FLAG: Flow Representation Generator based on Self-supervised Learning for Encrypted Traffic Classification

Wenting Wei
Xidian University

Tianjie Ju✉
Xidian University
Shanghai Jiaotong University

Han Liao
Southwest University
Jianghan University

Weike Zhao
Xidian University

Huaxi Gu
Xidian University

ABSTRACT

Due to its excellent ability in learning features from large scale raw data, deep learning (DL) has attracted much attention for encrypted traffic classification. However, most DL-based traffic classifiers usually rely on enormous labeled samples. Motivated by this, we investigate a self-supervised traffic classifier (FLAG) without sacrifice of identification accuracy, only depending on small labeled traffic samples and highly available unlabeled traffic samples. Specifically, focusing on local short-term characteristics of traffic, we design a preprocessing algorithm, termed as N-phrase Extration, to convert unlabeled raw traffic dataset into sequences of high-frequency phrases as input of Bidirectional Encoder. On account of their significance, potential timing characteristics from input sequences are mined by Bidirectional Encoder and embedded into robust representations with distributed vectors to enhance classifier's performance significantly. Our comprehensive experiments indicate FLAG can achieve 98.65% in 100% of dataset and 98.07% in 10% of dataset in terms of true positive rate in UNB ISCX VPN-nonVPN dataset, which are better than p-FP, FS-Net and Deep Packet.

1 INTRODUCTION

Network traffic classification is playing an increasingly significant role in network optimization applications. An accurate classifier for network traffic will contribute to traffic engineering to improve network performance and quality-of-service. However, encryption procedure turns raw flows into a pseudo-random-like format [12], which poses considerable challenge for traffic classification.

Existing traffic classification methods can be divided into three categories: rule-based methods, statistical learning methods and deep learning (DL) based methods. With the increasing adoption of dynamic assignments in new application, rule-based methods do not work well due to their dependence on deterministic rules and expert knowledge [17][16]. Especially with the development of traffic encryption technologies, it is extremely difficult to construct rules for encrypted traffic by domain knowledge. Traditional statistical learning methods rely heavily on statistical or time series features extracted by humans, but how to design an effective traffic feature set is still an open question.

Inspired by recent improvement of artificial intelligence, DL provides a way to traffic identification [2][20]. DL methods can extract high-level features automatically for traffic classification by learning the raw input layer by layer, without constructing features based on domain knowledge [3]. With sufficient datasets and computing resources, DL-based encrypted traffic classifiers employing common DL algorithms, such as SAE and CNN [1], can already achieve good performance. However, as users' awareness of privacy protection increases, it is always time-consuming to obtain sufficient data with ground-truth label for training DL models in most network scenarios, resulting in a poor performance and unexpected behaviors of some apps with the increase of training time [19]. Thus, a robust DL-based classifier without the demand of large-quantity labeled data needs to be designed urgently.

In this paper, FLAG is proposed to classify encrypted traffic. It can be divided into four modules: Data Processing, Phrase Expanding, Bidirectional Encoder and Finetuning. Specifically, Raw flows in pcap format are transfered into bytecodes during Data Processing. N-phrase Extraction is used in Phrase Expanding to reconstruct training samples in

APNet 2021, June 24–25, 2021, Shenzhen, China, China

ACM ISBN 978-1-4503-8587-9/21/06…$15.00
https://doi.org/10.1145/3469393.3469394

bytecodes and extract meaningful phrases as new features from unlabeled dataset. It traverses unlabeled flows to count frequency of each N-phrase (N = 1, 2, 3) and convert raw flows using Greedy Algorithm. Then, BERT [4] is used in Bidirectional Encoder to embed traffic data into dense vector form for pre-training, in which Transformer [18] Encoder digs deep and robust sequence features. In addition, dynamic masking technology [10] is applied to improve Bidirectional Encoder, thereby enhancing the performance of trained vector. Finally, softmax layer is added in Finetuning module to fine-tune the whole model for supervised learning, so that the whole model can be applied to our classification task.

Compared to previous work, small labeled samples and highly available unlabeled samples are used for training with significant improvements in performance. We summarize our contributions as follows.

(1) FLAG is an end-to-end traffic classification solution for encrypted traffic, which independently learns vital features and contextual information from traffic, without requiring expert knowledge for feature extraction.

(2) Self-supervised learning is used to extract informative phrases and generate robust and high-quality flow representations, so that FLAG can achieve high accuracy with small labeled samples and sufficient unlabeled samples.

(3) Compared with other work, we carried out encrypted traffic classification work on UNB ISCX VPN-nonVPN dataset and achieved breakthroughs in various indicators regardless of whether labeled data is sufficient.

The rest of the paper is organized as follows. Section 2 describes the preliminaries. Section 3 introduces the model we proposed, namely FLAG. Section 4 conducts comprehensive experiments between FLAG and current models. We have carried out detailed experiments and analysis on various indicators. Section 5 summarizes the paper and introduces possible future work directions.

2 PRELIMINARIES

In this section, we first define the traffic classification problem we need to solve. Then we briefly introduce the Transformer framework and self-attention mechanism.

2.1 Problem Definition

Traffic classification is mainly to identify different categories of network traffic. Assume that there are N samples and C classes in our dataset. The kth sample can be represented as $x_k = \{l_1^{(k)}, l_2^{(k)}, ..., l_{n_k}^{(k)}\}$, where n_k is the length of the kth sequence and $l_p^{(k)}$ is the packet value in time step i. We are going to build an end-to-end model $\psi(x_p)$ to predict a label $\hat{A}_k$ that is exactly the real label A_p. Our dataset can be represented as $D = \{(x_1, A_1), (x_2, A_2), ..., (x_N, A_N)\}$.

Our proposal is first pretrained on a large number of unlabeled flows and subsequently fine-tuned using labeled flows.

2.2 Transformer Network

The Transformer network[18] is one of the most popular neural networks to model sequences, which consists of Encoder and Decoder. Transformer uses the attention mechanism to replace LSTM [5] and Positional Encoding to solve the word order problem. It has made huge breakthroughs in Natural Language Processing (NLP). Transformer is an Encoder-and-Decoder architecture. The encoder consists of $N = 6$ individual layers. Each layer has two sub-layers, one is multi-head mechanism layer and another is position-wise fully connected feed-forward layer. Eesidual connection and layer normalization are connected between each two sub-layers. That is, the output of sub-layers is LayerNorm(x + SubLayer(x)). The decoder also consists of $N = 6$ individual layers. However, there is a third sub-layer in decoder which performs multi-head attention over the output of the encoder stack.

2.3 Attention Mechanism

A simple encoder-and-decoder model won't focus on different features when predicting results, which means each feature has the same contribution. Attention[18] mechanism can focus on some important raw features in order to generate high-quality sequences.

The Attention mechanism is essentially an addressing process. Given a task-related "Query" vector, it calculates the Attention Value by calculating the attention distribution of vector "Key" and attaching it to vector "Value". Actually, an attention function is a set of key-value pairs. To specific, the attention function of Transformer network is "Multi-Head Attention", which contains several attention heads to allow each head to focus on different features. The formula of multi-head attention mechanism is as follows:

$$Q_i = QW_i^Q, K_i = KW_i^K, V_i = VW_i^V, i = 1, ..., n \quad (1)$$

$$head_i = softmax(\frac{Q_i K_i^T}{\sqrt{d_k}})V_i, i = 1, ..., n \quad (2)$$

$$MultiHead(Q, K, V) = Concat(head_1, ..., head_n)W^0 \quad (3)$$

where n denotes the number of heads, W is the weight vector of training models, and Q, K and V are the vector "Query", "Key" and "Value" respectively.

3 PROPOSED METHOD

In this section, we will introduce our proposed end-to-end model FLAG in detail. The architecture of FLAG can be seen in Figure 1, which can be divided into four phases.

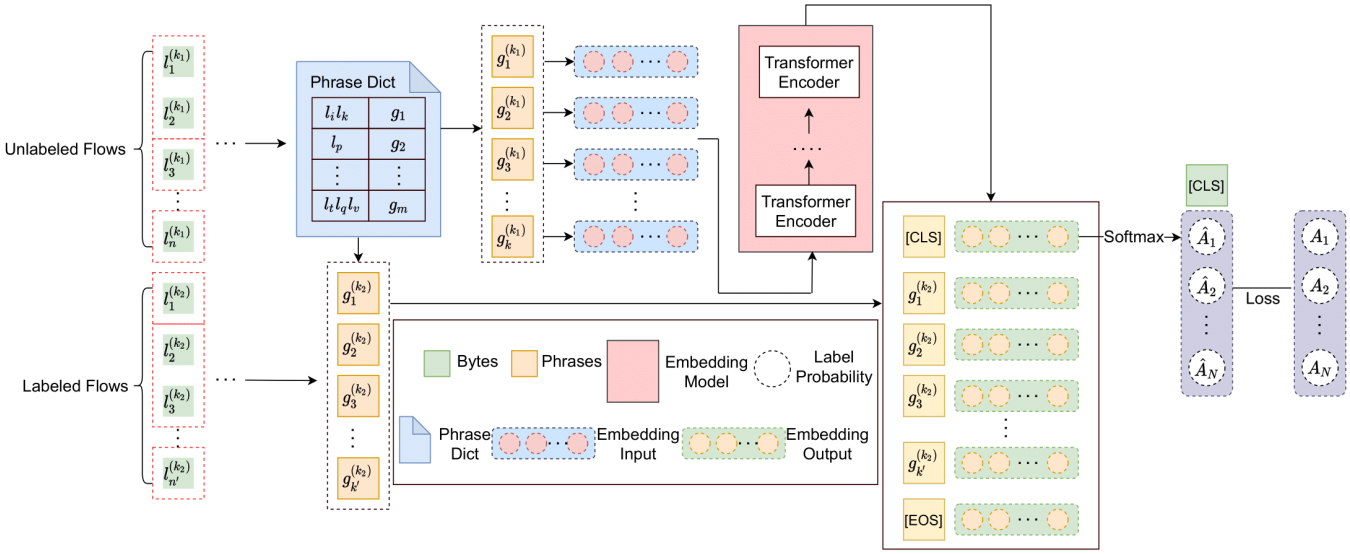


Figure 1: The architecture of FLAG.

3.1 Data Processing

The dataset we use (ISCXVPN2016) [7] is captured at data-link layer. It consists of labeled network traffic, including full packets in pcap format. Considering that data preprocessing is not the focus of our paper, our preprocessing method is similar to (Mohammad et al. 2020) [12] and we make certain adjustments. The specific process is as follows.

1. Ethernet headers are removed since they are useless for classification task.

2. Zeros are injected to the end of UDP segment's headers to make them equal length with TCP headers.

3. IP is masked and irrelevent packets are removed.

4. Raw packets are converted into bytes. Unlike (Mohammad et al. 2020)[12], we do not limit the length of bytecodes.

3.2 Phrase Expanding

For Bidirectional Encoder, we need to generate a phrase dictionary W corresponding to the flows. A simple method is to use bytecodes in raw flows as a dictionary. However, this dictionary only has 256 kinds of phrases. With such a small number of phrases, it is difficult for our Bidirectional Encoder to learn enough useful representations. Therefore, N-phrase Extraction Algorithm is proposed to expand the phrase dictionary.

N-phrase Extraction can extract informative byte phrases throuth unlabeled dataset. It is used to convert raw flows into sequences composed of n-phrases ($n = 1, 2, 3$) where n is the length of the phrase in bytes. Consider the whole dataset, the frequency of a phrase g_t is $f(g_t)$. A phrase dictionary $D = \{g_1, g_2, ..., g_m\}$ contains all of these n-phrases, where m is a parameter to control the length of our dictionary. To specific, our task is to discover m informative phrases to fill in our dictionary. With such a dictionary, we can build a mapping which map raw flows to sequences composed of n-phrases. By traversing the unlabeled flow in unlabeled dataset, the m phrases with the highest frequency can be added to the dictionary. Each phrase in the dictionary has an index.

For example, the raw sequence of x_k can be represented as $x_k = [l_1^{(k)}, l_2^{(k)}, ..., l_i^{(k)}, l_{i+1}^{(k)}, l_{i+2}^{(k)}, ..., l_{n_k}^{(k)}]$. We use the greedy algorithm to determine whether it is a phrase in the dictionary from left to right. If 3-phrase sequence $g_t = [l_1^{(k)}, l_2^{(k)}, l_3^{(k)}]$ is in the dictionary, then the raw sequence is transfered to $y_k = [g_t, l_4^{(k)}, l_5^{(k)}, ..., l_{n_k}^{(k)}]$. If not, we will check 2-phrase and 1-phrase sequence sequentially. The priority of each phrase is $3 - phrase > 2 - phrase > 1 - phrase$. The detailed steps of proposed N-phrase Extraction are shown in Algorithm 1

Finally, N-phrase Extraction is selected for this paper. Compared with bytecodes, N-phrase Extraction takes into account some important 2-phrase sequences and 3-phrase sequences, which considers more effective contextual information and makes the converted sequences more like sentences in natural language.

3.3 Bidirectional Encoder

Embedding is an important concept in Natual Language Processing. It is actually a function which maps original vector space into a dense and low-dimension (semantic) space. We use Bidirectional Encoder [4] to better embed flows. Our encoder is a stack of Transformer encoder layers [18] which

Algorithm 1 N-phrase Extraction

Input: Origin flow set $X = [x_1, x_2, ..., x_i, ..., x_N]$.
Output: Transfered flow set $Y = [y_1, y_2, ..., y_i, ..., y_N]$.

```
Initialize the phrase dictionary D.
// Calculate the frequency of each phrase.
for i = 1 to N do
    for j = 1 to n_k − 2 do
        f_[l_j^(i)] ← f_[l_j^(i)] + 1
        f_[l_j^(i), l_{j+1}^(i)] ← f_[l_j^(i), l_{j+1}^(i)] + 1
        f_[l_j^(i), l_{j+1}^(i), l_{j+2}^(i)] ← f_[l_j^(i), l_{j+1}^(i), l_{j+2}^(i)] + 1
    end for
end for
// Generate the phrase dictionary D = {g_1, g_2, ..., g_m}.
D ← reverseSort(D)[1 : m]
// Transfer X to Y according to the dictionary D.
for i = 1 to N do
    j ← 0, y_i ← [] // y_i is a list to store transfered flow.
    while j < length(x_i) − 2 do
        // Using Greedy Algorithm, the priority is 3 − phrase > 2 − phrase > 1 − phrase.
        if [l_j^(i), l_{j+1}^(i), l_{j+2}^(i)] ∈ D then
            y_i ← y_i + [l_j^(i), l_{j+1}^(i), l_{j+2}^(i)], j ← j + 3
        else
            if [l_j^(i), l_{j+1}^(i)] ∈ D then
                y_i ← y_i + [l_j^(i), l_{j+1}^(i)], j ← j + 2
            else
                y_i ← y_i + [l_j^(i)], j ← j + 1
            end if
        end if
    end while
end for
```

consists of multiple self-attention "heads" [14]. We use a Transformer architecture with 12 layers. Each block uses 6 self-attention heads and 768 hidden dimensions.

We feed the processed unlabeled flows into Bidirectional Encoder with special tokens: $\{[\text{CLS}], g_1, g_2, ..., g_n, [\text{EOS}]\}$ where n is a parameter that controls the maximum sequence length during training, [CLS] is the representation of this sequence and [EOS] marks the end of the sequence.

Bidirectional Encoder uses dynamic masking technology [10] as its training task. Dynamic masking technology makes our encoder learn more from different masking scheme to enrich encoder's represent ablity. A random sample of phrases in the input sequence is selected to replace with a special token [MASK], another phrase or the phrase itself. To avoid using the same mask for each training instance in every epoch, training data is duplicated 10 times so that each sequence is masked in 10 different ways over the 40 epochs of training.

After masking, the pre-training model will try to predict the phrase that is masked or replaced. It compares predicted value with true value to calculate the loss of the model, and uses backpropagation method to update the weights of the model. Finally, thanks to the attention mechanism, the vector [CLS] can learn a robust representation of the entire flow. We can apply the vector [CLS] to the next phase.

3.4 Classification Layer

After Bidirectional Encoder, the vector [CLS] is feed into the classification layer for supervised learning. The previous phases have enabled the model to learn enough important characteristics of raw flows. Therefore, in the Classification Layer, we do not use complex neural networks (such as RNN [21]) but softmax layer to fine-tune our model, which can avoid the risk of overfitting and reduce the complexity of our model. The softmax function is used to generate the label distribution of ith sample:

$$P(x_i) = \frac{\exp(x_i)}{\sum_{j=1}^{N} \exp(x_j)} (i = 1, ..., N) \tag{4}$$

where N is the total number of classes. We can use the output of the softmax layer to represent the probability of encrypted traffic belonging to each category and compare it with the true value. Then cross entropy function is used to calculate loss and backpropagate it to update weights.

4 EVALUATION

In this section, we have a brief introduction on our experimental settings. Then, we present our experiment comparison results to evaluate the efficiency of our proposed model.

4.1 Experimental Settings

In this paper, we utilize a representative dataset of real-world traffic in ISCX which is rich enough in diversity and quantity [8]. According to (Mohammad, 2020)[12], it is divided into 12 traffic categories. They are Chat, Email, File Transfer, Streaming, Torrent, VoIP, VPN: Chat, VPN: Email, VPN: File Transfer, VPN: Streaming, VPN: Torrent and VPN: VoIP.

We test our FLAG using Python 3.7 and implement with TensorFlow. All of simulations were carried out on an Intel(R) Core(TM) i7-7700K CPU @ 4.20GHz processor with 64 GB of memory. As for Bidirectional Encoder, we use a Transformer architecture with 6 layers. Each block consists of 6 self-attention heads and 768 hidden dimensions.

To better evaluate performance of our proposed model, True Positive Rate (TPR) and False Positive Rate (FPR) are used as indicators referring to [15][12]. TPR stands for the

percentage of the flows that are rightly classified among total flows. FPR is the percentage of the flows that are wrongly classified among total flows. Fractional combination of TPR and FPR (FTF) shows the overall accuracy. These metrics are described mathematically as follows.

$$TPR_{AVE} = \frac{1}{N}\sum_{i=0}^{C} TPR_i * FlN_i, \tag{5}$$

$$FPR_{AVE} = \frac{1}{N}\sum_{i=0}^{C} FPR_i * FlN_i, \tag{6}$$

$$FTF = \sum_{i=0}^{C} w_i \frac{TPR_i}{1 + FPR_i}, \tag{7}$$

$$w_i = \frac{FlN_i}{N}, \tag{8}$$

where FlN_i is the number of the ith flows class and w_i is the radio between FlN_i and the total flows number N.

4.2 Results Comparison

Consider that most traffic in reality is unlabeled, 40% of the dataset has the label removed for self-supervised learning and 60% of the dataset for supervised learning. Labeled data can be divided into training set, validation set and test set according to the ratio of 6:2:2. To prevent overfitting during training, the model with the highest accuracy on verification set is selected, so as to ensure that the test set would not be touched during training process. Finally, results are shown in the test set.

Several works had been conducted on encrypted traffic classification. We compare our model with FS-Net [9], Deep packet [12] and p-FP [13], where Deep packet employed two deep neural network structures, namely stacked autoencoder (SAE) and convolution neural network (CNN) and p-FP used a basic algorithm, multi-Layer perceptron (MLP) classifier. In our work, we adopt 1) CBOW [11] and Text-CNN [6] and 2) FLAG which is expounded detailedly in section 3.

The main reason for the excellent performance of current deep learning algorithms is that they utilize large amounts of labeled data for training. Therefore, in order to verify the performance of each algorithm in the case of sufficient or insufficient labeled data, we conducted two groups of controlled experiments. The first experiment trains the whole labeled data (60% of the total dataset). And for another, we randomly sample 10% of our labeled data, which is 6% of the total dataset. Comparison results are listed in Table 1.

N-phrase Extract algorithm is innovatively utilized in CBOW and FLAG by converting raw flows into semantic sequences before pretraining. To investigate its effects on encrypted traffic classification, we conduct an ablation study,

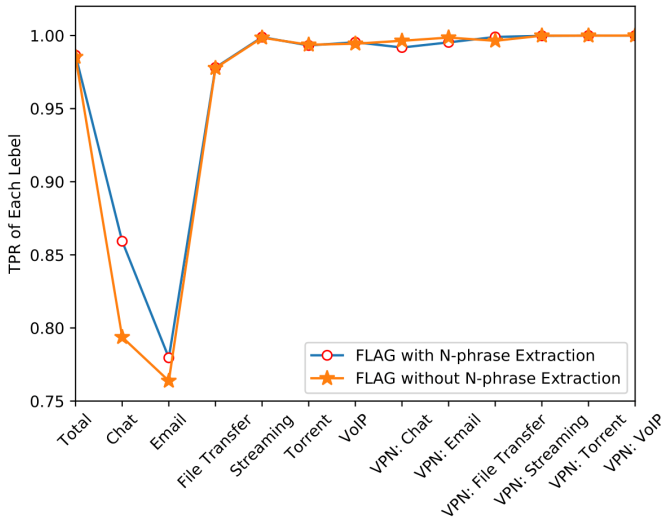
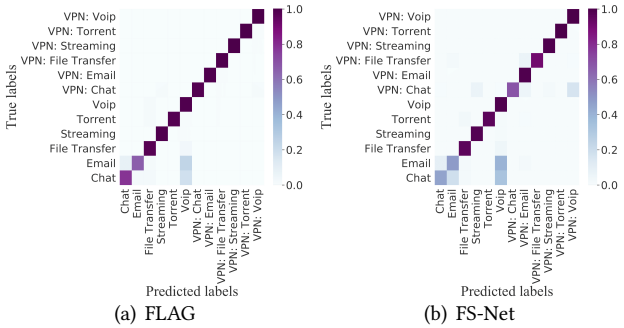


Figure 2: Comparison of Bidirectional Encoder using N-phrase Extraction and raw flows.



(a) FLAG (b) FS-Net

Figure 3: Normalized confusion matrices using (a) FLAG (b) FS-net model on 10% labeled dataset.

which use the same model except the use of N-phrase Extract algorithm. Fig.2 shows the ablation experiment comparision results. Experiments show that N-phrase Extraction can slightly improve the overall performance of FLAG (98.65% with N-phrase Extraction in TPR and 98.53% with bytecodes in TPR), and has obvious advantages in identifying categories with less data such as Chat and Email.

To conduct a further study on FLAG performance, normalized confusion matrixs are made to compare with FS-net on 10% labeled dataset, which is shown in Fig. 3(a) - Fig. 3(b).

Finally, we further analyze the inference delay of FLAG. In our experiment (a NVIDIA GeForce GTX 1080 Ti), it takes 323.37s to infer 1% datasets (5218 flows in total) on FLAG, It means that each flow takes an averge of 0.062s, which is acceptable as the inference delay.

Table 1: Evaluation Result

	FS-Net		Deep Packet: SAE		Deep Packet: CNN		*p*-FP		CBOW		FLAG	
	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR	TPR	FPR
	Data volume: 100%											
Chat	0.2719	**0.0003**	0.7203	0.0016	0.7458	0.0009	0.5767	0.0005	0.7904	0.0005	**0.8593**	0.0005
Email	0.3678	0.0016	0.5841	0.0004	0.6088	0.0003	0.6804	0.0013	0.6738	**0.0002**	**0.7796**	**0.0002**
File Transfer	0.9588	0.0065	0.9769	0.0054	0.9755	0.0042	0.9701	0.0019	0.9720	**0.0013**	**0.9781**	0.0023
Streaming	0.9514	0.0004	0.9805	0.0003	0.9933	0.0001	0.9817	0.0002	0.9962	**0.0000**	**0.9989**	**0.0000**
Torrent	0.9704	0.0004	0.9635	**0.0000**	0.9842	0.0000	0.9702	0.0002	0.9896	**0.0000**	**0.9932**	**0.0000**
VoIP	0.9902	0.0371	0.9886	0.0192	0.9931	0.0207	0.9957	0.0250	**0.9977**	0.0230	0.9954	**0.0173**
VPN: Chat	0.5329	0.0002	0.6252	0.0003	0.9660	0.0002	0.8189	0.0001	0.9838	**0.0000**	**0.9918**	**0.0000**
VPN: Email	0.7114	**0.0000**	0.6349	**0.0000**	0.9833	0.0000	0.9270	0.0001	0.9859	**0.0000**	**0.9953**	**0.0000**
VPN: File Transfer	0.9634	0.0012	0.9861	0.0012	0.9920	**0.0001**	0.9793	0.0008	0.9974	**0.0001**	**0.9990**	**0.0001**
VPN: Streaming	0.9880	**0.0000**	0.9814	**0.0000**	0.9983	0.0001	0.9949	0.0000	0.9993	**0.0000**	**0.9998**	**0.0000**
VPN: Torrent	0.9835	0.0001	0.9976	0.0003	0.9991	0.0000	0.9939	0.0003	0.9998	**0.0000**	**1.0000**	**0.0000**
VPN: VoIP	0.9968	0.0033	0.9975	0.0021	0.9982	0.0002	0.9965	0.0011	0.9994	**0.0000**	**0.9999**	0.0001
Total	0.9648	0.0032	0.9782	0.0020	0.9821	0.0016	0.9780	0.0020	0.9832	0.0015	**0.9865**	**0.0012**
FTF	0.9496		0.9692		0.9734		0.9688		0.9751		**0.9797**	
	Data volume: 10%											
Chat	0.4519	0.0012	0.5312	0.0021	0.5846	0.0021	0.4706	0.0011	0.7419	0.0024	**0.7791**	**0.0009**
Email	0.4793	0.0021	0.2962	0.0004	0.4692	0.0009	0.4938	0.0011	0.2198	**0.0001**	**0.6642**	0.0005
File Transfer	0.9607	0.0032	0.9699	0.0079	0.9637	0.0043	0.9639	0.0031	0.9626	**0.0008**	**0.9713**	0.0043
Streaming	0.9815	0.0006	0.9601	0.0007	0.9730	0.0004	0.9646	0.0004	0.9891	**0.0000**	**0.9923**	0.0002
Torrent	0.9552	**0.0000**	0.9564	0.0002	0.9673	0.0001	0.9732	0.0003	**0.9871**	0.0001	0.9824	0.0001
Voip	0.9926	0.0327	0.9871	0.0287	0.9894	0.0312	0.9923	0.0322	**0.9981**	0.0302	0.9916	**0.0226**
VPN: Chat	0.6904	**0.0000**	0.6877	0.0012	0.7818	0.0006	0.6465	0.0007	0.9696	0.0003	**0.9792**	0.0001
VPN: Email	0.9124	**0.0000**	0.5051	0.0001	0.8104	0.0001	0.6493	0.0002	0.9810	0.0001	**0.9905**	**0.0000**
VPN: File Transfer	0.9886	0.0010	0.9327	0.0012	0.9658	0.0007	0.9642	0.0015	0.9882	0.0002	**0.9972**	**0.0001**
VPN: Streaming	0.9932	0.0003	0.9942	0.0004	0.9932	0.0003	0.9937	**0.0000**	0.9959	**0.0000**	**0.9971**	**0.0000**
VPN: Torrent	0.9970	**0.0000**	0.9743	0.0002	0.9984	0.0001	0.9853	0.0004	0.9994	**0.0000**	**0.9997**	**0.0000**
VPN: Voip	0.9967	0.0010	0.9866	0.0016	0.9929	0.0010	0.9908	0.0018	**0.9988**	**0.0002**	0.9990	**0.0002**
Total	0.9705	0.0027	0.9689	0.0028	0.9711	0.0026	0.9701	0.0027	0.9762	0.0022	**0.9807**	**0.0000**
FTF	0.9583		0.9560		0.9590		0.9579		0.9659		**0.9713**	

Among these experiments, we can draw following conclusions from the results.

Firstly, N-phrase Extraction (used in CBOW and FLAG) can indeed generate important representations in raw flows. Ablation experiments indicate that N-phrase Extraction is an effective phrase expanding algorithm which considers contextual information of each byte, so that the sequence after phrase expansion has stronger timing characteristics and semantic richness. Secondly, FLAG always outperforms FS-Net with about 0.5% - 1.5% improvement and CBOW with about 0.4% improvement in TPR, which proves that Bidirectional Encoder can better obtain sequence representation. Finally, It is not difficult to find from the normalized confusion matrices that FLAG's ability to recognize "Chat", "Email" and "VPN Chat" is significantly better than FS-Net. However, FLAG still often misjudges "Chat" and "Email" as "VoIP".

5 CONCLUSION

In this paper, a self-supervised encrypted traffic classifier, termed as FLAG, is proposed in the small labeled sample case. Thanks to pre-training with unlabeled samples, FLAG can reduce the overhead of labeling data while ensuring high classification performances. N-phrase Extraction is design to extract phrases with more contextual information than bytes from raw traffic data. In addition, Bidirectional Encoder is used to generate robust and high-quality vectors for fine-tuning. Experiments showed that FLAG can always achieve the best performance compared to several current models, regardless of the lack of labeled data.

In the future, more comprehensive ablation experiments will be conducted to prove each component (N-phrase Extraction, Bidirectional Encoder, etc.) contributes to the success of FLAG while the delta between FLAG and other type of methods like rule-based methods will be studied in detail. In addition, we will focus on proving the importance of each component theoretically, not just intuitively.

REFERENCES

[1] Giuseppe Aceto, Domenico Ciuonzo, Antonio Montieri, and Antonio Pescapé. 2018. Mobile Encrypted Traffic Classification Using Deep Learning. In *IEEE/ACM Network Traffic Measurement and Analysis*

Conference (TMA'18).

[2] G. Aceto, D. Ciuonzo, A. Montieri, and A Pescapé. 2020. Toward Effective Mobile Encrypted Traffic Classification through Deep Learning. *Neurocomputing* 409 (2020).

[3] G. Aceto, D. Ciuonzo, A. Montieri, and A Pescapé. 2021. DISTILLER: Encrypted traffic classification via multimodal multitask deep learning. *Journal of Network and Computer Applications* (2021).

[4] Jacob Devlin, Ming Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. (2018).

[5] S Hochreiter and J Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[6] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. *Eprint Arxiv* (2014).

[7] Arash Habibi Lashkari, Gerard Draper-Gil, Mohammad Saiful Islam Mamun, and Ali A. Ghorbani. 2016. Characterization of Encrypted and VPN Traffic Using Time-Related Features. In *The International Conference on Information Systems Security and Privacy (ICISSP)*.

[8] Arash Habibi Lashkari, Gerard Draper-Gil, Mohammad Saiful Islam Mamun, and Ali A Ghorbani. 2017. Characterization of tor traffic using time based features.. In *ICISSp*. 253–262.

[9] C. Liu, L. He, G. Xiong, Z. Cao, and Z. Li. 2019. FS-Net: A Flow Sequence Network For Encrypted Traffic Classification. In *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*. 1171–1179. https://doi.org/10.1109/INFOCOM.2019.8737507

[10] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. (2019).

[11] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *Computer Science* (2013).

[12] Mohammad, Lotfollahi, Mahdi, Jafari, Siavoshani, Ramin, Shirali, Hossein, Zade, and Mohammdsadegh and. 2020. Deep packet: a novel approach for encrypted traffic classification using deep learning. *Soft Computing* 24, 3 (2020), 1999–2012.

[13] Se Eun Oh, Saikrishna Sunkam, and Nicholas Hopper. 2019. p1-fp: Extraction, classification, and prediction of website fingerprints with deep learning. *Proceedings on Privacy Enhancing Technologies* 2019, 3 (2019), 191–209.

[14] Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A Primer in BERTology: What We Know About How BERT Works. *Transactions of the Association for Computational Linguistics* 8 (2020), 842–866. https://doi.org/10.1162/tacl_a_00349

[15] M. Shen, M. Wei, L. Zhu, and M. Wang. 2017. Classification of Encrypted Traffic With Second-Order Markov Chains and Application Attribute Bigrams. *IEEE Transactions on Information Forensics and Security* 12, 8 (2017), 1830–1843. https://doi.org/10.1109/TIFS.2017.2692682

[16] Hongtao Shi, Hongping Li, Dan Zhang, Chaqiu Cheng, and Xuanxuan Cao. 2018. An Efficient Feature Generation Approach based on Deep Learning and Feature Selection Techniques for traffic classification. *Computer Networks* 132, FEB.26 (2018), 81–98.

[17] Hamid Tahaei, Firdaus Afifi, Adeleh Asemi, Faiz Zaki, and Nor Badrul Anuar. 2020. The rise of traffic classification in IoT networks:: A survey. *Journal of Network and Computer Applications* 154 (2020), 102538.

[18] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762* (2017).

[19] Q. Wang, A. Yahyavi, B. Kemme, and W. He. 2015. I know what you did on your smartphone: Inferring app usage over encrypted data traffic. In *2015 IEEE Conference on Communications and Network Security (CNS)*.

[20] WangXin, ChenShuhui, and SuJinshu. 2020. Real Network Traffic Collection and Deep Learning for Mobile App Identification. *Wireless Communications and Mobile Computing* (2020).

[21] Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. 2014. Recurrent Neural Network Regularization. *Eprint Arxiv* (2014).