ETCC: Encrypted Two-Label Classification Using CNN

Yan Li and Yifei Lu

1Nanjing University of Science and Technology, Nanjing 210094, China
2State Key Lab of Mathematical Engineering and Advanced Computing, Wuxi 214215, China

Correspondence should be addressed to Yifei Lu; luyifei@njust.edu.cn

Received 1 January 2021; Revised 26 January 2021; Accepted 17 February 2021; Published 8 March 2021

Copyright © 2021 Yan Li and Yifei Lu. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Due to the increasing variety of encryption protocols and services in the network, the characteristics of the application are very different under different protocols. However, there are very few existing studies on encrypted application classification considering the type of encryption protocols. In order to achieve the refined classification of encrypted applications, this paper proposes an Encrypted Two-Label Classification using CNN (ETCC) method, which can identify both the protocols and the applications. ETCC is a two-stage two-label classification method. The first stage uses the corresponding classifier to classify applications according to the protocol used by the traffic. The second stage uses the corresponding classifier to classify applications according to the protocol used by the traffic. Experimental results show that the ETCC achieves 97.65% accuracy on a public dataset (CICDarknet2020).

1. Introduction

According to the forecast of Cisco’s Annual Internet Report [1], by 2023, the total number of global Internet users will increase from 3.9 billion in 2018 (51% of the global population) to 5.3 billion (66% of the global population), and the number of devices connected to IP networks will reach 29.3 billion, more than three times the global population. As more users and devices connect to the network, the applications will become more and more diversified, and Internet communication methods will become more and more complex, which also make network management more complicated [2]. But if we can identify the application type of network traffic, we can improve the level of network management. For example, many applications occupy a large amount of network bandwidth, causing other applications to operate abnormally. If Internet Service Providers (ISPs) can provide different levels of service quality according to different types of applications, the unfair use of network resources can be solved [3], and the user’s Internet experience will also be better.

On the other hand, in order to meet the needs of users for security and privacy, various encryption technologies are widely used in network communications [4]. Security Socket Layer (SSL), Virtual Private Network (VPN), Secure Shell (SSH), and The Onion Router (Tor) are currently the most common encryption methods [5]. But encryption not only protects users’ privacy but also poses other threats to users. Encryption technologies can help hackers hide their malicious behavior. Network managers need to be able to identify encrypted traffic in a timely manner, so as to quickly and accurately locate attacks on the network, cut off the transmission path, and reduce the harm of malicious behavior to users. Encryption also causes trouble to the IT team. The payload will change after the traffic is encrypted. This change brings additional challenges to the accurate identification of encrypted network protocols and encrypted network applications, resulting in the complexity and difficulty of the traffic analysis and network management [6]. Moreover, even if it can be classified accurately, it is difficult to guarantee real-time performance.

Encryption invalidates many early traffic classification methods, such as port-based classification, entropy-based classification, payload-based classification, and pattern matching-based classification. This is because the port, entropy, payload, and header of network traffic will change...
with encryption [6]. In recent years, machine learning methods have been the most commonly used method for classifying encrypted traffic. This is because encryption is usually only for the payload, and the machine learning method only care about statistical features, not the value of the payload. Hence, machine learning methods are less affected by encryption. This makes machine learning based methods more accurate than other methods.

Most encrypted application classification methods are based on single label. In other words, they directly use the classifier to determine the application type of network traffic. But under different encryption protocols, the characteristics of the application are also different. The encryption protocol mainly has two steps, the initialization of the connection and the transmission of encrypted data. The initialization of the connection is divided into initial handshake, identity verification, and shared key establishment. Because the encryption principles of different encryption protocols are different, these steps are very different, which leads to different representation of the final encrypted traffic [5]. Therefore, if we can classify encrypted applications on the basis of known encryption protocols, we can get more accurate results than single-label classification.

In this paper, we propose an Encrypted Two-Label Classification method, referred to as ETCC, to improve the accuracy of encrypted application classification. ETCC is a two-stage two-label classification method. The two labels are encryption protocol and application. The first stage classifies the protocol used for encrypted traffic. The second stage uses the corresponding classifier to classify applications according to the protocol used by the traffic. The contributions of this paper are summarized as follows:

1. We propose a two-stage two-label scheme called ETCC, which carries out refined application classification according to the encryption protocol used.
2. In the second stage of application classification, encrypted traffic can select the corresponding classifier according to the protocol type, instead of uniformly using the same classifier.
3. Our scheme can identify both the protocol and the application, which can meet various needs.

The rest of this paper is organized as follows. Section 2 introduces some encryption traffic classification methods and some multilabel classification methods. In Section 3, a scheme is proposed to achieve refined applications classification. And some experiments and evaluations are presented in Section 4. Finally, Section 5 concludes our work and proposes some future works.

2. Related Work

In this section, we introduce some methods for classifying encrypted traffic and methods for multilabel classification. These works give bright inspiration for our research.

In the early research, the commonly used methods include port-based classification, entropy-based classification, payload-based classification, and pattern matching-based classification. In the early days of Internet development, every application had a fixed port number assigned by IANA [7]. Therefore, we only need to check the IANA TCP/UDP list to know the type of application. However, with the emergence of technologies such as port confusion and network address translation (NAT), port-based methods have become no longer feasible. Entropy-based methods classify encrypted traffic by extracting geometric features between traffic. Casino et al. [8] propose a method to distinguish encrypted and nonencrypted traffic based on the entropy value. They only analyze a random subset, not the complete network traffic, to ensure real-time performance. The payload-based method can no longer analyze the contents of the package and cannot be used anymore [9]. The method based on pattern matching judges whether it is encrypted traffic and encryption protocol type by checking the header format but cannot further judge the application type. In summary, we need more advanced methods to achieve encrypted traffic classification task.

The most commonly used method is based on machine learning. The differences of these methods are reflected in feature extraction, model selection, and parameter setting. Liu et al. [10] only consider first N packets in a sliding window, which not only reduces the dimension of encrypted traffic characteristics but also reduces the number of data packets in each flow. Similarly, Hasan et al. [11] analyze the first 64 packets to identify Android applications. Finally, they state that most Android applications can be identified through the TCP/IP header. Shen et al. [12] combine the certificated packet length and the first application data size as a unique fingerprint for a given application and then use the second-order Markov chain to classify encryption applications. Cui et al. [13] propose the SPCaps model, which uses capsule neural networks (CapsNet) to learn the spatial features of encrypted traffic. The advantage of this model is that it simultaneously learns the position of the feature in the package and the order between the packages. Ly Vu et al. [14] used time series as an entry point to classify encrypted traffic. Their method is divided into two steps. The first step is to extract behavior patterns based on the time series of packets. The second step is to classify according to the correlation between time series samples. Zeng et al. [15] think more comprehensively. Their scheme not only analyzes spatial features but also analyzes temporal features and coding features. However, these works still ignore the suddenness of network traffic and cannot capture complex nonlinear features. The framework proposed in [16] leverages multifractal feature extraction technology, which can capture the self-similarity of network traffic structure in a wide time range. Because it is always difficult to consider comprehensively when extracting features, Wang et al. [17] took a different approach and directly converted the flow into a picture and put it into the model for classification. Lotfollahi et al. [18] employ CNN and SAE to classify encrypted traffic, respectively. There is no need for an expert to extract features and provide reference for many later studies.

The classification in general scenarios was introduced earlier, but, for specific scenarios, using specific methods can be more efficient. Shen et al. [19] introduce the traffic
classification in Ethereum. Because these flows are all generated on the same platform, it will be more difficult to distinguish. To this end, they study where the existing methods are easy to misclassify and extract features from three aspects: packet length, packet burst, and time series. In order to evaluate quality of experience (QoE) and bring better services to users, Orsolic et al. [20] propose a system for YouTube videos called YouQ. They collect YouTube videos and evaluate the QoE of the videos based on the traffic characteristics of each video session. Similarly, Tarun Mangla et al. [21] evaluated the QoS of encrypted HTTP-based adaptive streaming (HAS) sessions. Anderson et al. [22] analyze TLS encrypted sessions in commercial malware sandboxes and two enterprise networks. They claim that the choice of features has a great impact on performance. In order to monitor and detect specific users, Pierre-Olivier Brissaud et al. [23] propose a scheme for monitoring HTTP/2 communication based on the TLS protocol. This scheme is designed to detect whether the user has performed certain specified operations. The QUIC (Quick UDP Internet Connection) protocol is a new default encrypted Internet communication protocol that provides many improvements to speed up HTTP communication while making it more secure. However, since it is a new type of protocol, the amount of data available is very small. Rezaei et al. [24] propose a semisupervised learning based method that first trains the model with a large amount of unlabeled data and then retrains the model with a small amount of labeled data. For network traffic classification, it greatly reduces the amount of labeled data required.

The studies on multilabel classification are very few. There are two common ways to deal with multilabel classification. Convert the multilabel classification problem into several single-label classification methods, or integrate multilabels into a single label. Grigorios Tsoumakas et al. [25] give a detailed introduction to multilabel classification and compare several classification methods, which provide a lot of guidance for our research. Tien Thanh Nguyen et al. [26] propose a Bayes-based method that not only considers the relationship between labels and features but also considers the relationship between label pairs. Jesse Read et al. [27] constructed a multilabel Hoeffding tree with classifiers at the leaves. Moreover, they create a new set of benchmarks in predictive performance and time complexity. Darshin Kalpesh Shah et al. [28] use RNN and LSTM to classify multilabel text. The performance is significantly better than Logistic Regression and ExtraTrees. Ou Guangjun et al. [29] present a graph convolution networks based multilabel zero-shot learning model to recognize novel categories. Most of the multilabel classification is aimed at the problem of category independence. However, Nadia Ghamrawi et al. [30] study the problem of high label dependence. Jesse Read et al. [31] also study the high dependency between labels. They use a chaining method to model the label relationship. Pengcheng Yang et al. [32] regard the multilabel classification task as a sequence generation problem and used the sequence generation model for classification. Experiments show that this method can effectively capture the correlation between labels. These works help us a lot. Similarly, a two-stage two-label method is proposed in our paper, in which the protocols are classified in the first stage, and then applications are classified in the second stage. The biggest difference between our method and other multilabel classification methods is that our method will select the corresponding classifier for the second stage classification based on the results of the first stage. We achieve refined classification and two-label classification can meet various needs.

### 3. Methodology

In this section, we propose a two-stage, two-label scheme to classify encrypted applications, called ETCC. Our scheme consists of three modules: preprocessing, first label and second label module. They are used to preprocess data, classify protocols, and classify applications, respectively. Figure 1 presents the details.

#### 3.1. Preprocessing Module

This module is used to process raw data and convert them into a format suitable for the input of the classifier.

First, we collect some encrypted traffic and label them with protocols and applications.

Second, we select and extract some features. A flow is a collection of packets with the same IP five-tuple (Source IP, Destination IP, Source Port, Destination Port and Protocol). Because the packets of the same flow are usually the same encryption protocol and application, we process data in units of flows. We use spatial features and temporal features to distinguish encrypted traffic, because these two features are not easily affected by encryption. Spatial features are related to quantity and size. Temporal features are features associated with time series. The specific features are shown in Table 1.

Third, we use the Sequential Floating Forward Selection (SFFS) algorithm [33] to select the most suitable features. We finally selected 41 features about Port, Protocol, Flow Duration, Length of Packet, Flow Bytes/s, Packets/s, Flow IAT, Forward IAT, Backward IAT, Flag Count, and Active Time. Detailed features are shown in Table 1. Through these simplified features, we can get a classifier with better generalization and faster speed.

Finally, we apply Min-Max Scaling [34] to normalize feature to meet the input requirements of supervised classifier and speed up model training. The formula of Min-Max Scaling is shown in

$$X_{\text{norm}} = \frac{X - X_{\text{min}}}{X_{\text{max}} - X_{\text{min}}}$$

where $X_{\text{max}}$ is the maximum value of the sample data, $X_{\text{min}}$ is the minimum value of the sample data, $X$ is the current sample value, and $X_{\text{norm}}$ is the normalized value of the current sample.

After this, the feature values are all mapped to the interval $[0,1]$ and fed into the first label module.

#### 3.2. First Label Module

We leverage this module to classify various encryption protocols into $m$ categories. At first, we choose CNN and LSTM classifiers and test their...
performance, respectively. In the end, we apply CNN, which performs better. The reason for applying CNN is addressed in Section 4.3.

Figure 2 depicts the architecture for CNN. It contains convolution, pooling, flatten, and dense layers. The convolution layer is used to extract different features of the

| Feature                  | Description                                                                 |
|--------------------------|-----------------------------------------------------------------------------|
| IP                       | (Source IP, destination IP)                                                 |
| Port                     | (Source port, destination port)                                             |
| Protocol                 | The protocol of the flow                                                    |
| Flow duration            | The duration of the flow                                                    |
| Length of packet         | Total packets in the [forward, backward] direction                         |
| Flow packet length       | [Total, max, min, mean, std] size of packet in [forward, backward] direction |
| Flow bytes/s             | # of bytes transferred per second                                           |
| Flow packets/s           | # of packets transferred per second                                         |
| Packets/s                | # of [forward, backward] packets per second                                 |
| Flow IAT                 | [Max, min, mean, std] time between two flows                               |
| Forward IAT              | (Total, max, min, mean, std) time between two packets sent in the forward direction |
| Backward IAT             | (Total, max, min, mean, std) time between two packets sent in the backward direction |
| Flags                    | # of times the [PSH, URG] flag was set in packets travelling in the [forward, backward] direction (0 for UDP) |
| Flag count               | # of packets with {FIN, SYN, RST, PSH, ACK, URG, CWE, ECE}                  |
| Header length            | Total bytes used for headers in the [forward, backward] direction          |
| Ratio                    | {Down, up} ratio                                                           |
| Average packet size      | Average size of packet                                                     |
| Segment size avg         | Average size observed in the [flow, forward, backward] direction           |
| Bytes/Bulk avg           | Average number of bytes bulk rate in the [forward, backward] direction     |
| Packet/Bulk avg          | Average number of packets bulk rate in the [forward, backward] direction   |
| Bulk rate avg            | Average number of bulk rate in the [forward, backward] direction           |
| Subflow packets          | The average number of packets in a subflow in the [forward, backward] direction |
| Subflow bytes            | The average number of bytes in a subflow in the [forward, backward] direction |
| Init win bytes           | # of bytes sent in initial window in the [forward, backward] direction     |
| Forward Act data pkts    | # of packets with at least 1 byte of TCP data payload in the forward direction |
| Forward seg size min     | Minimum segment size observed in the forward direction                     |
| Active time              | [Mean, max, min, std] time a flow was active before becoming idle          |
| Idle time                | [Mean, max, min, std] time a flow was idle before becoming active           |
input. However, if several convolution layers are used continuously, the amount of calculation will become very large, and the pooling layer can effectively reduce the amount of calculation through downsampling. Next, the flatten layer will convert the convolved data to one-dimensional and facilitate connection to the dense layer. The dense layer combines all local features into global features at the end to get the classification results.

Figure 3 depicts the architecture for LSTM. The input layer and output layer of LSTM are similar to CNN, but the difference lies in the intermediate calculation process. LSTM cells can learn two pieces of information: new input information and previous memory. This allows LSTM to effectively use historical information so that it can learn long dependencies [35].

After input and calculation, the output layer can get a probability distribution of the flow classification $p_1, p_2, \ldots, p_m$. We define $p_{\text{max}} = \max(p_1, p_2, \ldots, p_m)$ that determines the prediction category.

Finally, protocol types of encrypted traffic are obtained. We sent this $m$ encrypted application traffic to the next module.

### 3.3. Second Label Module

On the basis of known encryption protocols, we leverage this module to further classify encrypted applications into $n$ categories.

Corresponding to the $m$ encryption protocols obtained in the last module, we prepare $m$ classifiers. That is, each protocol corresponds to a classifier. Encrypted traffic selects the corresponding classifier according to its protocol type, and each classifier is only responsible for the application classification of a specific protocol. By using different classifiers for different protocols, we can get more accurate results.

We choose CNN and LSTM in this module. In the end, we apply CNN. The performance of these two algorithms is addressed in Section 4.3.

### 4. Experiment and Evaluation

In this section, we do some experiments to evaluate ETCC and compare it with the state-of-the-art method. We deploy our model on Ubuntu 16.04 OS, equipped with NVIDIA GTX 1050 GPU.

#### 4.1. Dataset Description

Three public datasets CICDarknet2020 [36], ISCXTor2016 [37], and ISCXVPN2016 [38] are used to evaluate ETCC. These datasets include four types of protocols and five types of applications. The four protocols are Tor, Non-Tor, VPN, and Non-VPN. The five applications are chat, FTP, email, audio, and video, as shown in Table 2.

CICDarknet2020 is a complete dataset covering Tor traffic and VPN traffic. The specific quantity of each type of data is shown in Table 3. Since ISCXTor only has Tor traffic and ISCXVPN only has VPN traffic, we mix them together as a dataset, called ISCX-Tor-VPN. In order to eliminate errors caused by data sample selection, ISCX-Tor-VPN uses the same sample quantity as CICDarknet2020. In addition, we set the ratio of the train set to the test set with 4:1.

#### 4.2. Parameter Settings

We deployed experiments for each classifier in each stage.

For the first label module, the structures of the CNN classifier and the LSTM classifier are shown in Figure 4. The dropout layer is used to discard neurons with a certain probability to prevent model overfitting and improve the generalization ability. Furthermore, we set the activation function, loss function, batch size, and epochs with ReLU, categorical_crossentropy, 32, and 15, respectively. For optimizer, the CNN classifier uses SGD, and the LSTM classifier uses Adam.

For the second label module, we have four classifiers to classify encrypted applications. The structures of the CNN classifier and the LSTM classifier are shown in Figure 5. Other parameters are the same as the last module.
4.3. Results and Discussion. In this section, we analyze the performance of ETCC on the two datasets and compare ETCC with the state-of-the-art method.

We evaluate the classification results after the first label module. Figure 6 shows confusion matrices of the results. Rows and columns represent the true category and predicted category. The value represents the probability of a category being classified into each category.

From Figure 6, we find that, under the same model, the results of CICDarknet2020 are better than the results of ISCX-Tor-VPN. This is because the data of CICDarknet2020 is generated under the same network environment, and ISCX-Tor-VPN is a mixed dataset, which makes the distinction between Tor and Non-Tor and between VPN and Non-VPN smaller. For the two classifiers, it is obvious that the results of CNN are better, so we choose CNN as the first stage classifier. In addition, we also find that the easily confused categories are VPN and Tor and Non-VPN and Non-Tor. It is not difficult to understand that there are some similar characteristics between encrypted traffic and non-encrypted traffic.

Tables 4 and 5 show the experimental results of the second label module on the premise that the first label module uses CNN classifier. Accuracy, precision, recall, and
$F_1$ are used to evaluate the scheme. They are defined as follows:

$$\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN},$$

$$\text{precision} = \frac{TP}{TP + FP},$$

$$\text{recall} = \frac{TP}{TP + FN},$$

$$F_1 = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}.$$

For category $X$, TP is the number correctly classified into $X$, TN is the number correctly classified into Not-$X$, FP is the number incorrectly classified into $X$, and FN is the number incorrectly classified into Not-$X$.

As can be seen from the Tables 4 and 5, CNN performs better than LSTM. For CICDarknet2020, except the $F_1$ of Tor, other indicators CNN performs better. For ISCX-Tor-VPN, except the precision of Tor, the precision of Non-VPN, and the $F_1$ of Non-VPN, other indicators CNN performs better. This is because CNN has a better understanding of local features, while LSTM can memorize some context information. In our dataset, the category of a flow has little relationship with the flow before and after it, so CNN
performs better. Therefore, we also choose CNN as the second stage classifier; the worst indicator also exceeds 91.1%.

Tables 6 and 7 show the performance with the second label module and CNN classifier. We find the classification capabilities of Non-Tor and Non-VPN classifiers are better than Tor and VPN classifiers. This proves that encryption makes traffic classification more difficult. Another observation is that the precision of email is very low; this is because the sample size of the email in the dataset is very small. This phenomenon will not occur when the sample size is balanced. Moreover, audio and video achieve the best classification results.

Finally, we compare the results of CICDarknet2020, ISCX-Tor-VPN and the state-of-the-art method [39], as shown in Table 8. The result of CICDarknet2020 is better than that of ISCX-Tor-VPN. The reason is as mentioned earlier; that is, ISCX-Tor-VPN is a mixed dataset, and data is less distinguishable. Moreover, compared with [39], except

Table 4: The performance of CICDarknet2020 with the second label module (%).

| Protocol | CNN | LSTM |
|----------|-----|------|
|          | Accuracy | Precision | Recall | F1 | Accuracy | Precision | Recall | F1 |
| Tor      | 94.3     | 90.6   | 94.8   | 92.7 | 91.8     | 84.3      | 91.4   | 97.7 |
| Non-Tor  | 98.8     | 97.0   | 98.3   | 97.6 | 98.6     | 96.3      | 97.8   | 97   |
| VPN      | 99       | 97.3   | 98     | 97.6 | 98.8     | 96.9      | 97.6   | 97.2 |
| Non-VPN  | 98.5     | 97.3   | 98.5   | 97.9 | 98.3     | 97.1      | 98.2   | 97.6 |
the precision of email and the recall of video, other indicators are improved. Total precision and recall increase by 1% and 1.6%, respectively. In general, our ETCC significantly improves the classification accuracy of encrypted applications through a two-stage two-label method. HX this proves that applications have different characteristics under different protocols, and the classification of applications on the basis of known protocols will result in more accurate results.

5. Conclusion and Future Work

In this paper, to achieve refined classification of encrypted applications, we propose a two-stage two-label scheme. The first stage classifies the protocol used for encrypted traffic. The second stage uses the corresponding classifier to classify applications according to the protocol used by the traffic. The experimental results prove that our scheme is effective and feasible.

Furthermore, we discuss two-label classification in this paper. We will consider more labels in the future and propose more practical solutions. In addition, our method is based on the identification of encryption protocols. Once the traffic uses an unknown encryption protocol, the application classification results will be affected. Therefore, we will consider the use of unknown encryption protocols in our future work.

Data Availability

The datasets used in this paper are mainly obtained through the website https://www.unb.ca/cic/datasets/darknet2020.html; https://www.unb.ca/cic/datasets/tor.html; https://www.unb.ca/cic/datasets/vpn.html. The raw/processed data required to reproduce these findings cannot be shared at this time as the data also form part of an ongoing study.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.
Acknowledgments

This research was supported in part by the National Natural Science Foundation of China under Grant no. 61702267, Jiangsu Planned Projects for Postdoctoral Research Funds, and in part supported by the Open Project Program of the State Key Laboratory of Mathematical Engineering and Advanced Computing.

References

[1] U. Cisco, Cisco Annual Internet Report (2018–2023) White Paper, 2020.
[2] A. Jakalan, J. Gong, Q. Su, X. Hu, and A. M. S. Abdelgder, “Social relationship discovery of IP addresses in the managed IP networks by observing traffic at network boundary,” Computer Networks, vol. 100, pp. 12–27, 2016.
[3] J. Zhang, X. Chen, Y. Xiang, W. Zhou, and J. Wu, “Robust network traffic classification,” IEEE/ACM Transactions on Networking, vol. 23, no. 4, pp. 1257–1270, 2014.
[4] M. H. Mazhar and Z. Shafiq, “Real-time video quality of experience monitoring for https and quic,” in Proceedings of the IEEE INFOCOM 2018-IEEE Conference on Computer Communications, pp. 1331–1339, Honolulu, HI, USA, April 2018.
[5] P. Velan, M. Čermák, P. Čeleda, and M. Drašar, “A survey of methods for encrypted traffic classification and analysis,” International Journal of Network Management, vol. 25, no. 5, pp. 355–374, 2015.
[6] P. Wang, S. Li, F. Ye, Z. Wang, and M. Zhang, “PacketCGAN: exploratory study of class imbalance for encrypted traffic classification using CGAN,” in Proceedings of the ICC 2020-2020 IEEE International Conference on Communications (ICC), pp. 1–7, San Jose, USA, June 2020.
[7] 2020 Service name and transport protocol port number registry https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml.
[8] F. Casino, K.-K. R. Choo, and C. Patsakis, “HEDGE: efficient traffic classification of encrypted and compressed packets,” IEEE Transactions on Information Forensics and Security, vol. 14, no. 11, pp. 2916–2926, 2019.
[9] M. Finsterbusch, C. Richter, E. Rocha, J.-A. Muller, and K. Hanssgen, “A survey of payload-based traffic classification approaches,” IEEE Communications Surveys & Tutorials, vol. 16, no. 2, pp. 1135–1156, 2013.
[10] Y. Liu, J. Chen, P. Chang, and X. Yen, “A novel algorithm for encrypted traffic classification based on sliding window of flow’s first N packets,” in Proceedings of the 2017 2nd IEEE International Conference on Computational Intelligence and Applications (ICCIA), pp. 463–470, Beijing, China, September 2017.
[11] H. F. Alan and J. Kaur, “Can Android applications be identified using only TCP/IP headers of their launch time traffic?” in Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks, pp. 61–66, Darmstadt, Germany, July 2016.
[12] M. Shen, M. Wei, L. Zhu, and M. Wang, “Classification of encrypted traffic with second-order Markov chains and application attribute bigrams,” IEEE Transactions on Information Forensics and Security, vol. 12, no. 8, pp. 1830–1843, 2017.
[13] S. Cui, B. Jiang, Z. Cai, Z. Lu, S. Liu, and J. Liu, “A session-packets-based encrypted traffic classification using capsule neural networks,” in Proceedings of the 2019 IEEE 21st International Conference on High Performance Computing and Communications, pp. 429–436, Zhangjiajie, China, August 2019.
[14] L. Vu et al., “Time Series Analysis for Encrypted Traffic Classification: A Deep Learning Approach,” in Proceedings of the 2018 18th International Symposium on Communications and Information Technologies (ISCIT), pp. 121–126, Austin, Texas, USA, September 2018.
[15] Y. Zeng, H. Gu, W. Wei, and Y. Guo, “$Deep-Full-Range$: a deep learning based network encrypted traffic classification and intrusion detection framework,” IEEE Access, vol. 7, pp. 45182–45190, 2019.
[16] E. Areström and N. Carlsson, “Early online classification of encrypted traffic streams using multi-fractal features,” in Proceedings of the IEEE INFOCOM 2019-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), pp. 84–89, Paris, France, May 2019.
[17] W. Wang, M. Zhu, J. Wang, X. Zeng, and Z. Yang, “End-to-end encrypted traffic classification with one-dimensional convolution neural networks,” in Proceedings of the 2017 IEEE International Conference on Intelligence and Security Informatics (ISI), pp. 43–48, Beijing, China, July 2017.
[18] M. Lotfollahi, M. J. Siaovoshani, R. S. H. Zade, and M. Saberian, “Deep packet: a novel approach for encrypted traffic classification using deep learning,” Soft Computing, vol. 24, no. 3, pp. 1999–2012, 2020.
[19] M. Shen, J. Zhang, L. Zhu, K. Xu, X. Du, and Y. Liu, “Encrypted traffic classification of decentralized applications on ethereum using feature fusion,” in Proceedings of the 2019 IEEE/ACM 27th International Symposium on Quality of Service (IWQoS), pp. 1–10, Passau, Germany, June 2019.
[20] I. Orsolic, D. Pevec, M. Suznjevic, and L. Skorin-Kapov, “A machine learning approach to classifying YouTube QoE based on encrypted network traffic,” Multimedia Tools and Applications, vol. 76, no. 21, pp. 22267–22301, 2017.
[21] T. Mangla, E. Halepovic, M. Ammar, and E. Zegura, “Using session modeling to estimate HTTP-based video QoE metrics from encrypted network traffic,” IEEE Transactions on Network and Service Management, vol. 16, no. 3, pp. 1086–1099, 2019.
[22] B. Anderson and D. McGrew, “Machine learning for encrypted malware traffic classification: accounting for noisy labels and non-stationarity,” in Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1723–1732, New York, NY, August 2017.
[23] P.-O. Brissaud, J. Francois, I. Chrisment, T. Cheole, and O. Bettan, “Transparent and service-agnostic monitoring of encrypted web traffic,” IEEE Transactions on Network and Service Management, vol. 16, no. 3, pp. 842–856, 2019.
[24] S. Rezaei and X. Liu, “How to Achieve High Classification Accuracy with Just a Few Labels: A Semi-supervised Approach Using Sampled Packets,” 2018, https://arxiv.org/abs/1812.09761.
[25] G. Tsoumakas and I. Katakis, “Multi-label classification,” IEEE Transactions on Knowledge and Data Engineering, vol. 25, no. 5, pp. 84–89, Paris, France, May 2013.
[26] T. Mangla, E. Halepovic, M. Ammar, and E. Zegura, “Using session modeling to estimate HTTP-based video QoE metrics from encrypted network traffic,” IEEE Transactions on Network and Service Management, vol. 16, no. 3, pp. 1086–1099, 2019.
[27] B. Anderson and D. McGrew, “Machine learning for encrypted malware traffic classification: accounting for noisy labels and non-stationarity,” in Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1723–1732, New York, NY, August 2017.
[28] P.-O. Brissaud, J. Francois, I. Chrisment, T. Cheole, and O. Bettan, “Transparent and service-agnostic monitoring of encrypted web traffic,” IEEE Transactions on Network and Service Management, vol. 16, no. 3, pp. 842–856, 2019.
[29] S. Rezaei and X. Liu, “How to Achieve High Classification Accuracy with Just a Few Labels: A Semi-supervised Approach Using Sampled Packets,” 2018, https://arxiv.org/abs/1812.09761.
[30] G. Tsoumakas and I. Katakis, “Multi-label classification,” IEEE Transactions on Knowledge and Data Engineering, vol. 25, no. 5, pp. 84–89, Paris, France, May 2013.
[31] T. Mangla, E. Halepovic, M. Ammar, and E. Zegura, “Using session modeling to estimate HTTP-based video QoE metrics from encrypted network traffic,” IEEE Transactions on Network and Service Management, vol. 16, no. 3, pp. 1086–1099, 2019.
[32] B. Anderson and D. McGrew, “Machine learning for encrypted malware traffic classification: accounting for noisy labels and non-stationarity,” in Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1723–1732, New York, NY, August 2017.
[33] P.-O. Brissaud, J. Francois, I. Chrisment, T. Cheole, and O. Bettan, “Transparent and service-agnostic monitoring of encrypted web traffic,” IEEE Transactions on Network and Service Management, vol. 16, no. 3, pp. 842–856, 2019.
[34] S. Rezaei and X. Liu, “How to Achieve High Classification Accuracy with Just a Few Labels: A Semi-supervised Approach Using Sampled Packets,” 2018, https://arxiv.org/abs/1812.09761.
[35] G. Tsoumakas and I. Katakis, “Multi-label classification,” IEEE Transactions on Knowledge and Data Engineering, vol. 25, no. 5, pp. 84–89, Paris, France, May 2013.
[28] D. K. Shah, M. A. Sanghvi, R. P. Mehta, P. S. Shah, and A. Singh, "Multilabel Toxic Comment Classification Using Supervised Machine Learning Algorithms," in Proceedings of the Machine Learning for Predictive Analysis, pp. 23–32, Springer, 2020.

[29] G. Ou, G. Yu, C. Domeniconi, X. Lu, and X. Zhang, "Multi-label zero-shot learning with graph convolutional networks," Neural Networks, vol. 132, pp. 333–341, 2020.

[30] N. Ghamrawi and A. McCallum, "Collective multi-label classification," in Proceedings of the 14th ACM International Conference on Information and Knowledge Management, pp. 195–200, Bremen, Germany, October 2005.

[31] J. Read, B. Pfahringer, G. Holmes, and E. Frank, "Classifier chains for multi-label classification," in Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 254–269, Bled, Slovenia, September 2009.

[32] P. Yang, X. Sun, W. Li, S. Ma, W. Wu, and H. Wang, "SGM: Sequence Generation Model for Multi-Label Classification," 2018, https://arxiv.org/abs/1806.04822.

[33] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," Journal of Machine Learning Research, vol. 3, pp. 1157–1182, 2003.

[34] S. Patro and K. K. Sahu, "Normalization: a preprocessing stage," 2015, https://arxiv.org/abs/1503.06462.

[35] Y. Li, Y. Lu, and L. S. T. M.-B. A., "DDoS Detection Approach Combining LSTM and Bayes," in Proceedings of the 2019 Seventh International Conference on Advanced Cloud and Big Data (CBD), pp. 180–185, Suzhou, China, September 2019.

[36] A. H. Lashkari, "DDarknet: A Contemporary Approach to Detect and Characterize the Darknet Traffic Using Deep Image Learning," in Proceedings of the Presented at the International Conference on Communication and Network Security, Tokyo, Japan, November 2020.

[37] A. H. Lashkari, G. Draper-Gil, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of tor traffic using time based features," in Proceedings of the 3rd International Conference on Information Systems Security and Privacy, pp. 253–262, Porto, Portugal, February 2017.

[38] G. Draper-Gil, A. H. Lashkari, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of encrypted and vpn traffic using time-related," in Proceedings of the 2nd International Conference on Information Systems Security and Privacy (ICISSP), pp. 407–414, Rome, Italy, February 2016.

[39] Z. Zou, J. Ge, H. Zheng, Y. Wu, C. Han, and Z. Yao, "Encrypted traffic classification with a convolutional long short-term memory neural network," in Proceedings of the 2018 IEEE 20th International Conference on High Performance Computing and Communications, pp. 329–334, Exeter, UK, June 2018.