STNN: A Novel TLS/SSL Encrypted Traffic Classification System based on Stereo Transform Neural Network

Yu Zhang, Shiman Zhao, Jianzhong Zhang, Xiaowei Ma, Feilong Huang
College of Cyber Science, Nankai University, Tianjin, China
yuzhangjob@gmail.com, 2120170445@mail.nankai.edu.cn, zhangjz@nankai.edu.cn,
mxwei97@gmail.com and bjwindcloudy@mail.nankai.edu.cn

***Abstract*—Nowadays, encrypted traffic classification has become a challenge for network monitoring and cyberspace security. However, the existing methods cannot meet the requirements of encrypted traffic classification because of the encryption protocol in communication. Therefore, we design a novel neural network named Stereo Transform Neural Network (STNN) to classify encrypted network traffic. In STNN, we combine Long Short Term Memory (LSTM) and Convolution Neural Network (CNN) based on statistical features. STNN gains average precision about 95% , average recall about 95% , average F1-measure about 95% and average accuracy about 99.5% in multi-classification. Besides, the experiment shows that STNN obviously accelerates the convergence rate and improves the classification accuracy.**

***Index Terms*—encrypted traffic classification, deep learning neural network, privacy security**

I. Introduction

The process of associating network traffic with specific applications is known as Traffic Classification [1]. However, some traditional traffic classification methods are no longer applicable because of dynamic port and payload encryption [2]. But, the methods based on statistical features do not require prior knowledge of the structure of the traffic packet or the payload of the traffic packet. It depends on the characteristics of the traffic transfer, such as the time interval between packets, packet size, and so on. Importantly, encrypted traffic classification based on statistical features protects privacy security. However, there are still several important problems about classification of encrypted traffic. (1)For different applications, the representative features of each application are not the same, so that effective feature selection consumes a lot of time. (2)For a multi-class classifier, the model may be retrained when a new application comes, so that the training time is very long. (3)For multi-class classification, the traditional approach does not handle all classifications, so that the accuracy of some applications is low.

To handle above challenges, we propose a new learning model named Stereo Transform Neural Network (STNN) based on the statistical features.

II. Related work

Currently, the most popular deep learning algorithm for encrypted traffic classification is CNN. Lotfollahi et al [3] propose a method named Deep Packet. They use 1D-CNN to distinguish between VPN and non-VPN network traffic. Yang et al [4] extract features from the traffic packets. Besides, they do the experiment on several algorithms including Autoencoder and 2D-CNN. Jing et al [5] propose a classification method based on 3D-CNN. They extract the first n packets of each flow. Every packet is transformed into a 2D image using one-hot encoding. The images of the same flow put together a 3D input file. Although the accuracy of these methods is high, training time is long. Besides, the effect of classification cannot meet our requirement because the metrics of multi-class classification are not just accuracy.

Based on the limitation of single model on multi-class classification, researchers propose novel hybrid models. Tong et al [6] propose a novel hybrid model. They combine 1D-CNN with Random Forest. CNN is used to decrease the size of features and Random Forest as a classifier. Manuel et al [7] propose a method based on Recurrent Neural Network (RNN) and 2D-CNN to classify different applications. They consider only the first 20 packets in a flow lifetime. Zhuang et al [8] propose a new deep learning architecture for encrypted network traffic. The architecture makes up of Convolutional Long Short-Term Memory. All of above these hybrid methods involve the payload or some header information of the traffic packet. However, in the era of traffic explosion, handling packets alone can be costly. More importantly, in some cases, privacy policies and laws prohibit accessing or storing packet content which limits the use of payload features [9].

In conclusion, in order to protect privacy, we should not extract the payload of traffic packets for classification. In addition, the training process of the model takes a lot of time. Therefore, we propose a novel learning model combining LSTM and CNN for encrypted traffic classification. Different from previous works, our model is simple in structure and fast in convergence.

978-1-7281-2583-1/19/$31.00 ©2019 IEEE
DOI 10.1109/ICPADS47876.2019.00133

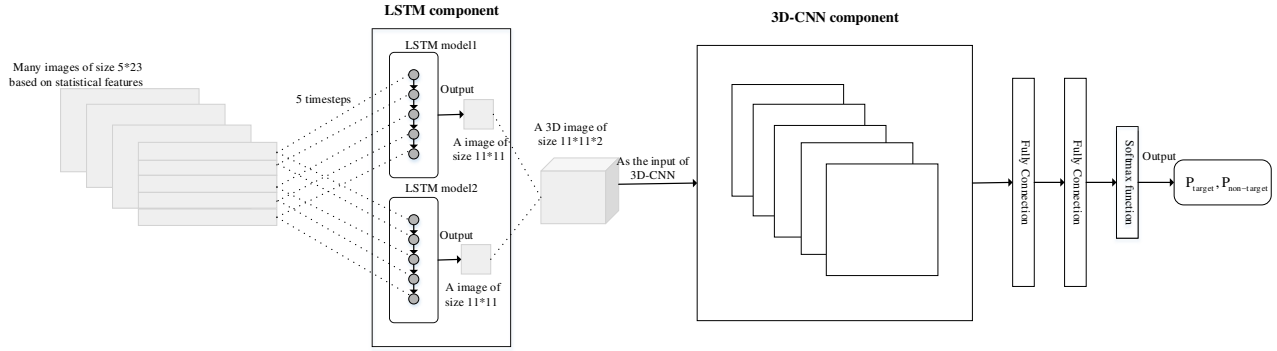

Fig. 1. The structure of the one-class classifier

III. Proposed scheme: STNN

STNN is a multi-classification system. It constructs a one-class classifier for each application. Therefore, STNN can identify different applications by constructing many one-class classifiers. Besides, STNN does not need to be retrained for new applications, providing the possibility for model extension. The framework of STNN combines Long Short Term Memory Network (LSTM) and Convolutional Neural Network (CNN) based on statistical features. Specifically, each one-class classifier consists of LSTM and CNN. First, the statistics of each traffic flow are filled onto an image according to specific rules. Next, each image is fed to two LSTM. The representation vectors extracted by LSTM are combined into a three-dimensional (3D) image. CNN extracts representative features from the 3D image. Finally, softmax function gives the one-class classification results directly based on representative features. A final output of STNN is determined by the results of all one-class classifiers.

A. Offline Training

In the offline training phase, each one-class classifier of the STNN extracts corresponding representative features for online identification of encrypted network traffic. Each one-class classifier consists of LSTM component and CNN component, where LSTM is used to learn the representation vectors and CNN is used to extract the representative features from representation vectors further. The structure of the one-class classifier is shown in Fig 1. Therefore, the following sections mainly introduce (1) the preprocessing of TLS/SSL encrypted traffic, (2) LSTM component for extracting representation vectors, (3) CNN component for extracting representative features and (4) voting mechanism of multi-class classification.

1) The preprocessing of TLS/SSL encrypted traffic: In recent years, TLS/SSL (Transport Layer Security/ Secure Sockets Layer) protocol is used to encrypt network traffic for secure communication. In this paper, we only study encrypted network traffic based on TLS/SSL protocol. Data cleaning is very important before traffic classification. It has been reported that 70% of the smartphone traffic is background traffic and only 30% is directly related to the user interactions [9]. Therefore, data cleaning is inevitable. We first filter the non-TLS/SSL network traffic. And then, we clean data based on [10] [11]. Finally, we can figure out the application that the flow belongs to.

2) LSTM component for extracting representation vectors: LSTM is well-suited to learn knowledge from experience to classify different applications. Obviously, a traffic flow has two directions, both forward and backward. Besides, a traffic flow can be divided into two phases, both the handshake and the data transfer. According to the directions and phases of a flow, we construct the image. Besides, we apply LSTM model to extract the representation vectors from the image.

Table I shows 23 statistical features. Then we construct 5 objects that adopt the 23 statistical features respectively. The 5 objects are as follows:

- the total packets of a flow,
- the forward packets of a flow,
- the backward packets of a flow,
- the handshake packets of a flow,
- the data transfer packets of a flow.

TABLE I
Basic flow-level features

Feature description	Total num
Min, Max, Mean, Median, Std of length, cumulative length, interval time, cumulative interval time	20
Sum packet num, flow packets/s, flow length/s	3

Therefore, we can get 115 (i.e., 115=23*5) statistical features altogether. We believe that there should be some correlation between different objects. Besides, the correlation plays a significant role on traffic classification. In order to classify encrypted traffic based on this correlation, we start to design the structure of LSTM component.

The 5 objects with 23 statistical features are filled onto an image. Each object with 23 statistical features is a line. Therefore, this image has 5 rows and 23 columns. Briefly, an image of size 5*23 is sent to LSTM component. Importantly, the LSTM component encapsulates same two LSTM models. This helps improve convergence and reduce training time. In short, the image is sent to same two LSTM models of LSTM

component. When the image is sent to a LSTM model, each row of the image is the input of the LSTM model. Besides,the timestep of the LSTM model is 5. Based on the structure of image, we can get the representation vectors during the LSTM learning. All the states of each LSTM model are initialized to be zero. Meanwhile, each LSTM model has three layers, where the number of the hidden units of each layer is set as 115. And the number of the output units of each LSTM model is set as 121. Therefore, the output of LSTM component is two 121-dimension representation vectors. Next, the two 121-dimension representation vectors are converted to two 11*11 images. Finally, the representation vectors are combined into a 3D image of size 11*11*2. The 3D image, as the final output of LSTM component, is transmitted to the CNN component.

3) CNN component for extracting representative features: Since each traffic flow has a 3D image, CNN extracts representative features from the 3D image to classify different applications. A CNN model has two important parts: convolution layer and pooling layer. Different designs of CNN network have different effects. Previous study shows [5] that the depth of deep learning is more important than its width. Therefore, STNN applys 3D-CNN to handle 3D image. The parameters of 3D-CNN are listed in Table II.

TABLE II
THE PARAMETERS OF 3D-CNN NETWORK

Layer	Type	Filter	Output
1	conv+leaky_relu	3*3*3*32	11*11*3*32
2	conv+leaky_relu	3*3*3*32	11*11*3*32
3	pooling	2*2*2	6*6*2*32
4	conv+leaky_relu	3*3*3*64	6*6*2*64
5	conv+leaky_relu	3*3*3*64	6*6*2*64
6	pooling	2*2*2	3*3*1*64
7	conv+leaky_relu	3*3*3*128	3*3*1*128
8	BN+pooling	-	2*2*1*128
9	flatten+dense+dropout	-	512
10	dense+dropout	-	1024
11	softmax	-	2

Because small kernel can reduce network complexity, we set the size of convolution kernel as 3*3. Besides, the padding pattern is 'same'. During the training process, negative values can occur, so activation function 'leaky_relu' is better than activation function 'relu'.

4) voting mechanism of multi-classification: Based on the structure of the one-class classifier shown by Fig 1, we construct a one-class classifier for each application. Finally, the final decision is determined by the results of all one-class classifiers. In Fig 1, one-class classifier combines LSTM network and CNN network together to classify whether a testing flow belongs to itself. The last layer of one-class classifier is softmax layer. For a testing flow, the output of one-class classifier is two probability value (P_{target},$P_{\mathrm{non-target}}$). P_{target} represents the probability value of belonging to this classification. And $P_{\mathrm{non-target}}$ represents the probability value of not belonging to this classification. The sum of P_{target} and $P_{\mathrm{non-target}}$ is 1. Besides, the greater difference between these two probability values, the better classification effect of the one-class classifier on this testing flow. The smaller difference between the two probability values, the easier for this one-class classifier to misjudge this testing flow. Therefore, we work out a effective mechanism (i.e., voting list). We divide the quotient between these two probability values into k categories, such as $(0, 10^1]$,$(10^1, 10^2]$,$(10^2, 10^3]$,...,$(10^{k-2}, 10^{k-1}]$,$(10^{k-1}, +\infty]$. These k intervals are represented by $Gap_i\,(i = 1, 2, \ldots, 7)$. Specifically, Gap_1=$(0, 10^1]$, Gap_2=$(10^1, 10^2]$,...,Gap_k=$(10^{k-1}, +\infty]$. We design a voting list (VL) for each one-class classifier. The length of voting list is k (corresponding to Gap_k). We assume that N one-class classifiers construct a multi-classification system. O_i represents the outputs of i^{th} one-class classifier. CO_i represents the correct outputs of i^{th} one-class classifier. The voting list of the i^{th} one-class classifier is called VL_i. The voting list consists of k values. And each value is called v_j (j=1,2,,k). The j^{th} value in i^{th} voting list (VL_i) is called v_{ij}. Besides, the values of the v_{ij} is calculated by Equation (1)

$$\begin{aligned} v_{ij} &= \frac{|T|}{|C_i|} \\ T &= \{t | t \in Gap_j, t \in CO_i\} \\ C_i &= \{c | c \in Gap_j, c \in O_i\} \end{aligned} \tag{1}$$

Obviously, the voting lists are built during the training process and applied directly to the online identification. When a testing flow comes, STNN put the testing flow into N one-class classifiers. If many one-class classifiers determine this testing flow as true, these classifiers should further look up their own voting lists. And then, these one-class classifiers find the corresponding value in their voting lists based on their outputs. The testing flow is decided by the one-class classifier that has the maximum corresponding value.

B. Online Identification

During the offline training, each one-class classifier of STNN learns their own representative features. When offline training is finished, STNN can be used to identify traffic flows online. When a testing flow comes, there are three situations as follows.

- If the testing flow is judged to be true by only a one-class classifier, the final result should be determined as the label of the one-class classifier.
- If the testing flow is judged to be false by all one-class classifiers, the final result should be determined as unknown traffic.
- If the testing flow is judged to be true by more than one one-class classifier, the final result should be determined by voting mechanism.

IV. EXPERIMENTAL EVALUATION

In this section, we evaluate the effectiveness of STNN. All methods have been tested in python.

A. Dataset

We capture traffic from the campus network. Besides, public dataset provided by Lashkari [12] is used to extend our dataset. Our dataset has 17 applications, including Amazon, Baidu, Bilibili, Cloudfront, Dazhongdianping, Helpshift, Hockeyapp, Kugou, Meituan, Alicdn, Mmstat, Ziroom, Outlook, QQ, Sina, Weibo and Yangkeduo. The total dataset we captured is 500G. After the dataset is preprocessed according to section III-A1, it is divided into two parts, the training set and the testing set. Besides, the size of training set is 150,000 training flows and the size of testing set is 30,000 testing flows.

B. Metrics

We introduce the four widely-used metrics macroP, macroR, macroF1 and average accuracy. Besides, the four metrics are calculated by Equation (2) (3) (4) (5) respectively to evaluate the performance of multi-class classification.

$$macroP = \frac{1}{n}\sum_{1}^{n} precision_i \tag{2}$$

$$macroR = \frac{1}{n}\sum_{1}^{n} recall_i \tag{3}$$

$$macroF1 = \frac{2 * macroP * macroR}{macroP + macroR} \tag{4}$$

$$average\_accuracy = \frac{1}{n}\sum_{1}^{n} accuracy_i \tag{5}$$

C. The effectiveness of voting list

The voting list is very significant for STNN, as it can influence the effect of STNN. Of course, most of multi-classification systems usually decide this testing flow as the application that has a maximum P_{target} [13]. Thus, we make a comparison between the two methods. In Table III, we can see that voting list is better than maximum P_{target} alone.

TABLE III
THE COMPARISON RESULT

	macroP	**macroR**	**macroF1**	**average accuracy**
maximumP	0.9178	0.9473	0.9323	0.993
Voting list	0.9454	0.9485	0.9469	0.9953

D. Performance of the training process

In order to prove that the construction of STNN on CNN is significant, we compare the loss and accuracy between STNN and CNN. Results show that the increasing speed of accuracy of STNN comes faster than CNN. Besides, the difference of increasing speed also has the similar trend as the loss figure. Therefore, STNN can obviously improve the classification accuracy and accelerate the convergence rate.

E. Performance of STNN

Due to the limited space, we briefly describe the results. Results represent that STNN gains average precision about 95% , average recall about 95% , average F1-measure about 95% and average accuracy about 99.5% .

V. Conclusion

In this paper, we propose a novel neural network named STNN to classify encrypted network traffic using TLS/SSL protocol. STNN is a multi-classification system, which constructs a one-class classifier for each application. We validate the effectiveness of STNN with network traffic caught from the real world. The experimental results show that STNN has an outstanding performance for all the target applications. Besides, STNN is simple in structure and fast in convergence. In general, STNN is an efficient multi-classification system for encrypted network traffic.

Acknowledgment

This work is supported by the National Key Research and Development Project (No. 2018YFB0804702).

References

[1] A. Dainotti, A. Pescape, and K. C. Claffy, *Issues and Future Directions in Traffic Classification*, 2012.
[2] P. Velan, "A survey of methods for encrypted traffic classification and analysis," *International Journal of Network Management*, vol. 25, no. 5, pp. 355–374, 2015.
[3] M. Lotfollahi, R. S. H. Zade, M. J. Siavoshani, and M. Saberian, "Deep packet: A novel approach for encrypted traffic classification using deep learning," 2017.
[4] Y. Yang, C. Kang, G. Gou, Z. Li, and G. Xiong, "Tls/ssl encrypted traffic classification with autoencoder and convolutional neural network," in *2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, 2018.
[5] J. Ran, Y. Chen, and S. Li, "Three-dimensional convolutional neural network based traffic classification for wireless communications," in *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. IEEE, 2018, pp. 624–627.
[6] V. Tong, H. A. Tran, S. Souihi, and A. Mellouk, "A novel quic traffic classifier based on convolutional neural networks," in *2018 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2018, pp. 1–6.
[7] 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. PP, no. 99, pp. 1–1, 2017.
[8] Z. Zou, J. Ge, H. Zheng, Y. Wu, and Z. Yao, "Encrypted traffic classification with a convolutional long short-term memory neural network," in *2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, 2018.
[9] S. Rezaei and X. Liu, "Deep learning for encrypted traffic classification: An overview," 2018.
[10] C. Liu, Z. Cao, Z. Li, and G. Xiong, "Lafft: Length-aware fft based fingerprinting for encrypted network traffic classification," in *2018 IEEE Symposium on Computers and Communications (ISCC)*, 2018.
[11] C. Liu, Z. Cao, G. Xiong, G. Gou, S.-M. Yiu, and L. He, "Mampf: Encrypted traffic classification based on multi-attribute markov probability fingerprints," in *2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS)*. IEEE, 2018, pp. 1–10.
[12] "http://www.ahlashkari.com/data-sets.asp."
[13] N. Fu, Y. Xu, J. Zhang, R. Wang, and J. Xu, "Flowcop: Detecting" stranger" in network traffic classification," in *2018 27th International Conference on Computer Communication and Networks (ICCCN)*. IEEE, 2018, pp. 1–9.