Fast and Robust Online Traffic Classification Supporting Unseen Applications

Yue Gu*, Dan Li, Kaihui Gao, Yang Cheng
Department of Computer Science and Technology, Tsinghua University, Beijing, China
Tsinghua Shenzhen International Graduate School, Shenzhen, China
*guy18@mails.tsinghua.edu.cn

GLOBECOM 2021 - 2021 IEEE Global Communications Conference | 978-1-7281-8104-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/GLOBECOM46510.2021.9685631

***Abstract*—Online traffic classification is a fundamental toolkit in network management, such as QoS and network security. The speed and generalization ability of online classification are two requirements that need to be satisfied simultaneously. However, existing methods may suffer from generalization degradation on the traffic with unseen applications which are constantly emerging in the network, due to the feature distribution drift (FDD) caused by their non-robust feature engineering approaches.**

Based on Deep Metric Learning which can restrict the distances between samples explicitly and clustering algorithm which can learn multiple clusters within each category, this paper presents Robot, a fast and robust online traffic classification system. At its core, Robot leverages two building blocks to classify high-speed traffic flows: 1) For fast classification, Fast model classifies traffic and detects FDD samples simultaneously based on only one packet. 2) For robust classification, once FDD samples are detected, a flow collector will be triggered to collect flows and then Robust model, a multi-center model, will further identify them based on the hybrid of packet-level and flow-level features. Our comprehensive experiments demonstrate that Robot can achieve comparative classification speed and better generalization ability on the mixed traffic datasets with seen and unseen applications (with the FDD detection accuracy of up to 85.8% and with nearly 10% improvement in classification accuracy), compared with the state-of-the-art methods.

***Index Terms*—Online Traffic Classification, Deep Metric Learning**

I. Introduction

With the continuous development of the Internet, various applications are emerging in the network. Different applications generate different categories of network traffic that take on discriminated characteristics. Traffic classification (TC), which aims to identify the traffic categories (*e.g.* VoIP, Streaming, File Transfer, Chat, *etc.*) according to distinguishing characteristics [1], is crucial for the Internet Service Provider (ISP). On the one hand, TC is the first step to guarantee QoS to provide differentiated services for different traffic categories [1]. On the other hand, TC is vital for network security to identify malicious traffic [2]. To ensure the effectiveness of QoS and malicious traffic detection, ISPs have to deploy the online traffic classification (OTC) model on the edge devices in the backbone network, such as gateways. In addition to the need to identify encrypted traffic due to the widespread use of encryption techniques in network applications [3], OTC has two other practical issues to be considered.

First, the flow completion time of latency-sensitive applications (*e.g.* online search, social networking, *etc.*) directly affects the user experience and even the cost-efficiency of the ISPs. And most flows from these applications are short [4]. Therefore, to meet the performance requirements of these applications, OTC has to ensure the *classification speed* to be as fast as possible.

Second, while the traffic categories may be fixed (6 categories in this paper) in the online environment, unseen applications are constantly emerging [5]. It is impossible to obtain prior knowledge of all the applications running in the network. Therefore, OTC should also ensure the *generalization ability* on the traffic with unseen applications.

In recent years, some prior works try to classify the encrypted traffic based on machine learning techniques. Flow-based methods classify traffic based on flow statistics [6], [7] or time-series features [8], [9] and show promising accuracy. They have to collect several packets or the entire flows to ensure classification accuracy, affecting the classification speed which is vital for OTC. For fast classification, many works propose to classify traffic based on the raw content (header and/or payload) of one single packet [1], [10] instead of entire flows. Packet-based methods can reduce the classification latency from ms- to μs-level and show competitive accuracy. However, both the flow-based and packet-based methods may suffer from generalization degradation under the traffic with unseen applications.

Based on our analysis (more details in §II-B), the reason why the generalization ability may degrade is that the feature distribution may shift on the traffic of unseen applications, caused by the non-robust feature engineering methods used by existing works. The softmax loss function used in these feature engineering methods assumes that each category has only one cluster/center [11]. However, in reality, each category may contain multiple local clusters, since the traffic that belongs to the same category but is generated from different applications may have differentiated traffic characteristics.

We argue that an effective OTC model should show the satisfying performance of classification speed and generalization ability simultaneously. Packet-based methods are the first choices to meet the demand for fast classification. However,

This work was supported by the National Key Research and Development Program of China under Grant Grant 2019YFB1802600, the Research and Development Program in Key Areas of Guangdong Province under Grant 2018B010113001, and the National Natural Science Foundation of China under Grant 61772305.

it's hard to conduct better feature engineering to avoid the occurrence of feature distribution drift (FDD) and improve the generalization ability based on limited packet-level features. Therefore, we classify traffic based on packet-level features and meanwhile detect samples where FDD happens (FDD samples). Once FDD samples are detected, a robust model is needed to further identify them based on the hybrid of packet-level and flow-level features to obtain more accurate classification results.

In order to detect FDD samples, distance-based methods should be leveraged to gather samples from the same category. Then the samples that deviate from the original distribution can be detected. Besides, for purpose of improving generalization ability, the robust model should have the potential of learning multiple clusters/centers for each category to reduce the probability of the occurrence of FDD.

Clustering is a distance-based algorithm and can realize multiple clusters in each category (*e.g.* pre-defined k clusters for Kmeans). However, because of the lack of guidance with ground truth, clustering often shows unsatisfying performance on classification accuracy.

Recently, Deep Metric Learning (DML) shows the promising ability to restrict the distances between samples explicitly [12]. With DML, samples from the same categories can be gathered together within a circle, inspiring accurate FDD detection. Samples from different categories can be well distinguished, ensuring accurate classification.

Based on these inspirations, we propose Robot, a fast and robust online encrypted traffic classification system, which takes advantage of DML and clustering. Robot has two main building blocks: Fast model and Robust model. Fast model can detect FDD samples while classifying traffic based on packet-level features. The detected FDD samples will be further identified by Robust model, a multi-center model, based on the hybrid of packet-level and flow-level features. For the two building blocks, we carefully design different loss functions for DML-based feature extractors.

We conduct extensive experiments on two reconstructed encrypted datasets based on "ISCX VPN-nonVPN dataset". We evaluate the overall performance of Robot and assess the effectiveness of Fast model and Robust model separately. Robot shows comparative classification speed (with μs-level classification latency for most traffic) and better generalization ability on the mixed traffic datasets with both seen and unseen applications (with $7\% - 10\%$ improvement in classification accuracy), compared with the state-of-the-art methods. Fast model can achieve up to 85.8% FDD detection accuracy on the mixed datasets (while clustering model has the accuracy of 53.4%), and Robust model can achieve $15\% - 20\%$ improvement in classification accuracy on the FDD samples.

The main contributions of the paper are as follows.

1) We conduct a comprehensive study on the problem of generalization degradation in existing traffic classification methods, find the FDD phenomenon, and propose a novel FDD detection method.
2) We propose a compound end-to-end solution that leverages packet-based method and hybrid features-based multi-center method to achieve fast yet accurate online traffic classification.
3) We demonstrate the effectiveness of Robot with experiments under the mixed traffic with seen and unseen applications.

II. Background and Motivation

A. Background of Online Traffic Classification

Traffic classification (TC) is fundamental to modern network management, especially QoS and network security. To meet the demands of QoS, the ISPs have to deploy the classification model online. Online traffic classification (OTC) has two practical requirements. First, the online classification process should be as fast as possible to guarantee QoS for latency-sensitive applications. Second, the traffic classifier should show satisfying and consistent generalization on the traffic with unseen applications which are constantly emerging in the network.

B. Limitation of Existing Online Traffic Classification

Recently, many works focus on the traffic classification problem. The flow-based methods classify traffic based on flow-level features (*e.g.* time-series features or flow statistics) and obtain high classification accuracy at the expense of classification speed due to the need of collecting enough packets. Packet-based methods take packet content (raw bytes in header or payload) of one single packet as model input and achieve fast and accurate classification.

However, the above methods suffer from generalization degradation on the traffic with unseen applications. We evaluate two typical methods (FS-Net [8], a state-of-the-art flow-based method, and Deep Packet [1], a state-of-the-art packet-based method) and find that: (1) FS-Net can reach $96\% - 99\%$ accuracy on the traffic of applications already seen in the training set, while its accuracy may drop to $20\% - 39\%$ on the traffic of unseen applications. Similarly, (2) Deep Packet achieves $96\% - 98\%$ accuracy on the traffic of seen applications, while its accuracy may drop to $12\% - 33\%$ on the traffic of unseen applications.

In order to analyze the reason why generalization degradation happens, we visualize the feature distribution, extracted by Deep Packet, as shown in Fig. 1. The feature distribution has shifted on the traffic of unseen applications (Test Set AU), compared with the distribution on the traffic of seen applications (Test Set AS). And the similar phenomenon can be observed in FS-Net. We analyze that the feature engineering methods which are based on softmax loss function used by existing works only learn one single cluster/center within each category [11], leading to the feature distribution drift (FDD) on the traffic of unseen applications.

According to the analysis above, in order to make classification process as fast as possible, OTC should leverage packet-based classification model to ensure the classification speed. Meanwhile, in order to maintain consistent generalization on

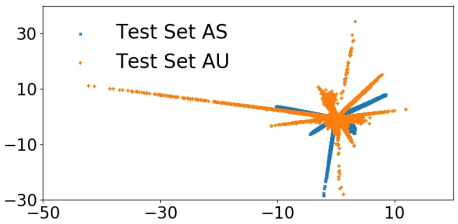


Fig. 1. The Feature Distribution Drift (FDD) happens on the traffic of unseen applications (Test Set AU). Test Set AS contains the traffic of applications already seen in the training set.

the unseen traffic, FDD detection should be considered in OTC, and robuster feature engineering should be used to classify the FDD samples. Therefore, we have to propose a fast and accurate FDD detection model and a robust classifier.

C. Deep Metric Learning

Deep Metric Learning (DML) aims to learn an embedding representation of the data that ensures the distance between samples from the same class (positive pairs) close and samples from different classes (negative pairs) far [12]. Many DML methods use pair-based loss, such as contrastive loss [13], triplet loss [14], *etc.* Because of these effective loss functions, DML has been widely used in face recognition, pedestrian recognition, and clustering [15].

Contrastive loss takes sample pairs as input and enables the distance to be as close to zero as possible for positive pairs and to be as far to a pre-defined margin m as possible for negative pairs. Specifically, the loss is written as follows:

$$\mathcal{L}\left(\{D_{ij}\}\right) = \sum_{y_{ij}=1} D_{ij} + \sum_{y_{ij}=0} \left[m - D_{ij}\right]_+ \tag{1}$$

where D_{ij} represents the distance between sample i and j, $y_{ij} = 1$ represents that sample i and j are positive pairs, $y_{ij} = 0$ represents that they are negative pairs, $[]+$ is the hinge function and m is the pre-defined parameter.

Compared with traditional Deep Learning methods, DML can restrict the distances between samples explicitly [12], which is suitable for accurate FDD detection and traffic classification.

III. Robot Design

In this section, we first overview the design of Robot. Then we show the design details of each part in Robot.

A. Design Overview

Fig. 2 shows the overview of the whole online classification process. Robot contains two main building blocks, namely, Fast model and Robust model. Fast model is a packet-based classifier that can detect FDD samples while classifying. Robust model is a multi-center classifier that aims to accurately classify the detected FDD samples based on the hybrid of flow-level and packet-level features. Once the FDD samples are detected by Fast model, a flow collector will be triggered to collect multiple packets (from the flows that FDD samples belong to) to obtain the hybrid features. Both Fast model and Robust model contain the processes of DML-based feature extraction and clustering, while the loss functions used in DML are totally different.

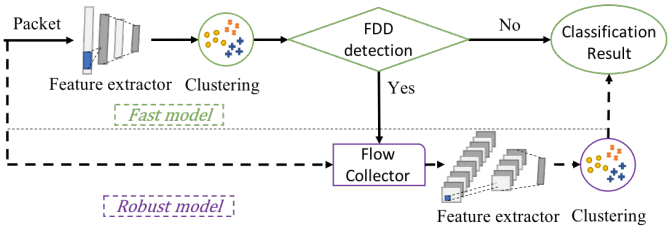


Fig. 2. The overview of Robot.

B. Fast Model

For purpose of fast classification, Fast model classifies traffic based on raw bytes of one single packet. Considering that payload can be futile for classification due to encryption, we take raw bytes of packet headers as input (set IP address and port to be zero to avoid overfitting). As shown in Fig. 2, Fast model first extracts features from input by a DML-based feature extractor and then performs classification and FDD detection simultaneously by clustering.

Due to the fact that distance-based methods (*e.g.* clustering) are effective in the detection tasks, such as anomaly detection [16], *etc.*, we select the clustering algorithm as the base model of FDD detection. The FDD detection process is to determine whether the distances between the sample and all the cluster centers exceed a given FDD detection threshold (D). To this end, it's vital to set a rational threshold to achieve accurate FDD detection. Thus, we aim to find a suitable embedding space, in which positive samples can be gathered within a circle, based on which the threshold D can be decided.

We use a DML-based feature extractor to learn an embedding space as described above. Since 1D-CNNs are an ideal choice for the packet-based traffic classification task [1], we construct the feature extractor in Fast model with 1D-CNNs. In order to explore the effects of different constrained distances between positive pairs on the FDD detection accuracy, instead of applying contrastive loss (introduced in §II-C) to DML directly, we modify the contrastive loss by introducing another pre-defined parameter a. The modified loss function is shown in the following formula:

$$\mathcal{L}\left(\{D_{ij}\}\right) = \sum_{y_{ij}=1} \left[D_{ij} - a\right]_+ + \sum_{y_{ij}=0} \left[m - D_{ij}\right]_+ \tag{2}$$

where a confines the distances between positive sample pairs and can be greater than or equal to zero. When $a = 0$, (2) is the contrastive loss.

After DML-based feature extraction process trained by (2), Kmeans++ [17] will be used as our clustering model to perform classification and FDD detection. Our clustering model is obtained offline with a large number of labeled samples. Each cluster is labeled by the category with the most samples. Each category has one cluster and a is the diameter of each cluster. In the classification process, the label corresponding to the nearest cluster center is taken as the classification result. For FDD detection, we compute distances between each sample and all the cluster centers. Once all the distances exceed D, this sample will be regarded as an FDD sample.

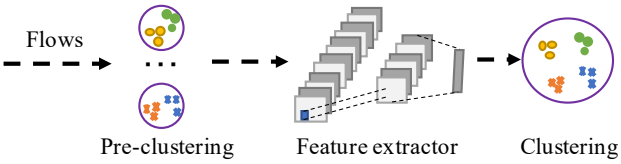


Fig. 3. The training process of Robust model.

C. Robust Model

For purpose of improving generalization ability on the FDD samples, Robust model takes steps from two perspectives: feature enhancement and model enhancement.

When FDD samples are detected, a flow collector will be triggered to collect 20 packets from the flows that FDD samples belong to. For feature enhancement, Robust model uses hybrid features as model input. Besides the raw bytes of packet header, Robust model also takes flow-level features (the sequences of packet interval and packet size from each flow) into account.

For model enhancement, the robust model should have the potential of learning multiple clusters/centers within each category to reduce the probability of the occurrence of FDD. Clustering is such an algorithm that can realize multiple clusters in each category (*e.g.* Kmeans: the number of clusters relies on the pre-defined k). However, because of the lack of guidance with ground truth, clustering often shows unsatisfying performance on classification accuracy. Thus, Robust model should combine clustering model with a more effective feature engineering method to help discriminate samples from different categories.

The training process of Robust model contains three stages, as shown in Fig. 3:

1) Pre-clustering: In order to increase the number of centers within each category, we pre-cluster the flows by Kmeans++ into k clusters for each category based on the hybrid features. k is a pre-defined parameter. In Fig. 3, $k = 2$.

2) Feature Extraction: Then to ensure the classification accuracy, we leverage DML to discriminate flows from different clusters and gather flows from the same clusters. We construct the DML-based feature extractor in Robust model with 2D-CNNs. The feature extractor is trained by (3), as shown below.

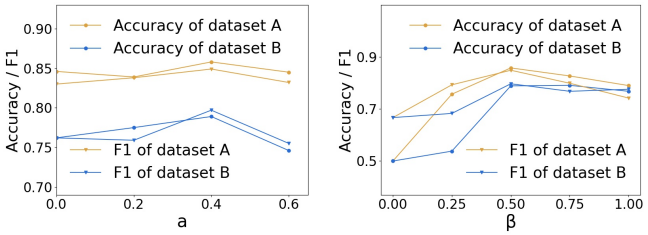
$$\mathcal{L}\left(\{D_{ij}\}\right) = \sum_{y_{ij}=2} \left[D_{ij} - a\right]_+ + \sum_{y_{ij}=1} \left[b - D_{ij}\right]_+ + \sum_{y_{ij}=0} \left[m - D_{ij}\right]_+ \tag{3}$$

where a confines the distances between positive pairs from the same cluster, b confines the distances between positive pairs from different clusters and m confines the distances between negative pairs.

3) Clustering: After feature extraction, we use Kmeans++ again to classify these flows. The model acquisition and classification processes of Kmeans++ are the similar with Fast model.

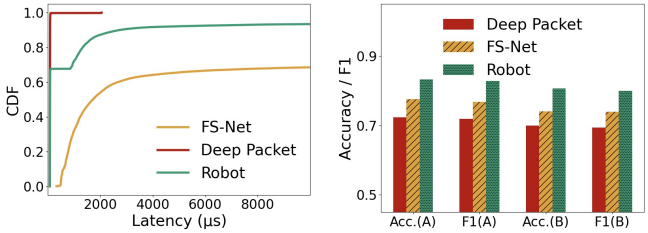
IV. Evaluation

This section demonstrates the effectiveness of Robot, where we try to answer the following questions: (1) How fast the system can react to a given traffic flow, *i.e.* classification speed, and (2) whether the system can identify flows from unseen applications correctly as expected, *i.e.* generalization ability. We conduct several comparative experiments.



(a) Different settings of a (b) Different settings of β

Fig. 4. FDD detection accuracy of different parameter settings.



(a) Classification latency (CDF) (b) Classification accuracy

Fig. 5. Performance of Robot and comparison methods.

A. Dataset for Evaluation

We construct two encrypted datasets based on "ISCX VPN-nonVPN dataset" [7], including 17 applications: Skype, Youtube, *etc.*, and 6 categories: VoIP, chat, *etc.* We refer to the datasets as dataset A and dataset B, respectively. Both datasets include one training set and two test sets. One of the test sets contains the traffic of applications already seen in the training set (test set AS for dataset A and test set BS for dataset B). The other one contains the unseen applications (test set AU for dataset A and test set BU for dataset B). The unseen applications are selected differently in both datasets. For dataset A, we elect Skype, Facebook, Hangouts, SFTPs to be the unseen applications. For dataset B, we elect voipbuster, FTPs, SCP, Skype as the unseen applications.

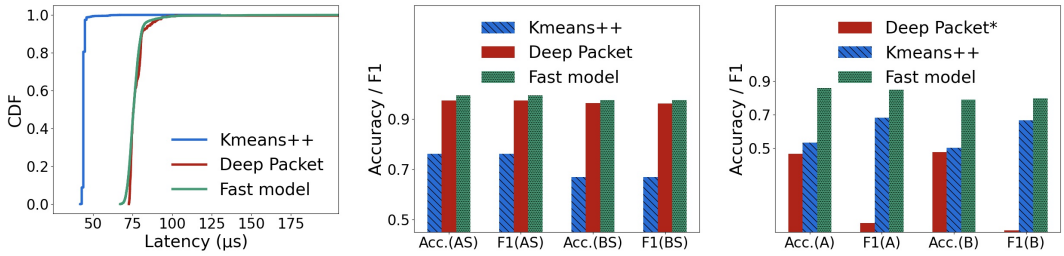
B. Experiment Setting

1) Experiment Environment: We run the experiments on a server, which contains two Intel® Xeon® CPU E5-2630 v3 (56 cores in all), 128GB RAM memory, a 2TB disk and an NIVIDIA Tesla K40C GPU. We build the neural network models using TensorFlow.

2) Experiment Metrics: Accuracy, F1-score, and classification latency are applied to evaluate the performance. Classification latency is the sum of time for both feature extraction and classification.

3) Comparison Methods:

- Deep Packet [1]: a packet-based traffic classification method, which takes raw bytes of packet content as model input and uses 1D-CNNs to classify the packets.
- FS-Net [8]: a flow-based traffic classification method, which learns the representative features from the packet sequences and classifies flows correctly.



(a) Classification latency (CDF) on the test sets with seen applications

(b) Classification accuracy on the test sets with seen applications

(c) FDD detection accuracy on the mixed test sets with both seen and unseen applications

Fig. 6. Performance of Fast model and comparison methods.

C. Parameter Selection

We first evaluate the effects of different values of parameter a on the FDD detection accuracy. For each dataset, we mix the two test sets with the ratio of 1:1 (The number of samples from seen applications is the same as that from unseen applications). The results on two datasets are shown in Fig. 4(a). The best performance ($accuracy = 85.8\%, F1 = 84.9\%$ for dataset A, $accuracy = 78.9\%, F1 = 79.7\%$ for dataset B) happens when $a = 0.4$.

We also conduct experiments on different settings of FDD detection threshold D. Since a is the diameter of the cluster, D is related to a as talked earlier in §III-B. We define $D = \beta \cdot a$ and set different values of β in our experiments. As seen in Fig. 4(b), when $\beta = 0.5$ ($D = a/2$, *i.e.* radius), the FDD detection model shows the best performance. Therefore, we set $a = 0.4$ and $\beta = 0.5$ in the following experiments.

D. Overall Performance

To verify the classification speed and generalization ability of Robot, we conduct several experiments on the mixed datasets, which simulate the real-world trace containing both seen and unseen applications. The mixed datasets are constructed by fusing two test sets in both datasets with the ratio of 2:1 (The ratio of the number of samples from seen applications to that from unseen applications is 2:1). We compare Robot with two state-of-the-art methods: Deep Packet (packet-based) and FS-Net (flow-based).

As shown in Fig. 5(a), Deep Packet shows the best performance with μs-level classification latency. Since it takes longer time to obtain flow-level features, FS-Net shows the worst performance with ms-level (even longer) classification latency. Robot is somewhere in between FS-Net and Deep Packet: most samples are classified correctly in a short time, only the detected FDD samples prolong the classification time.

As shown in Fig. 5(b), Robot achieves the best generalization performance compared with baselines, with an accuracy of 83%(F1 of 82%) for dataset A and accuracy of 80.6% (F1 of 80%) for dataset B. Furthermore, we note that FS-Net shows better performance than Deep Packet, which verifies our insight that the information brought by only packet-level features is limited. Compared with Deep Packet, we note that although Robust model sacrifices the classification speed, it classifies the FDD samples more accurately.

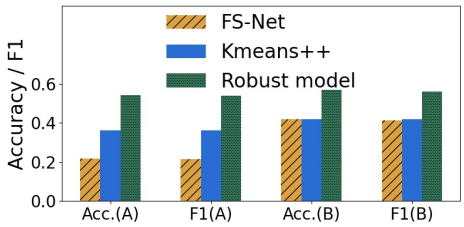


Fig. 7. Classification accuracy of Robust model and comparison methods on the detected FDD samples from two mixed datasets.

E. Performance of Fast Model

In order to verify the performance of Fast model, we assess the classification accuracy and classification latency on the test set AS, BS and evaluate the FDD detection accuracy on the mixed datasets (the same with §IV-C). Since clustering is a straightforward FDD detection method, we also take Kmeans++ as one of the baselines. We also combine packet-based feature engineering with a clustering model to detect FDD samples. We take the middle layer output of Deep Packet as features and name this combination method Deep Packet*. The results of contrast experiments with Kmeans++ and Deep Packet* are shown in Fig. 6

1) Classification latency: From Fig. 6(a), we can draw the conclusion that although the additional feature engineering methods introduced in Deep Packet and Fast model make the classification latency longer than Kmeans++, the μs-level latency is acceptable in the online environment.

2) Classification accuracy: From Fig. 6(b), we can see that since unsupervised learning lacks the guidance with ground truth, the classification accuracy of Kmeans++ is the worst among the three methods. Both Deep Packet and Fast model obtain good performance on the classification accuracy due to their effective feature engineering methods.

3) Accuracy of FDD detection: As shown in Fig. 6(c), Fast model shows the highest FDD detection accuracy among the three methods. Fast model achieves 85.8% FDD detection accuracy on dataset A and nearly 80% accuracy on dataset B. Both Kmeans++ and Deep Packet* perform badly on the FDD detection. By Kmeans++, the distances between the cluster centers are very close, leading to a very small value of D, resulting in a lot of false positive samples. By Deep Packet*, the distances between the cluster centers are very far, leading to a very large value of D, thus resulting in a number of false negative samples.

F. Performance of Robust Model

In order to verify the generalization ability of Robust model, we conduct experiments on the FDD samples. From Fig. 7, our approach is effective in improving the generalization ability to classify the FDD samples. By contrast with two other comparison methods: FS-Net and Kmeans++, Robust model achieves the best results with the accuracy of 54% on dataset A and 57.8% on dataset B ($15\% - 20\%$ higher than other methods, which is already a big breakthrough).

Although both FS-Net and Kmeans++ have poor generalization ability, it is still worth mentioning that Kmeans++ shows better performance than FS-Net due to its ability of learning multiple clusters for each category, even without additional feature engineering.

V. Related Work

There have been numerous works in the field of network traffic classification.

A. Flow-based Traffic Classification Methods

Flow-based methods [6], [8], [9] classify traffic based on flow-level features, including time-series features and flow statistics. Specifically, time-series features refer to packet length sequences or packet interval sequences of all or first n packets from each flow. Flow statistics are calculated based on the time-series features, including minimum value, maximum value, mean, variance, and percentiles, *etc.*

AppScanner [6] leverages flow statistics and adopts Support Vector Regression algorithm (SVR) and Random Forest algorithm (RF) to achieve high classification accuracy. Considering that the expert experience is required to compute and select statistical features, making the process of feature extraction prone to human error, FS-Net [8] proposes to use a multi-layer encoder-decoder structure and a reconstruction mechanism based on raw packet sequences, *i.e.* time-series features, without any additional manual feature extraction process and achieves nearly 99% classification accuracy. [9] also uses time-series features as the input of the combination model of Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) and achieves comparable classification accuracy.

B. Packet-based Traffic Classification Methods

Lately, many works propose to classify traffic based on packet content (raw bytes in header or payload). Deep Packet [1] is the first representative work that proposes to classify traffic based on the raw bytes of packets. Each packet is taken as the input of the classification model: One-Dimensional Convolutional Neural Network (1DCNN) or Sparse Automatic Encoding machine (SAE). Nearly 98% classification accuracy can be achieved by Deep Packet. BSNN [10] transforms raw bytes of payload into segments which are then fed into two-level attention encoders that are made of RNN unit with attention layers, and show its great performance in its paper.

VI. Conclusion

In this paper, we firstly conduct a comprehensive study on the problem of generalization degradation which is ignored by existing traffic classification works. Then, this paper presents Robot, a fast and robust online traffic classification system based on Deep Metric Learning and clustering. Robot leverages both packet-based method and hybrid features-based multi-center method to achieve fast yet accurate online traffic classification. Finally, we demonstrate the effectiveness of Robot with several experiments. Overall and break-down evaluations show that Robot can keep comparative classification speed and better generalization ability under the traffic with unseen applications.

References

[1] M. Lotfollahi, M. J. Siavoshani, R. S. H. Zade *et al.*, "Deep packet: A novel approach for encrypted traffic classification using deep learning," *Soft Computing*, vol. 24, no. 3, pp. 1999–2012, 2020.

[2] W. Wang, M. Zhu, X. Zeng *et al.*, "Malware traffic classification using convolutional neural network for representation learning," in *2017 IEEE ICOIN Conference*. IEEE, 2017, pp. 712–717.

[3] W. Wang, M. Zhu, J. Wang, X. Zeng, and Z. Yang, "End-to-end encrypted traffic classification with one-dimensional convolution neural networks," in *2017 IEEE ISI Conference*. IEEE, 2017, pp. 43–48.

[4] L. Quan and J. Heidemann, "On the characteristics and reasons of long-lived internet flows," in *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*, 2010, pp. 444–450.

[5] T. van Ede, R. Bortolameotti, A. Continella, J. Ren, D. J. Dubois, M. Lindorfer, D. Choffnes, M. van Steen, and A. Peter, "Flowprint: Semi-supervised mobile-app fingerprinting on encrypted network traffic," in *Network and Distributed System Security Symposium, NDSS 2020*. Internet Society, 2020.

[6] V. F. Taylor, R. Spolaor, M. Conti, and I. Martinovic, "Appscanner: Automatic fingerprinting of smartphone apps from encrypted network traffic," in *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2016, pp. 439–454.

[7] 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)*, 2016, pp. 407–414.

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

[9] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, and J. Lloret, "Network traffic classifier with convolutional and recurrent neural networks for internet of things," *IEEE Access*, vol. 5, pp. 18 042–18 050, 2017.

[10] R. Li, X. Xiao, S. Ni, H. Zheng, and S. Xia, "Byte segment neural network for network traffic classification," in *2018 IEEE/ACM 26th IWQoS Conference*. IEEE, 2018, pp. 1–10.

[11] Q. Qian, L. Shang, B. Sun, J. Hu, H. Li, and R. Jin, "Softtriple loss: Deep metric learning without triplet sampling," in *IEEE/CVF ICCV Conference*, 2019, pp. 6450–6458.

[12] F. Xu, W. Zhang, Y. Cheng, and W. Chu, "Metric learning with equidistant and equidistributed triplet-based loss for product image search," in *Proceedings of The Web Conference 2020*, 2020, pp. 57–65.

[13] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *2005 IEEE CVPR Conference*, vol. 1. IEEE, 2005, pp. 539–546.

[14] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE CVPR Conference*, 2015, pp. 815–823.

[15] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell, "Distance metric learning with application to clustering with side-information," in *NIPS*, vol. 15, no. 505–512. Citeseer, 2002, p. 12.

[16] G. Münz, S. Li, and G. Carle, "Traffic anomaly detection using k-means clustering," in *GI/ITG Workshop MMBnet*, 2007, pp. 13–14.

[17] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," Stanford, Tech. Rep., 2006.