Mobile Encrypted Traffic Classification Based on Message Type Inference

Yige Chen[1,2], Tianning Zang[1,2(✉)], Yongzheng Zhang[1,2], Yuan Zhou[3], and Peng Yang[3]

[1] Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
zangtianning@iie.ac.cn
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[3] National Computer Network Emergency Response Technical Team/Coordination Center of China, Beijing, China

Abstract. With the growing attention to the security and privacy of mobile communications, advanced cryptographic protocols are widely applied to protect information confidentiality and prevent privacy leakage. These cryptographic protocols make it difficult to classify encrypted traffic for network management and intrusion detection. Existing mobile encrypted traffic classification approaches intend to alleviate this problem for TLS 1.2 encrypted traffic through modeling message attributes. However, these approaches are facing tough challenges in classifying TLS 1.3 traffic because most plaintext handshake messages are encrypted in TLS 1.3. To tackle this problem, we propose a mobile encrypted traffic classification approach based on Message Type Inference (MTI). We use a Recurrent Neural Network-Conditional Random Field (RNN-CRF) network to infer the hidden message types of encrypted handshake messages. Moreover, we employ machine learning to integrate three kinds of length features. The experimental results demonstrate that the RNN-CRF network achieves 99.92% message type inference accuracy and 98.96% F1-score on a real-world TLS 1.3 dataset and our proposed approach MTI achieves 96.66% accuracy and 96.64% F1-score on a fourteen application real-world TLS 1.3 dataset. In addition, we compare MTI with existing encrypted traffic classification approaches, which demonstrates MTI performs considerably better than state-of-the-art approaches for TLS 1.3 traffic.

Keywords: Encrypted traffic classification · Message type inference · RNN-CRF

1 Introduction

Mobile encrypted traffic classification is a fundamental research problem for network management and security, which intend to classify mobile encrypted traffic into applications. As the proportion of encrypted traffic on the Internet dramatically increases [12], many network management and cybersecurity technologies

H. Gao and X. Wang (Eds.): CollaborateCom 2021, LNICST 406, pp. 117–137, 2021.
https://doi.org/10.1007/978-3-030-92635-9_8

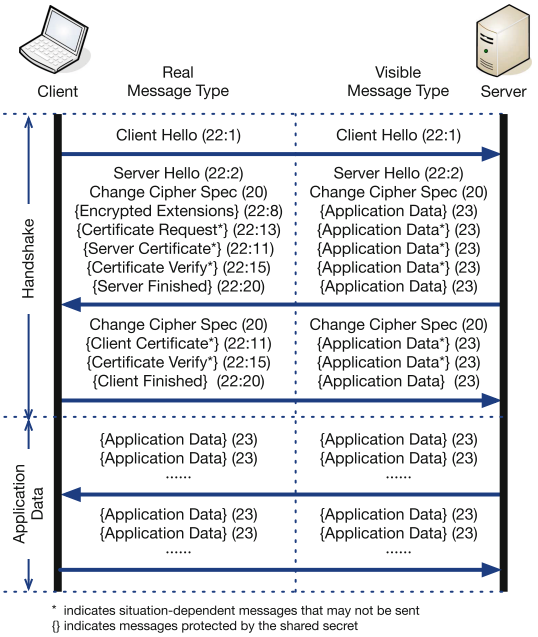

Fig. 1. A schema of TLS 1.3 communication between peers

largely rely on mobile encrypted traffic classification, such as application Quality-of-Service (QoS) [1,8] and network intrusion detection [6]. Existing encrypted traffic classification approaches primarily focus on Transport Layer Security 1.2 (TLS 1.2) [7] since it has been widely used in mobile applications [11]. With the continued development and popularity of Transport Layer Security 1.3 (TLS 1.3) [26], existing works are facing the problem of poor applicability or even inapplicability because of the confidentiality improvement of the cryptographic communication protocol. Figure 1 is a schema of TLS 1.3 communication between peers, including a client on the left and a server on the right. Each row in the figure represents a TLS message in the TLS session, and the numbers in the parentheses are message types. A full TLS 1.3 session consists of handshake messages and application data messages. The handshake messages intend to negotiate the security parameters of a session while the application data messages transfer actual payloads. The left column is a sequence of real message types before being encrypted, while the right column is a sequence of the visible message types after these messages are encrypted. Notice that many real handshake message types are masked by Application Data (23) after encryption, including Encrypted Extensions (22:8), Certificate Request (22:13), Server/Client Certificate (22:11), Certificate Verify (22:15), and Server/Client Finished (22:20). The handshake messages that can still be seen by third-party observers consist of Client Hello (22:1), Server Hello (22:2), and Change Cipher Spec (20). Therefore, the approaches based on message type information will face a significant drop in classification accuracy.

Table 1. A comparison example between named-entity recognition and message type inference

Sentence	Named-entity	Flow	Message type
All	B-event[a]	(22:1, 517)[d]	22:1
Tropical	I-event[b]	(22:2, 127)	22:2
Cyclones	I-event	(20, 6)	20
Are	O[c]	(23, 2519)	22:11
Driven	O	(20, 6)	20
By	O	(23, 58)	22:20
High	B-substance	(23, 92)	23
Heat	I-substance	(23, 292)	23
Content	I-substance	(23, 62)	23
Waters	I-substance	(23, 31)	23

[a] The B- prefix indicates the beginning of a chunk.
[b]The I- prefix indicates that the tag is inside a chunk.
[c]The O tag indicates that the token belongs to no chunk.
[d]The message feature tuple consists of visible message type and message length.

In addition to the difficulties caused by the evaluation of cryptographic protocols, the evolution of the DNS protocol also causes an impact on the attribute-based approaches. The traditional DNS protocol transfers plain DNS records without encryption. Therefore, third-party observers can directly extract the DNS records and exploit the content of these records to facilitate encrypted traffic classification [4]. The DNS over TLS (DoT) [15] and DNS over HTTPS (DoH) [14] can encrypt and protect the DNS traffic between DNS querier and recursively DNS server. Therefore, when DNS traffic is unavailable due to the use of these two secure DNS protocols by the DNS querier, we should consider how to maintain the accuracy of classification.

In this paper, we propose a mobile encrypted traffic classification approach for TLS 1.3 encrypted traffic based on Message Type Inference (MTI), which contains a Recurrent Neural Network-Conditional Random Field (RNN-CRF) network and a feature classifier based on machine learning. Although most handshake messages of TLS 1.3 encrypted traffic are encrypted and masked as application data messages, the message length and some plaintext handshake information are still available, so we can exploit these information to infer the types of encrypted handshake messages as alternative message attributes. This message type inference problem is similar to a Natural Language Processing (NLP) sequence tagging task called Named-Entity Recognition (NER). Table 1 presents a comparison example between the NER and the message type inference problem. The NER example in the table left means to determine the named-entity of the sentence's words while the message type inference in the table right is to predict the real message type of the message feature tuple that contains the visible message type and message length. A fairly common method to solve

the NER problem is adopting a Recurrent Neural Network-Conditional Random Field (RNN-CRF) network, where RNN encodes sequential words to predict Named-Entity probability distributions in word-level and CRF integrates the neighbor tag information in sentence-level to modify prediction results [16,22]. Because the protocol specification defines the handshake process, the relatively fixed chronological order of various handshake messages in an independent session can provide the neighbor tag information of TLS 1.3 messages. Take Fig. 1 as an example, for the visible message types in figure right, the ninth message Change Cipher Spec (20) explicitly divides the handshake messages from the client and server, thereby indicating the real type of neighboring messages. Therefore, we can use the RNN-CRF network to learn sequential feature and neighbor tag information to realize message type inference and conduct machine learning to integrate various features to classify mobile encrypted traffic.

We briefly summarize our contributions as follows:

- We propose a mobile encrypted traffic classification approach for TLS 1.3 encrypted traffic based on message type inference (MTI). Our approach overcomes the accuracy decline caused by the handshake message encryption.
- We adopt an RNN-CRF network to realize message type inference for encrypted handshake messages and conduct machine learning to integrate different kinds of features to realize mobile encrypted traffic classification.
- We train the RNN-CRF network upon a TLS 1.3 encrypted traffic dataset with decryption keys to recover real message types. Then, we compare MTI with state-of-the-art approaches on a real-world TLS 1.3 dataset. The experimental results show that MTI can reach 96.66% accuracy and 96.64% F1-score, which outperforms several state-of-the-art approaches.

The rest of this paper is organized as follows. Section 2 summarizes the related work. Section 3 introduces the details of mobile encrypted traffic classification based on message type inference. Section 4 introduces the evaluation and presents the experimental results. Section 5 discusses and concludes this paper.

2 Related Work

In this section, we introduce the related work of the mobile encrypted traffic classification, including traditional unencrypted traffic classification, sequence feature-based encrypted traffic classification, and attribute feature-based encrypted traffic classification.

2.1 Traditional Unencrypted Traffic Classification

Traditional unencrypted traffic-oriented approaches mainly rely on port numbers or payloads to classify unencrypted traffic into applications. The port-based approaches inspect the port numbers of the unencrypted traffic and identify the application to which the traffic belongs by looking up a pre-stated port-to-application list or a public list provided by an authority such as Internet

Assigned Numbers Authority (IANA) [24]. Nevertheless, these approaches will be invalid when the applications use port hopping or random port numbers to hide the real port. The payload-based approaches inspect the content transmitted by both communication paries and infer its application through fingerprint matches. A straightforward approach is to directly extract the feature fragments of the traffic and compare them with the collected feature records to find the most likely application [9]. With the popularity of cryptographic protocols in communications to protect user privacy, these two kinds of approaches are no longer applicable because of the unified port number and confidential payload of the encrypted traffic.

2.2 Sequence Feature-Based Encrypted Traffic Classification

The sequence feature widely used in the classification include message type sequences and packet length sequences. Many existing works adopt developed sequence-oriented models to model sequence features to conduct traffic classification, such as Markov chain models and deep learning-based models. Korczyński et al. (2014) first use first-order Markov chains to model message type sequences for each application and classify encrypted traffic by calculating the similarity between the message type sequences and each application' Markov model [18]. Then, Shen et al. (2017) strengthen the first-order Markov model to the second-order one to promote classification accuracy and propose Second-Order Markov chain fingerprints with application attribute Bigram (SOB) [28]. When the message type sequence-based approaches process encrypted traffic of TLS 1.3, these approaches can only exploit visible message type sequences for Markov chain modeling. Since most handshake messages are masked as application data messages, the diversity of message type sequences will be greatly reduced, which will lead to an inevitable accuracy decline of these approaches.

Liu et al. (2018) try to apply Markov chain models to packet length sequences and propose Multi-attribute Markov Probability Fingerprints (MaMPF) to handle the large number of unique packet lengths [20]. Since the statistical distribution of the packet length frequency obeys the power law, they choose high-frequency packet lengths as length blocks and replace other packet lengths with nearest length blocks in distance priority rules to reduce the number of unique packet lengths. Zhang et al. (2020) focus on the unknown traffic in the encrypted traffic and build an autonomous deep learning-based model on packet length sequences to filter out the encrypted traffic with low classification confidence as unknown encrypted traffic [31]. Then, they label the unknown traffic into new applications through an autonomous clustering model and add this labeled traffic to the dataset to retrain the classifier that supports the new applications. Although the message lengths are independent of the content of the message, TLS 1.3 may combine multiple handshake messages into a new single message and encrypt the single message into an application data message. Thus, some message length information may be omitted and these approaches based on the message length will face a slight accuracy reduction in classifying TLS 1.3 encrypted traffic.

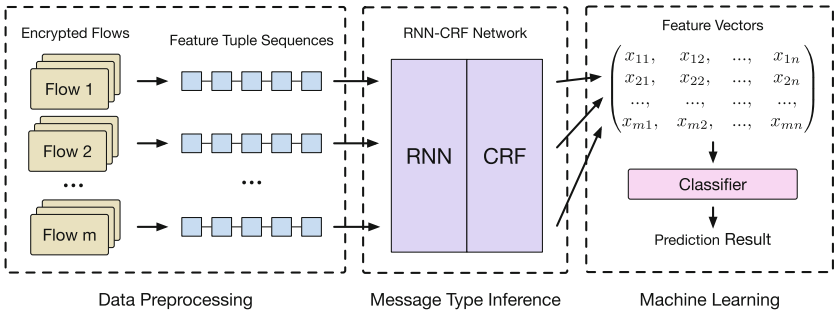

Fig. 2. The system overview of MTI

2.3 Attribute Feature-Based Encrypted Traffic Classification

The attributes commonly used to model TLS 1.2 encrypted traffic include statistic features and string features. Taylor et al. (2017) focus the statistical feature of the bursts in the flow, such as minimum, maximum, mean, median absolute deviation, standard deviation, etc. [30]. Based on the statistical features, they propose an automatic fingerprinting framework called AppScanner, which can identify the mobile application of encrypted traffic in real-time. Chen et al. (2019) propose a string feature-based approach named Multi-Attribute Associated Fingerprints (MAAF) [4]. The MAAF exploits several different attributes, including domain names, certificates, and application data lengths, to train the classifier for mobile encrypted traffic classification. The string attributes adopted by MAAF are extracted from plaintext DNS traffic and certificate messages. Currently, DNS traffic may be encrypted by DoT/DoH and certificate messages will be encrypted in TLS 1.3, which will lead to a severe compatibility challenge of MAAF in dealing with DoT/DoH and TLS 1.3.

3 System Introduction

In this section, we first provide an overview of our proposed mobile encrypted traffic classification system. Then, we describe the detail of the data preprocessing module of the system. Next, We present the design of the RNN-CRF network and the training of the network. Finally, we introduce how to perform machine learning to integrate different features for encrypted traffic classification.

3.1 System Overview

Our mobile encrypted traffic classification system consists of three modules, including Data Preprocessing, Message Type Inference, and Machine Learning, as shown in Fig. 2. The data preprocessing module extracts TLS 1.3 encrypted traffic from online raw traffic as system input and organizes this encrypted traffic into flows. In this paper, we intend to classify the encrypted traffic in units of

flows to determine which application these flows belong to. In general, a TLS 1.3 encrypted flow is composed of handshake messages for session key negotiation and application data messages that carrying communication payloads. For each TLS message, we can directly extract two features without modification, namely message length and visible message type, and store these two features as a feature tuple. Therefore, we can represent TLS 1.3 encrypted flows as feature tuple sequences for subsequent processing. We also try to extract related DNS traffic of the flow through IP address matching and adopt it as a classification feature. For all input encrypted flows, the RNN-CRF network tries to recover the real message types of encrypted handshake messages through rigorous analysis of feature tuple sequences and outputs predicted message type sequences. Based on the real message types inferred by the RNN-CRF network, we can format the sequence of feature tuples into a uniform vector of message length, where each dimension stores the length of a specific-type message. Finally, we can train a classifier for the uniform vectors through supervised machine learning and implement classification for the uniform vectors of other encrypted flows.

3.2 Data Preprocessing

Online raw traffic contains raw packets that are captured from the Internet in the order of timeline. For the captured traffic to be classified, we first select out TLS 1.3 encrypted packets and organize these packets into independent flows. The organization of flows generally relies on the IP addresses, port numbers, and TCP sequence and acknowledgment numbers. We consider flows as basic units in the traffic classification since all packets in a flow must belong to the same session initiated by a client and a server and these packets belong to the same application.

The basic transmission units of TLS flows are messages that are reassembled by TCP packets. Compared with TLS 1.2, TLS 1.3 flows encrypt many key handshake messages and mask them with Application Data (23) to protect user privacy in communication, such as Encrypted Extensions (22:8), Certificate Verify (22:15), and Server Certificate (22:11). For each TLS 1.3 flow, we extract the message lengths and the visible message types of all messages to form a feature tuple sequence, where each tuple contains the message length and visible message type of a TLS message.

In general, mobile applications will find the optimal server IP address in the application initialization phase by querying the recursive DNS server for the preset domain name of the application server. Thus, the preset domain name is particularly relevant to the application and the DNS traffic can be used as an important classification feature in the mobile encrypted traffic classification [4]. We can find out the related plain text DNS traffic of a given encrypted flow by comparing the answer record of the DNS traffic and the server IP address of the flow. However, we cannot get the DNS traffic in the following situations and should ignore the related DNS traffic in the traffic classification. 1) The application connects with the server through hard-coded IP addresses. 2) The application obtains server IP addresses through an alive TLS session with the

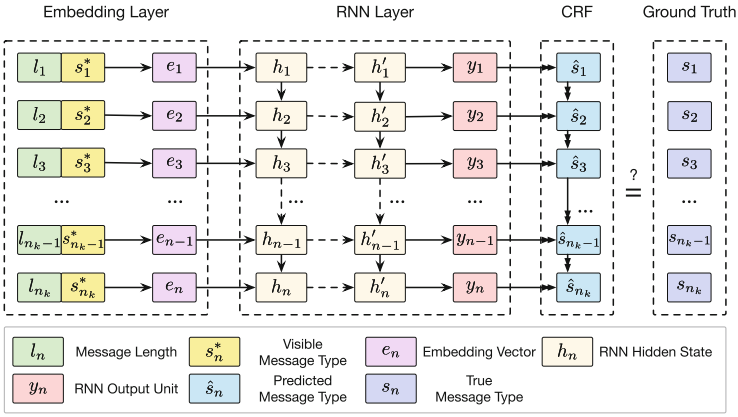

Fig. 3. The architecture of RNN-CRF network

connected server. 3) The client adopts DNS over HTTPS (DoT) or DNS over TLS (DoH) technology to connect a recursive DNS server for domain name querying.

3.3 Message Type Inference

Problem Definition. We first formulate the problem of message type inference. Let the sequential feature tuples of the k_{th} encrypted flow extracted by the data preprocessing module be

$$X^{(k)} = [(l_1^{(k)}, s_1^{*(k)}), (l_2^{(k)}, s_2^{*(k)}), ..., (l_{n_k}^{(k)}, s_{n_k}^{*(k)})] \tag{1}$$

where $l_i^{(k)}$ is the length of the i_{th} message, $s_i^{*(k)} \in S_{type}$ is the visible message type of the i_{th} message, n_k is the number of messages in the k_{th} flow. For the k_{th} encrypted flow, when we obtain the TLS secret of the encrypted flow, we can parse out the ground truth message type sequence

$$Y^{(k)} = [s_1^{(k)}, s_2^{(k)} ..., s_{n_k}^{(k)}] \tag{2}$$

where $s_i^{(k)} \in S_{type}$ is the real message type of the i_{th} message. We aim to build a sequence labeling model $\Phi(X^{(k)})$ to predict the message type sequence $\widehat{Y}^{(k)}$, and try to make the prediction result $\widehat{Y}^{(k)}$ the same as the ground truth message type sequence $Y^{(k)}$.

RNN-CRF Network. To solve this sequence labeling problem, we propose an RNN-CRF network that consists of an embedding layer, a recurrent neural network (RNN), and a conditional random field (CRF), as shown in Fig. 3. The embedding layer embeds the input feature tuple that consists of the message length l_i and the visible message type s_i^* into a integrated embedding vector e_i

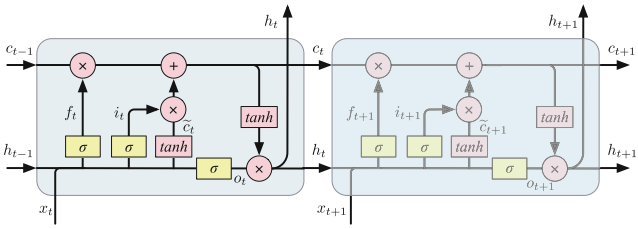

Fig. 4. The architecture of LSTM cells

that contains both the message length and message type information. The RNN layer is designed for processing and encoding sequence inputs and regulate the flow of information along the sequence. As a result, the hidden states h_i is a representation of all previous inputs. The RNN output y_i corresponds to the i_{th} message and represents the feature of the predicted message type. Finally, the conditional random field can reasonably combine the neighbor tag information of the previously predicted message type $\hat{s}_{i-1}$ with the predicted feature y_i provided by the RNN layer to infer the message type $\hat{s}_i$.

Embedding Layer. The input of the RNN-CRF network is feature tuple sequences, so we need to embed feature tuples into embedding vectors for the subsequent numerical calculation. We first create two randomly initialized character embedding matrix $E_l, E_{s^*} \in \mathbb{R}^{K \times d}$ for message length embedding and visible message type embedding, where K is the size of feature element set and d is the dimension of embedding vectors. In essence, the embedding matrixes are embedding vector lookup tables, and we can map the feature tuple $(l_i^{(k)}, s_i^{*(k)}), i \in [1, n_k]$ into a tuple of two independent embedding vectors $(E_l(l_i^{(k)}), E_{s^*}(s_i^{*(k)}))$. Since the subsequent RNN-CRF network only receives numerical vector sequences, and the embedding dimensions of two kinds of features are the same, we can concatenate two kinds of independent embedding vectors into an integrated vector and define the embedding function as $e_i^{(k)} = E(l_i^{(k)}, s_i^{*(k)}) = E_l(l_i^{(k)}) \oplus E_{s^*}(s_i^{*(k)})$.

Compared with scalar message lengths, the embedding vectors have multiple dimensions to form the feature information and can increase the discrimination of different message lengths. We represent the discrete visible message types with embedding vectors so that we can perform numerical calculations on the message type features. Besides, the embedding matrixes are trainable during the model training, and we can learn the task-oriented optimal values of these two embedding matrixes to improve the representation of the embedding vectors.

RNN Layer. The Recurrent Neural Network (RNN) is one kind of neural network where connections between RNN cells form a directed graph along a temporal sequence [27]. The RNN is widely used to process sequence inputs in many research fields, such as text classification in natural language processing [32]. The hidden state of the RNN cell typically stores the information of inputted sequence elements. Along with the input of sequence elements, RNN calculates

the current hidden state based on the previous state and current input. As a result, RNN can integrate the information of all elements in sequence order, which determines its suitability for processing sequence input.

In this paper, we adopt a variant of RNN called Long Short-Term Memory (LSTM) [13]. As shown in Fig. 4, a common LSTM unit at time step $i \in [1, n_k]$ contains a cell $\mathtt{c}_i$, an input gate $\mathtt{i}_i$, an output gate $\mathtt{o}_i$, and a forget gate $\mathtt{f}_i$. The outputs of input gate, output gate, and forgot gate at time step i are

$$\mathtt{i}_i = \sigma(W_\mathtt{i} \cdot [\mathtt{h}_{i-1}, x_i] + b_\mathtt{i}) \tag{3}$$

$$\mathtt{o}_i = \sigma(W_\mathtt{o} \cdot [\mathtt{h}_{i-1}, x_i] + b_\mathtt{o}) \tag{4}$$

$$\mathtt{f}_i = \sigma(W_\mathtt{f} \cdot [\mathtt{h}_{i-1}, x_i] + b_\mathtt{f}) \tag{5}$$

where $W_\mathtt{i}$, $W_\mathtt{o}$, and $W_\mathtt{f}$ are trainable weight matrices. $b_\mathtt{i}$, $b_\mathtt{o}$, and $b_\mathtt{f}$ are bias vector parameters need to be learned. $\mathtt{h}_{i-1}$ is the hidden state at time step $i-1$. The sigmoid function $\sigma = 1/(1+e^{-x})$ is an activation function to introduce non-linearity. The cell input $\tilde{\mathtt{c}}_i$, cell state $\mathtt{c}_i$, and hidden state $\mathtt{f}_i$ are

$$\tilde{\mathtt{c}}_i = tanh(W_\mathtt{c} \cdot [\mathtt{h}_{i-1}, x_i] + b_\mathtt{c}) \tag{6}$$

$$\mathtt{c}_i = \mathtt{f}_i \cdot \mathtt{c}_{i-1} + \mathtt{i}_i \cdot \tilde{\mathtt{c}}_i \tag{7}$$

$$\mathtt{h}_i = \mathtt{o}_i \cdot tanh(\mathtt{c}_i) \tag{8}$$

where $W_\mathtt{c}$ and $b_\mathtt{c}$ are trainable weight and bias. The forgot gate $\mathtt{f}_i$ and input gate $\mathtt{i}_i$ jointly determine how to forget the previous cell state $\mathtt{c}_{i-1}$ and how to receive the current cell input $\tilde{\mathtt{c}}_i$. Finally, the output gate $\mathtt{o}_i$ controls how information is processed as the hidden state $\mathtt{h}_i$. The three gates of the LSTM effectively control the flow of information between adjacent LSTM cells and solve the vanishing gradient problem of vanilla RNN [23]. For the hidden state $\mathtt{h}_i$ at time step i, we use a dense layer to map the message type features to the tagging classes. The output score of the RNN layer r_i is

$$r_i = \sigma(W_r \cdot \mathtt{h}_i + b_r) \tag{9}$$

where W_r and b_r are the parameters of the dense layer. The r_i is a classification score vector where each dimension stores the score of the corresponding tag.

Conditional Random Field. The conditional random field (CRF) is a statistical model that considers neighboring information to make structured predictions [19]. Therefore, the CRF is suitable to solve the sequence tagging problem by making use of neighbor tag information. Take the NER task in natural language processing as an example, the linear chain CRFs can efficiently use past tags to predict the current tag [16,22]. In this paper, we connect a CRF network after the RNN layer to form an RNN-CRF network, which combines the feature of past inputs and neighbor tag information at sentence level to perform message type tagging. The parameter of a CRF layer is a state transition metric.

Suppose the state transition metric $|A|_{s,s'}$ determines the transition score from the message type s to s' in consecutive time steps and the transition metric $|A|_{s,s'}$ is independent of the message type position in the sequence. The parameter of the RNN-CRF network is

$$\tilde{\theta} = \tilde{\theta}_{RNN} + |A|_{s_i,s_j} \tag{10}$$

where $\tilde{\theta}_{RNN}$ if the parameters of the RNN layer. The final score of the flow $X^{(k)}$ with a message type prediction $\widehat{Y}^{(k)}$ is the sum of CRF score and RNN layer score, which is as follows

$$score(X^{(k)}, \widehat{Y}^{(k)}, \tilde{\theta}) = \sum_{i=1}^{n_k} (|A|_{\hat{s}_{i-1}^{(k)}, \hat{s}_i^{(k)}} + r_{i_{\hat{s}_i^{(k)}}}) \tag{11}$$

where $\hat{s}_i^{(k)}$ is the predicted message type of the i_{th} message in k_{th} flow, r_i is score output of RNN layer for i_{th} message. Given a flow input $X^{(k)}$ and RNN-CRF network parameters $\tilde{\theta}$, we can get the optimal message type prediction $\widehat{Y}^{(k)}$ with the highest final score by the Viterbi algorithm [10].

Network Training. In order to train the RNN-CRF network on the collected dataset, we use a negative log-likelihood function $-\log p(Y|X)$ as the loss function, where X, Y indicate the input flow and the corresponding application, respectively. The likelihood function $p(Y|X)$ is defined as the probability of classifying X as Y, so we intend to maximize the $p(Y|X)$ in the training of the RNN-CRF network, which is equivalent to minimize the negative log-likelihood function $-\log p(Y|X)$.

In the previous subsection, we have defined the prediction score $score(X, Y)$ of the flow X with a prediction Y. Therefore, we can define the likelihood function as the quotient of the power of the prediction score $e^{score(X,Y)}$ and the power sum of all potential Y scores $\sum_Y e^{score(X,Y)}$. So, the loss function of the RNN-CRF network can be defined as

$$\begin{aligned} loss &= -\log p(Y|X) = -\log \frac{e^{score(X,Y)}}{\sum_Y e^{score(X,Y)}} \\ &= -(score(X,Y) - \log \sum_i Z(Y_n = i)) \end{aligned} \tag{12}$$

where $\sum_i Z(Y_n = i)$ is the power sum of all n-length potential path score with $Y_n = i$. We define $Z(Y_t = i)$as

$$Z(Y_t = i) = \sum_{Y_t} e^{score(start \to Y_t)} \tag{13}$$

We can recursively calculate $\log Z(Y_t = i)$ when we obtain the power sum of all $t-1$-length potential path score, the state transition metric $|A|$, and the RNN layer score r_t

$$\begin{aligned} \log Z(Y_t = i) &= \log \sum_{Y_t} e^{score(start \to Y_t)} \\ &= \log \sum_{Y_{t-1}=1}^{m} e^{score(start \to Y_{t-1}) + |A|_{Y_{t-1}, Y_t} + r_{t\,Y_t}} \end{aligned} \tag{14}$$

where m indicates the number of potential message type tags in the prediction. As a result, we can improve the calculation speed of $\sum_i Z(Y_n = i)$ through dynamic programming algorithm. Based on the loss function, we can use a gradient descent algorithm to train the network upon the collected dataset.

3.4 Machine Learning

Based on the real message type sequence inferred by the RNN-CRF network, we can align the message types of the encrypted handshake messages in different encrypted flows and generate the feature vectors of these encrypted flows. The features we use include the length of the relevant DNS packet, the lengths of the type-tagged encrypted handshake messages, and the lengths of application data messages. For a given encrypted flow, we can obtain these features in chronological order as the communication is established and progressed. Therefore, we can determine the combination of these features according to the performance requirements of real-time prediction and classification accuracy. When we adopt more features in traffic classification, we can generally achieve higher classification accuracy while we need to face a slower classification efficiency and a larger classification latency. In mobile encrypted traffic classification, when we need to predict the application of the encrypted flow before its establishment, we can only exploit the length of DNS traffic to make classifications. When we want to predict the application of the encrypted flow before payload transmission, we can make use of the length of the DNS traffic and the lengths of the type-tagged handshake messages based on the trained RNN-CRF network. When we require a high-accurate encrypted traffic classifier, we should adopt all available features in both the training of the classifier and the classification of the encrypted traffic.

We adopt several supervised machine learning methods to train an effective classifier on these feature vectors for mobile encrypted traffic classification. In this paper, we select three alternative supervised machine learning models, including C4.5 [25], Random Forest [2], and XGBoost [3]. The C4.5 is a landmark algorithm to generate a decision tree for a given set of feature vectors. In detail, the C4.5 adopts the normalized information gain as its splitting criterion to build the decision tree, so the training and prediction of the C4.5 are remarkably efficient. The Random Forest is an ensemble learning method designed for classification or regression tasks. Compared with the single tree generated by C4.5, the Random Forest builds a set of decision trees for random training data and feature sampling results to solve the overfitting problem caused by the single decision tree. The XGBoost is an optimized gradient boosting decision tree implementation designed for both efficiency and effectiveness. The gradient boosting decision tree is an ensemble model of decision trees that are

Table 2. Summary of the experimental dataset

#	Application	Flows	Packets	Domain	Size (MB)
1	Booking.com	3690	695989	42	498.79
2	Breitbart	3515	516367	126	318.38
3	Canva	3762	610755	20	418.77
4	ESPN	3885	408017	140	236.72
5	Facebook	3634	1925147	86	1300.90
6	Fox News	3690	812188	111	500.00
7	Okezone	3736	338559	66	197.27
8	Quizlet	3735	525844	41	324.50
9	Spotify	3712	296209	14	158.48
10	Steam	3771	1004865	24	695.20
11	VK	3851	1329839	137	917.81
12	Wikipedia	3684	588059	6	337.96
13	Yahoo! News	3539	850082	145	562.11
14	Zillow	3886	626698	65	415.16
	Total	52090	10528618	1023	6882.04

recursively created to predict the errors or prior created trees. Therefore, each machine learning method has its specific advantages and applicable scenarios. The C4.5 is suitable for large-scale classification tasks because of its simplicity and efficiency. The feature sampling mechanism of Random forest can effectively avoid the overfitting problem in features learning and provides high classification accuracy. The gradient boosting of XGBoost can effectively deal with complex features and generates a set of cooperating decision trees to achieve high classification accuracy. We can choose the appropriate method according to the performance requirement of the specific scenario.

4 Evaluation

In this section, we first introduce the experimental datasets in the evaluation. Then, we specify the experimental settings and describe the definition of the evaluation metrics. Finally, we present experimental results of the message type inference, our proposed approach, and the comparisons with existing approaches.

4.1 Preliminary

Dataset. To objectively evaluate the effectiveness of our proposed classification approach, we first need to collect a TLS 1.3 mobile encrypted traffic dataset and a TLS 1.3 traffic dataset with ground truths of encrypted handshake message

Table 3. Summary of the dataset with TLS decryption keys and the experimental results of RNN-CRF

Message type	Dataset		RNN-CRF (GRU)		RNN-CRF (LSTM)	
	Count	Ratio	Prec.	Rec.	Prec.	Rec.
20	19988	0.1254	**1.0000**	**1.0000**	**1.0000**	**1.0000**
21	3632	0.0228	**0.9980**	**0.9980**	**0.9980**	**0.9980**
22:1	10225	0.0641	**1.0000**	**1.0000**	**1.0000**	**1.0000**
22:11	3569	0.0224	**0.9889**	0.9944	0.9882	**0.9979**
22:15	3075	0.0193	**0.9992**	0.9992	**0.9992**	**1.0000**
22:2	10233	0.0642	**1.0000**	**1.0000**	**1.0000**	0.9998
22:20	17484	0.1097	**0.9996**	0.9997	0.9991	**0.9999**
22:25:15:20	60	0.0004	**1.0000**	**1.0000**	**1.0000**	**1.0000**
22:4	7685	0.0482	**0.9980**	0.9987	**0.9980**	**0.9997**
22:4:4	2056	0.0129	**0.9988**	**1.0000**	0.9976	0.9988
22:8	7724	0.0485	**0.9977**	0.9994	0.9974	**0.9997**
22:8:20	1639	0.0103	**1.0000**	**1.0000**	**1.0000**	0.9984
22:8:25:15:20	187	0.0012	0.8824	**0.8000**	**0.9524**	**0.8000**
23	71732	0.4500	0.9996	**0.9993**	**0.9999**	0.9992
Total/Acc/F1	159407	1.0000	0.9991	0.9874	**0.9992**	**0.9896**

types. Chen et al. [4] describe two kinds of mobile application traffic collection schemes, namely Active Traceset Collection and Passive Traceset Collection. The active scheme collects mobile application traffic by tracking the mobile phone connected to a controllable workstation and trying to operate mobile applications in the mobile phone to generate online traffic of the specified mobile applications. The passive scheme first collects unlabeled traffic from pre-deployed port mirroring switches and utilizes DNS records to separate the traffic of different applications. To purposefully collect encrypted traffic of TLS-1.3-employed applications, we adopt the Active Traceset Collection scheme and collect a real-world dataset that contains 52,090 encrypted flows of fourteen mobile applications. Table 2 presents the summary of the experimental dataset.

We also collect a TLS 1.3 traffic dataset with TLS decryption keys by assigning a desktop web browser to access TLS-1.3-employed websites and capturing both the encrypted browsing traffic and TLS decryption keys. Based on the TLS decryption keys, we can decrypt TLS handshake messages and recover message type ground truths to support the training of the RNN-CRF network. Table 3 left presents the statistic of the TLS-1.3 traffic dataset with ground truths of encrypted handshake message types.

Experiment Settings. We first consider the parameters of the RNN-CRF network for message type inference. We embed sequential elements to 50 dimension vectors through the embedding layer. We set the number of the RNN layers to 2

and set the dimension of the RNN hidden states to 100. We enhance the robustness of the neural network by setting the dropout [29] to 0.5 to randomly omit network nodes during model training. Besides, we adopt Adam optimizer [17] with a 0.0002 learning rate for the training of the RNN-CRF network.

The machine learning for the feature vectors should consider some encrypted traffic-related parameters. The feature vector of a given encrypted flow includes the length of the relevant DNS packets, the lengths of handshake messages, and the lengths of application data messages. We limit the maximum number of the adopted messages in machine learning to 15. The machine learning model is selected from C4.5, Random Forest, and XGBoost. We conduct a random parameter search and employ 5-fold cross-validation to obtain the optimal parameters of the selected machine learning model on the training data.

We randomly divide the experimental dataset into a training dataset and a test dataset, which respectively account for 70% and 30% of the original dataset. We take the average results of 5 repeated experiments to reduce random errors.

Evaluation Metrics. We employ three common metrics to evaluate the effectiveness of a given approach on the mobile encrypted traffic classification task. The precision and recall of the application i are defined as follows:

$$Precision_i = \frac{TP_i}{TP_i + FP_i}, \quad Recall_i = \frac{TP_i}{TP_i + FN_i} \tag{15}$$

where True Positive TP_i indicates the number of application i's samples that are correctly classified to application i, False Positive FP_i indicates the number of samples that are wrongly classified to application i, and False Negative FN_i indicates the number of application i's samples that are misclassified to other classes. Besides, we use F1-score to represent the overall classification effectiveness, which is defined as:

$$F1_i = \frac{Precision_i * Recall_i}{Precision_i + Recall_i}, \quad F1 = \frac{\sum_{i=1}^{n} F1_i}{n} \tag{16}$$

where the F1-score of application i is defined as the harmonic mean of $Precision_i$ and $Recall_i$, and the overall F1-score is defined as the macro average of all $F1_i$.

4.2 Analysis of the Message Type Inference

We first study the effectiveness of the RNN-CRF network in inferring the real message types of encrypted messages. We evaluate the RNN-CRF network on the TLS 1.3 traffic dataset with ground truths of handshake message types. Since we focus on the accuracy of message type inference, we calculate the evaluation metrics in the granularity of messages instead of flows. Our RNN-CRF network adopts LSTM to build its RNN layer, so we compare it with another RNN variant called Gated Recurrent Unit (GRU) [5].

Table 3 gives the experimental results of using LSTM and GRU RNN-CRF network for message type inference. Both LSTM and GRU perform high accuracy, while LSTM is slightly better than GRU. The F1-scores of LSTM and

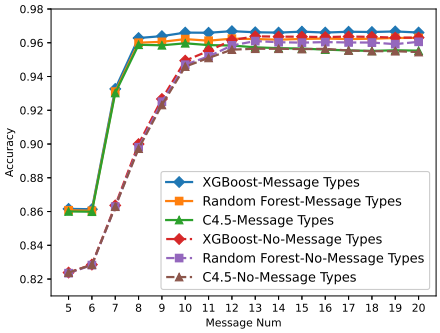

Fig. 5. Comparison results of the inferred message types

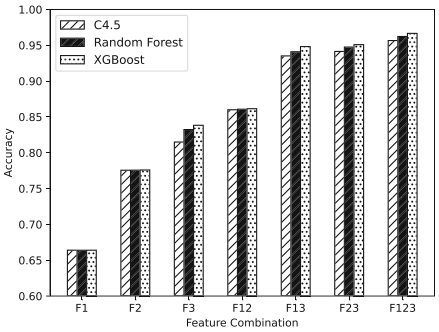

Fig. 6. Comparison results of the feature combination

GRU reach 98.96% and 98.74%, respectively. Although the dataset is imbalanced for some message types, the RNN-CRF network can still show high precision and recall for the message types with a small volume, such as 22:25:15:20 and 22:8:25:15:20. Based on the experimental results, we can train an RNN-CRF network upon the whole dataset with TLS decryption keys and use the model to predict the message types of the encrypted messages in other datasets.

In this subsection, we study the effectiveness of the adopted features in machine learning, including the length of relevant DNS packets, the inferred message types of encrypted handshake messages, the lengths of handshake messages, the lengths of application data messages. We first study the impact of using the inferred message types of encrypted handshake messages on classification accuracy. When we do not use inferred message types to align the message types of the encrypted handshake messages in feature vector generation, we directly use the length of relevant DNS packets and message lengths in chronological order to build feature vectors. Besides, we vary the number of adopted message lengths to study the impact of the inferred message types under different numbers of message lengths. We employ three machine learning models to eliminate the random error caused by the machine learning model.

4.3 Analysis of Adopted Features

Figure 5 shows the comparison results of the inferred message types. We can find that when the number of message lengths takes from 5 to 11, the classification accuracy of using massage types is larger than not using it for all three machine learning methods. Besides, the accuracy disparity between using message types and not using message types increases sharply first and then decreases slowly. The classification accuracy of using message types is overall higher than not using message types. This may be because when the number of messages is small, aligning the message lengths of the same message type in constructing feature vectors can reasonably avoid the information deviation caused by the dimension dislocation of the feature vectors. Thus, when the number of messages is limited

Table 4. Experimental results of different machine learning models

Application	C4.5		Random forest		XGBoost	
	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.
Booking.com	0.9646	0.9718	0.9715	0.9788	**0.9765**	**0.9820**
Breitbart	0.8731	0.9145	0.8876	0.9279	**0.8886**	**0.9369**
Canva	0.9445	0.9705	0.9427	0.9697	**0.9478**	**0.9724**
ESPN	0.9190	0.9058	0.9277	0.9127	**0.9351**	**0.9253**
Facebook	0.9867	0.9818	**0.9953**	0.9821	0.9936	**0.9837**
Fox News	0.9319	0.9115	0.9410	0.9299	**0.9417**	**0.9380**
Okezone	0.9488	0.9582	0.9638	0.9732	**0.9702**	**0.9740**
Quizlet	0.9463	0.9427	0.9596	0.9438	**0.9601**	**0.9551**
Spotify	0.9838	0.9658	0.9711	**0.9708**	**0.9862**	0.9674
Steam	0.9829	0.9840	0.9753	0.9893	**0.9849**	**0.9901**
VK	0.9939	0.9910	0.9939	0.9876	**0.9955**	**0.9926**
Wikipedia	0.9979	0.9995	0.9987	**1.0000**	**0.9992**	**1.0000**
Yahoo! News	0.9622	0.9404	0.9717	0.9461	**0.9736**	**0.9504**
Zillow	0.9554	0.9525	0.9713	0.9581	**0.9790**	**0.9627**
Acc/F1	0.9566	0.9564	0.9623	0.9621	**0.9666**	**0.9664**

to a small number, the usage of message types can significantly improve the classification accuracy of encrypted traffic classification.

Then, we study the effect of other features in machine learning, including the length of the relevant DNS packet, the lengths of handshake messages, the lengths of application data messages. We intend to evaluate the classification accuracy of all kinds of feature combinations. We notate the length of the relevant DNS packet, the lengths of handshake messages, the lengths of application data messages as F1, F2, F3, respectively, and use the combination of notations to indicate the combination of features.

Figure 6 shows the experimental results of different feature combinations. As shown in the figure, the classification accuracy increases with the enrichment of the features. The accuracy of the combination of three features is significantly higher than other combinations and the accuracy of the combinations of two features is also higher than single features. In addition, we can find that when we only use one feature for machine learning, the lengths of application data messages can provide more discrimination information than the other two features. This may be because the lengths of application data messages are relevant to the payload transmission pattern of the application. Because of the TLS protocol specification, the number and content of handshake messages are somewhat restricted, so the lengths of handshake messages perform slightly lower classification accuracy than application data messages. The legal lengths of DNS packets are limited by the DNS protocol specification and each flow generally corresponds to one relevant DNS packet, so it performs the worst among the three

features. Since the length of application data messages is particularly important for feature learning, when we have high requirements for both real-time performance and classification accuracy, we need to focus on balancing the number of application data messages and the time cost to capture traffic.

We also compare the classification results of using different machine learning models, as presented in Table 4. We find that the accuracy of XGBoost achieves 96.66%, and the F1-score achieves 96.64%, which is a little better than C4.5 and Random Forest. XGBoost also has better precision and Recall for most applications, which demonstrates that when we only focus on the classification accuracy, XGBoost is indeed better than the other two models. Therefore, when the encrypted traffic classification task requires high accuracy and can tolerate other relatively poor performances, such as time complexity, we prefer to choose XGBoost as the machine learning model.

4.4 Comparisons with Existing Approaches

The attribute-based approaches face severe compatibility problems to the encrypted handshake messages, so we only compare our proposed approach with three sequence-based state-of-the-art approaches. The comparison approaches include Second-Order Markov chain fingerprints with application attribute Bigram (SOB) [28], Multi-attribute Markov Probability Fingerprints (MaMPF) [20], and Flow Sequence Network (FS-Net) [21]. SOB employees second-order Markov chain to model message type sequence and use machine learning models to incorporate the lengths of certificate messages and application data messages [28]. Since most handshake messages including the certificate message are encrypted as application data messages, we take the visible message type sequence as model input and leave the length of the certificate empty. MaMPF transfers first message length sequences into length block sequences and then employees Markov chain to model message type sequences and length block sequences [20]. So, we can directly apply MaMPF on the message length sequence and visible message type sequence of TLS 1.3 encrypted traffic. Compared with the former two approaches, FS-Net only uses an auto-encoder neural network to model message length sequences [21], so we can fully apply FS-Net on TLS 1.3 encrypted traffic.

Table 5 presents the experimental results of comparisons with existing approaches. As shown in the table, the accuracy and F1-score of MTI reach 96.66% and 96.64% respectively, which are significantly better than the other approaches. The accuracy and F1-score of SOB are only 61.77% and 61.45%. This is because the message types of most handshake messages are masked by application data messages, which conceals much of the feature information for the encrypted traffic classification. MaMPF performs better than SOB, which is attributed to the usage of message length sequence information. The accuracy and F1-score of FS-Net reach 90.92% and 90.90%. This demonstrates that FS-Net can appropriately deal with the classification of TLS 1.3 encrypted traffic.

Table 5. Experimental results of different approches

Application	SOB		MaMPF		FS-Net		MTI	
	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.
Booking.com	0.6914	0.5850	0.7372	0.5190	0.9291	0.9321	**0.9765**	**0.9820**
Breitbart	0.5376	0.4520	0.6895	0.5741	0.6963	0.8249	**0.8886**	**0.9369**
Canva	0.8301	0.5877	0.6990	0.7327	0.9231	0.9698	**0.9478**	**0.9724**
ESPN	0.6457	0.5002	0.8221	0.6393	0.8656	0.7916	**0.9351**	**0.9253**
Facebook	0.6158	0.8406	0.6205	0.8059	0.9921	0.9679	**0.9936**	**0.9837**
Fox News	0.4346	0.3820	0.6338	0.5799	0.8685	0.7703	**0.9417**	**0.9380**
Okezone	0.5311	0.6233	0.8478	0.6695	0.8726	0.9214	**0.9702**	**0.9740**
Quizlet	0.6362	0.6589	0.7530	0.8257	0.9030	0.8957	**0.9601**	**0.9551**
Spotify	0.6639	0.8272	0.6206	0.9216	0.9698	0.9588	**0.9862**	**0.9674**
Steam	0.7816	0.6889	0.9029	0.7276	0.9187	0.9455	**0.9849**	**0.9901**
VK	0.7421	0.7673	0.7410	0.8452	0.9686	0.9812	**0.9955**	**0.9926**
Wikipedia	0.7814	0.8563	0.9514	0.5900	0.9936	0.9900	**0.9992**	**1.0000**
Yahoo! News	0.4102	0.5603	0.5048	0.8315	0.9020	0.8765	**0.9736**	**0.9504**
Zillow	0.4088	0.3447	0.7714	0.6595	0.9475	0.8994	**0.9790**	**0.9627**
Acc/F1	0.6177	0.6145	0.7093	0.7084	0.9092	0.9090	**0.9666**	**0.9664**

5 Discussion and Conclusion

In this paper, we design an RNN-CRF network to infer the real message types of TLS 1.3 encrypted handshake messages and classify TLS 1.3 mobile encrypted traffic through the machine learning. The RNN network primarily considers the input visible message type and message length information, while the CRF network focuses on the neighbor tag information of the message type sequence. The evaluation results demonstrate the applicability of MTI to TLS 1.3 traffic, which shows 96.66% classification accuracy and 96.64% F1-score. However, when we encounter encrypted DNS traffic of DoT/DoH, we cannot obtain the relevant DNS information of the encrypted flow but only make use of message type and length information, which would cause slight classification accuracy decreases. In future work, we plan to study how to make use of DoT/DoH traffic as an encrypted traffic classification feature and then maintain the classification accuracy when encountering the DNS traffic of DoT/DoH.

Acknowledgment. This work is supported by the Strategic Priority Research Program of the Chinese Academy of Sciences (No.XDC02030100), the National Key Research and Development Program of China (Grant No.2018YFB0804704), and the National Natural Science Foundation of China (Grant No.U1736218).

References

1. Aceto, G., Ciuonzo, D., Montieri, A., Pescapé, A.: Mobile encrypted traffic classification using deep learning. In: 2018 Network traffic measurement and analysis conference (TMA), pp. 1–8. IEEE (2018)
2. Breiman, L.: Random forests. Mach. Learn. **45**(1), 5–32 (2001)
3. Chen, T., Guestrin, C.: Xgboost: a scalable tree boosting system. In: Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 785–794. ACM (2016)
4. Chen, Y., Zang, T., Zhang, Y., Zhouz, Y., Wang, Y.: Rethinking encrypted traffic classification: a multi-attribute associated fingerprint approach. In: 2019 IEEE 27th International Conference on Network Protocols (ICNP), pp. 1–11. IEEE (2019)
5. Cho, K., et al.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1724–1734 (2014)
6. Cova, M., Kruegel, C., Vigna, G.: Detection and analysis of drive-by-download attacks and malicious javascript code. In: Proceedings of the 19th international conference on World wide web, pp. 281–290 (2010)
7. Dierks, T., Rescorla, E.: The transport layer security (tls) protocol version 1.2. IETF RFC5246 (2008)
8. Fiedler, M., Hossfeld, T., Tran-Gia, P.: A generic quantitative relationship between quality of experience and quality of service. IEEE Netw. **24**(2), 36–41 (2010)
9. Finsterbusch, M., Richter, C., Rocha, E., Muller, J.A., Hanssgen, K.: A survey of payload-based traffic classification approaches. IEEE Commun. Surv. Tutor. **16**(2), 1135–1156 (2013)
10. Forney, G.D.: The viterbi algorithm. Proc. IEEE **61**(3), 268–278 (1973)
11. Google: Google Online Security Blog: An Update on Android TLS Adoption. https://security.googleblog.com/2019/12/an-update-on-android-tls-adoption.html
12. Google: HTTPS encryption on the web - Google Transparency Report. https://transparencyreport.google.com/https/overview
13. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
14. Hoffman, P., McManus, P.: Dns queries over https (doh). IETF RFC8484 (2018)
15. Hu, Z., Zhu, L., Heidemann, J., Mankin, A., Wessels, D., Hoffman, P.: Specification for dns over transport layer security (tls). IETF RFC7858 (2016)
16. Huang, Z., Xu, W., Yu, K.: Bidirectional lstm-crf models for sequence tagging. arXiv preprint arXiv:1508.01991 (2015)
17. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
18. Korczyński, M., Duda, A.: Markov chain fingerprinting to classify encrypted traffic. In: 2014 IEEE International Conference on Computer Communications (Infocom), pp. 781–789. IEEE (2014)
19. Lafferty, J., McCallum, A., Pereira, F.C.: Conditional random fields: probabilistic models for segmenting and labeling sequence data. In: Proceedings of the Eighteenth International Conference on Machine Learning, pp. 282–289. Morgan Kaufmann (2001)
20. Liu, C., Cao, Z., Xiong, G., Gou, G., Yiu, S.M., He, L.: Mampf: encrypted traffic classification based on multi-attribute markov probability fingerprints. In: 2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS), pp. 1–10. IEEE (2018)

21. Liu, C., He, L., Xiong, G., Cao, Z., Li, Z.: Fs-net: a flow sequence network for encrypted traffic classification. In: 2019 IEEE International Conference on Computer Communications (Infocom), pp. 1–9. IEEE (2019)
22. Ma, X., Hovy, E.: End-to-end sequence labeling via bi-directional LSTM-CNNS-CRF. arXiv preprint arXiv:1603.01354 (2016)
23. McCloskey, M., Cohen, N.J.: Catastrophic interference in connectionist networks: the sequential learning problem. In: Psychology of learning and motivation, vol. 24, pp. 109–165. Elsevier (1989)
24. Qi, Y., Xu, L., Yang, B., Xue, Y., Li, J.: Packet classification algorithms: from theory to practice. In: IEEE INFOCOM 2009, pp. 648–656. IEEE (2009)
25. Quinlan, J.R.: C4.5: programs for machine learning. Elsevier (2014)
26. Rescorla, E.: The transport layer security (tls) protocol version 1.3. IETF RFC8446 (2018)
27. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. Nature **323**(6088), 533–536 (1986)
28. Shen, M., Wei, M., Zhu, L., Wang, M.: Classification of encrypted traffic with second-order Markov chains and application attribute bigrams. IEEE Trans. Inf. For. Secur. **12**(8), 1830–1843 (2017)
29. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)
30. Taylor, V.F., Spolaor, R., Conti, M., Martinovic, I.: Robust smartphone app identification via encrypted network traffic analysis. IEEE Trans. Inf. For. Secur. **13**(1), 63–78 (2017)
31. Zhang, J., Li, F., Ye, F., Wu, H.: Autonomous unknown-application filtering and labeling for dl-based traffic classifier update. In: 2020 IEEE International Conference on Computer Communications (Infocom), pp. 1–9. IEEE (2020)
32. Zhou, P., Qi, Z., Zheng, S., Xu, J., Bao, H., Xu, B.: Text classification improved by integrating bidirectional LSTM with two-dimensional max pooling. arXiv preprint arXiv:1611.06639 (2016)