *2020 IEEE Intl Conf on Dependable, Autonomic and Secure Computing (DASC)*. IEEE, 63–70.

[31] Zequn Niu, Jingfeng Xue, Dacheng Qu, Yong Wang, Jun Zheng, and Hongfei Zhu. 2022. A novel approach based on adaptive online analysis of encrypted traffic for identifying Malware in IIoT. *Information Sciences* 601 (2022), 162–174.

[32] Carlos Novo and Ricardo Morla. 2020. Flow-based detection and proxy-based evasion of encrypted malware C2 traffic. In *Proceedings of the 13th ACM Workshop on Artificial Intelligence and Security*. 83–91.

[33] Computer Emergency Response Team of Ukraine. 2022. Cyberattack on state organizations of Ukraine using the Formbook/XLoader malware (CERT-UA4125). https://cert.gov.ua/article/37688.

[34] Meng Shen, Jinpeng Zhang, Liehuang Zhu, Ke Xu, and Xiaojiang Du. 2021. Accurate decentralized application identification via encrypted traffic analysis using graph neural networks. *IEEE Transactions on Information Forensics and Security* 16 (2021), 2367–2380.

[35] Payap Sirinam, Mohsen Imani, Marc Juarez, and Matthew Wright. 2018. Deep fingerprinting: Undermining website fingerprinting defenses with deep learning. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. 1928–1943.

[36] Vincent F Taylor, Riccardo Spolaor, Mauro Conti, and Ivan Martinovic. 2017. Robust smartphone app identification via encrypted network traffic analysis. *IEEE Transactions on Information Forensics and Security* 13, 1 (2017), 63–78.

[37] Thijs van Ede, Riccardo Bortolameotti, Andrea Continella, Jingjing Ren, Daniel J Dubois, Martina Lindorfer, David Choffnes, Maarten van Steen, and Andreas Peter. 2020. FlowPrint: Semi-supervised mobile-app fingerprinting on encrypted network traffic. In *Network and Distributed System Security Symposium (NDSS)*, Vol. 27.

[38] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*, I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc.

[39] Wei Wang, Ming Zhu, Xuewen Zeng, Xiaozhou Ye, and Yiqiang Sheng. 2017. Malware traffic classification using convolutional neural network for representation learning. In *2017 International conference on information networking (ICOIN)*. IEEE, 712–717.

[40] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. 2018. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3733–3742.

[41] Renjie Xie, Jiahao Cao, Enhuan Dong, Mingwei Xu, Kun Sun, Qi Li, Licheng Shen, and Menghao Zhang. 2023. Rosetta: Enabling Robust TLS Encrypted Traffic Classification in Diverse Network Environments with TCP-Aware Traffic Augmentation. In *32nd USENIX Security Symposium (USENIX Security 23)*. USENIX Association, Anaheim, CA, 625–642.

[42] Hongbo Xu, Shuhao Li, Zhenyu Cheng, Rui Qin, Jiang Xie, and Peishuai Sun. 2022. TrafficGCN: Mobile Application Encrypted Traffic Classification Based on GCN. In *GLOBECOM 2022-2022 IEEE Global Communications Conference*. IEEE, 891–896.

[43] Xiaodu Yang, Sijie Ruan, Yinliang Yue, and Bo Sun. 2024. PETNet: Plaintext-aware encrypted traffic detection network for identifying Cobalt Strike HTTPS traffics. *Computer Networks* 238 (2024), 110120. https://doi.org/10.1016/j.comnet.2023.110120

[44] Tahmina Zebin, Shahadate Rezvy, and Yuan Luo. 2022. An explainable ai-based intrusion detection system for dns over https (doh) attacks. *IEEE Transactions on Information Forensics and Security* 17 (2022), 2339–2349.

[45] Haozhen Zhang, Le Yu, Xi Xiao, Qing Li, Francesco Mercaldo, Xiapu Luo, and Qixu Liu. 2023. TFE-GNN: A Temporal Fusion Encoder Using Graph Neural Networks for Fine-grained Encrypted Traffic Classification. In *Proceedings of the ACM Web Conference 2023*. 2066–2075.