Identification of Encrypted Traffic Through Attention Mechanism Based Long Short Term Memory

Haipeng Yao, *Member, IEEE,* Chong Liu, Peiying Zhang, Sheng Wu, *Member, IEEE,* Chunxiao Jiang, *Senior Member, IEEE,* and Shui Yu, *Senior Member, IEEE*

Abstract—Network traffic classification has become an important part of network management, which is beneficial for achieving intelligent network operation and maintenance, enhancing the network quality of service (QoS), and for network security. Given the rapid development of various applications and protocols, more and more encrypted traffic has emerged in networks. Traditional traffic classification methods exhibited the unsatisfied performance since the encrypted traffic is no longer in plain text. In this work, we modeled the time-series network traffic by the recurrent neural network (RNN). Moreover, the attention mechanism was introduced for assisting network traffic classification in the form of the following two models, the attention aided long short term memory (LSTM) as well as the hierarchical attention network (HAN). Finally, relying on the ISCX VPN-NonVPN dataset, extensive experiments were conducted, showing that the proposed methods achieved 91.2% in accuracy while the highest accuracy of other methods was 89.8% relying on the same dataset.

Index Terms—Network traffic classification, encrypted traffic, LSTM, attention mechanism.

1 Introduction

Network traffic classification plays a critical role in next-generation communication networks, which aims at classifying network traffic based on both the type of protocols, such as HTTP, FTP, and the type of applications, such as Facebook, Skype, etc. Meanwhile, network traffic classification is beneficial in terms of understanding the distribution of traffic flows, improving the utilization efficiency of network resources, enhancing quality of service (QoS) and guaranteeing network security. In addition, as network traffic becomes larger and larger, operators tend to adopt big data tools for stable storage and fast processing, such as Hadoop and Spark. Besides, distributed computing methods are also used in this task because of the high requirements for real-time calculation.

Recently, with the extensive demands for protecting both data transmission and user privacy, protocols and applications prefer to adopting encryption methods. Under such a circumstance, the amount of the encrypted traffic has grown extensively in current communication networks. A variety of encryption mechanisms have been employed [1], such as SSH, VPN, SSL, encrypted P2P, VoIP etc. These encryption algorithms are different from each other, since some encrypted data packets locate in the transport layer, while others locate in the application layer, which make the classification of encrypted traffic difficult and hence impose a huge challenge on the network traffic classification [2]. Moreover, even if the same encryption algorithm is utilized, the encrypted traffic can exhibit different data distributions because of the different distributions of the original traffic.

Traditional port-based traffic identification methods have unsatisfied performance on the classification of encrypted traffic, since they utilized the official standards defined by the Internet assigned numbers authority (IANA) to identity the type of applications. However, some protocols did not follow those standards, such as P2P protocol utilized the random port, and HTTP protocol used the 80 port for disguising. Moreover, some deep packet inspection (DPI) based methods conducted the traffic classification by regular expression for matching the payload data, while the payload of the encrypted packet was changed relying on the encryption algorithm. Consequently, the DPI based methods could only identify those coarse-grained protocols such as SSL, but completely failed to identify the encrypted traffic. In addition, most of the machine learning aided traffic classification methods [3], [4], [5] were based on manual extraction of data packets or their statistical features at the data flow level to train the classifier, such as the duration of a flow, the total number of packets, the length of a packet, the number of the bytes contained in a flow, as well as the interval of the packet arrival. Since these characteristics require prior knowledge and experience, the extraction of them would also be time-consuming. More importantly, it cannot be guaranteed that these features are really helpful

- *H. Yao and C. Liu are with State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China.*
  Email: yaohaipeng@bupt.edu.cn.
- *P. Zhang is with the College of Computer & Communication Engineering, China University of Petroleum (East China), Qingdao 266580, China. Email: 25640521@qq.com.*
- *S. Wu is with the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China. Email: thuraya@bupt.edu.cn.*
- *C. Jiang is with Tsinghua Space Center, Tsinghua University, Beijing 100084, China. Email: jchx@tsinghua.edu.cn.*
- *S. Yu is with School of Software, University of Technology Sydney, Australia.*

in improving the performance of classification. Therefore, to the best of our knowledge, all of these aforementioned methods cannot achieve preferable results on the encrypted traffic classification problem.

Recently, deep learning has rapidly developed and has witnessed its great success in a variety of areas, such as computer vision (CV), speech recognition, natural language processing (NLP), etc. Meanwhile, deep learning methods have been widely used in the scenario of communication networks. Network traffic classification can be regarded as a common classification problem in the field of machine learning, and some papers [6], [7], [8], [9], [10] have proposed the application of deep learning methods in solving such a problem. Nevertheless, most of them utilized CNN to extract the features from the traffic flow without considering the timing features among different packets. In this paper, we propose to utilize LSTM for network traffic classification tasks, which can not only omit the complex feature engineering, but also automatically learn the temporal relationship between traffic flows. The specific contributions of this paper can be summarized as follows.

- We treat the network traffic flow as time-series and analyze it as text data with the aid of LSTM model. Experimental results yield the best representation of the network traffic, where each flow contains 10 packets and each packet contains 1500 bytes.
- Two models are proposed for encrypted traffic classification, i.e., attention based LSTM and hierarchical attention network (HAN). The attention based LSTM focuses more on important data packets in the traffic flow, while HAN is capable of distinguishing the role of different bytes in each packet during the process of classification.
- Simulation results demonstrate that the classification accuracy rate of our proposed model can achieve 91.2%, which outperforms traditional machine learning based methods.

The organization of our paper is as follows. Section 2 discusses the related works. The methodology of traffic classification system is investigated in Section 3. Moreover, we propose the attention aided LSTM and HAN architecture in Section 4. In Section 5, we show the experimental results, followed by our conclusions and future work in Section 6.

2 Related Work

Network traffic classification has become a research hotspot in both academic and industrial fields in recent years [2], [11], [12], [13]. With the rapid growth of encrypted traffic in the network, traditional traffic classification methods based on both ports and deep packet inspection are unable to meet current requirement. Machine learning assisted traffic classification has been widely investigated for the identification and classification of encrypted traffic. Because network traffic datasets are often hundreds of GB or TB, the usage of big data has also attracted more and more attention. Algorithms combine with big data and distributed computing to solve the problems of large amount of training data and time-consuming calculation. Some people [14] use hadoop as a tool for data storage and preprocessing to help reduce dimensionality and extract features. The processed data is then used as input to the model. In addition, the Hadoop-based distributed machine learning model has also been widely used. Quoc *et al.* [15] and Yuan *et al.* [16] respectively proposed distributed SVM and C4.5 algorithms with Hadoop to reduce the training time. Recently, Aceto *et al.* [17] proposed a method which combined the benefits of deep learning and big data to solve network traffic classification.

2.1 Machine Learning Based Encrypted Traffic Classification

The machine learning based encryption traffic classification method mainly includes two parts: feature extraction and model construction. We will introduce the relevant work of the two parts separately.

In the state-of-the-art, Cao *et al.* [2] introduced the recent researches on encrypted traffic identification and summarized the cons and pros of common methods. In [3], Alshammari *et al.* investigated two common methods for encrypted traffic identification, i.e., expert driven system and data driven system, and concluded that the data driven system outperforms the expert driven system in terms of both detection accuracy rate and false positive rate. Moreover, Bacquet *et al.* [18] compared the classification performance of both C4.5 and MOGA relying on different SSH protocol dataset, and showed that C4.5 had better robustness while MOGA yielded higher classification accuracy. Furthermore, Dusi *et al.* [5] used Gaussian mixture model (GMM) and support vector machine (SVM) based method for SSH encrypted traffic classification. Zhang *et al.* [19] proposed an improved clustering algorithm to identify the encrypted network traffic, and it had better performance in comparison to the k-Means method. Moreover, in [20], Lashkari *et al.* proposed the ISCX Tor-NonTor dataset, which used the C4.5 and k-nearest neighbors (KNN) as the classifier and achieved an accuracy rate of 70%. In [21], Sun *et al.* conducted encrypted traffic classification relying on the combination of the statistic-based and signature-based method, which achieved 94.52% in F1 score. In [22], Shahbar *et al.* used the method of Bayes net, Naive Bayes, C4.5, Random Forest to conduct a comparative test on the Anon17 dataset which focused on Tor and I2P traffic. In this dataset, each sample contains 97 dimensional features and a label. In [23], Taylor *et al.* proposed the AppScanner that used the support vector classifier and random forest classifier to automatic identify mobile Apps. In [24], Aceto *et al.* proposed a framework where any classifier may be readily plugged-in/out to improve performance further.

As for feature selection, some people extract statistical features from the traffic flow. Alshammari *et al.* [4] extracted 22 stream features instead of using IP addresses and port numbers for distinguishing the SSH and Skype protocols, and found that C4.5 and RIPPER algorithm had better classification performance. Scherrer *et al.* [25] used non-Gaussian method and long memory statistical features to classify traffic. Meanwhile, others treated the network traffic as time-series data and designed classifiers relying on the timing features of the network traffic. In [26], Santos *et al.* proposed a time-series analysis method for classifying

HTTP protocol. Lashkari *et al.* [27] proposed the ISCX VPN-NonVPN dataset by using C4.5 decision tree to train the classifier, which had an accuracy rate of 88% in the case of only using the time features, such as duration, forward inter-arrival, backward inter-arrival, flow inter-arrival, etc. In [28], a KNN based algorithm was proposed relying on the dataset in [27], which extracted 111 features, and finally obtained an accuracy rate of 93%. In a nutshell, these feature selection methods need to manually extract features for each category. This process requires a comprehensive prior knowledge of the field so that we may lose the most important features during this process. This method is difficult to migrate quickly when encountering a new scene.

2.2 Deep-Learning Based Encrypted Traffic Classification

With the development of deep learning, the related algorithms have been applied to a range of fields. Recently, Rezaei *et al.* briefly summarized the application of deep learning in encrypted traffic classification in [29]. Deep learning based traffic classification can be traced back to 2015 [6], where Wang *et al.* used only a simple SAE automatic encoder. Later, Lotfollahi *et al.* [7] proposed the SAE and two layers convolutional neural network (CNN) model, which achieved the performance of 95% in F1 score in the context of application classification, as well as of 93% in F1 score in the context of protocol classification. The CNN based model was also used in [30], where Wang *et al.* used representation learning method for network anomaly detection and intrusion detection system. In [8], an one-dimensional CNN was proposed to represent traffic flows as pictures, which yielded the F1 score of 86.6% for the application classification. Furthermore, Lopez *et al.* [9] integrated the CNN and recurrent neural network (RNN) model for traffic classification. It represented one flow as 20 packets, and each packet extracted 6-dimensional features and finally obtained the F1 score of 95%. Then Aceto *et al.* used 1D-CNN, 2D-CNN, and LSTM proposed in [8], [9] for mobile encrypted traffic classification in [31]. In addition, Wang *et al.* [10] designed a hierarchical spatial-temporal feature-based model for the intrusion detection system. In [32], Rui *et al.* proposed the byte segment neural network(BSNN) which use attention encoder for protocol classification. It gains 95.8% F1 score on real-world data.

In this paper, we employ LSTM to model the timing features by viewing the network traffic as the time-series data, and introduce the attention mechanism to improve the long-term memory ability of LSTM. The closest work to our model is the BSNN proposed by Rui *et al.* in [32]. From the perspective of model structure, the BSNN is similar to the HAN, since they are both composed of two layers of attention encoder. But the input in our model is the entire traffic flow which is padded to a same dimension matrix and is normalized in the range of [0, 1]. While the input of BSNN is a packet which is divided into different segments and each byte is transformed to a one-hot vector in 256-dimension.

However, it is difficult to determine which method is more effective, since the dataset they have utilized are different from each other. In addition, their classification performances are based on the specific data and specific protocols, which may be difficult to extend to a more general encrypted traffic classification scenario. However we can find that [7], [8], [27] all use the ISCX VPN-NonVPN dataset, and the results among them are easy to compare.

3 Methodology

3.1 Dataset

As aforementioned, the dataset and evaluation criteria are not consistent in the network traffic classification. In such a case, the models and algorithms proposed in existing works cannot be compared with each other in terms of a common benchmark. In this paper, we focus on the ISCX VPN-NonVPN dataset [27], which contains two levels of traffic classification tasks. The first level is identification of protocol types (chat, email, etc.), while the second level is the identification of application types (facebook, skype, etc). We classify them from the perspective of the protocol type in this paper which contains 6 kinds of non-VPN data and 6 kinds of VPN data. The dataset is saved in the form of pcap file, where the name of each file is specified by protocol. In addition, those two kinds of data, i.e., VPN and NonVPN, allow us to train the classifier for the encrypted packets. The original data set is about 35G, so we execute the data preprocessing process on Hadoop platform, and the data after processing is about 1G. In addition, since we need to perform multiple sets of comparison experiments to select the best hyper-parameters, so our algorithms are distributed based on tensorflow to reduce training time.

3.2 Data Preprocessing

In order to facilitate the training model, we need to store the original packets in the form of pcap format according to three structures, i.e., category, flow and packet. Data preprocessing consists of the following four stages: traffic flow segmentation, unwanted field removal and data normalization, time-series data representation, and the segmentation of training set and test set. The process of data preprocessing is shown in Fig. 1.

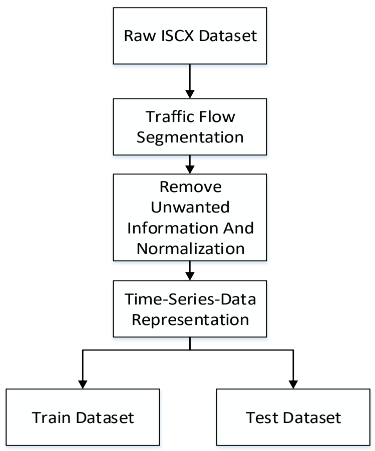

Raw ISCX Dataset
Traffic Flow Segmentation
Remove Unwanted Information And Normalization
Time-Series-Data Representation
Train Dataset
Test Dataset

Fig. 1. The flow diagram of data preprocessing.

3.2.1 *Traffic Flow Segmentation*

Let us take each network traffic flow $F_i,\ i = 1, 2, 3, ..., N$ for example, which consists of multiple data packets $P_j,\ j = 1, 2, 3, ..., M$. We utilize a five-tuple including the source IP, destination IP, source port, destination port, transport layer protocol such as TCP and UDP, to identify a traffic flow. Packets having the same five-tuple belong to the same traffic flow. In particular, we put the source-to-target and target-to-source packets together to form bidirectional flows. In such a case, we can use the SplitCap tool to split the original pcap files into many bi-flows. Then, we saved each bi-flow as a small file in the form of pcap file, and label the data with the corresponding file name. The number of flows that each class contains is shown in Fig. 2. In such a case, we can use the SplitCap tool to split the original pcap files with the aid of the above-mentioned five-tuple. Then, we save the data of each flow as a small file in the form of pcap file, and label the data with the corresponding file name. The number of flows that each class contains is shown in Fig. 2.

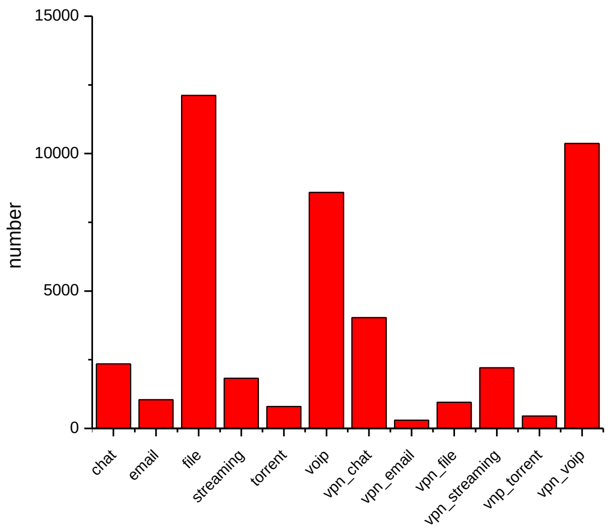


Fig. 2. The distribution of different classes of data.

As we can see from Fig. 2, there is a distinct difference between the data with different category, which is extremely unbalanced. Therefore, we use cost-sensitive learning method to reduce the impact of unbalanced dataset. Furthermore, some files may generate a large amount of flows after segmentation, and thus we are able to choose a portion of the flows and abandon the rest. However, because of the difference of preprocessing methods used, the final data sets may be very different.

3.2.2 *Unwanted Field Removal and Data Normalization*

Each packet consists of multiple protocol layers, so we can use all data or just the application layer (L7) data for classification. We will verify which is better in Section 5. In attention, because the dataset was collected by several fixed IP addresses where each IP address is responsible for collecting data of a certain protocol type, it can mingle external information so that the experimental results are greatly biased when using all data. Therefore, we chose to delete the data link layer information and IP address [7]. As for data normalization, considering the packets consisting of binary strings, each byte consists of 8 bits and can be represented as a decimal number in the range of $[0 - 255]$. The input of neural network needs to be normalized, so we normalize each byte in the range of $[0 - 1]$.

3.2.3 *Time-Series-Data Representation*

In order to use the traffic flow as an input to the model, we need to represent it as an N*M-dimensional matrix, where N means the number of packets in a traffic flow and M means the number of bytes in a packet. If a packet contains less than M bytes, we need to pad 0 after its data until its length reaches M. While if the packet length is larger than M, it needs to be truncated to retain only the first M bytes. The same method will also be used to N. This method can solve the problem of variable length sequences. Moreover, we use 0 as the padding value, which does not bring any additional information and bias to the classification result, because if the input of neural network is 0 then the output is also 0. To determine the best values for N and M, we first visualize the data distribution as shown in Fig. 3.

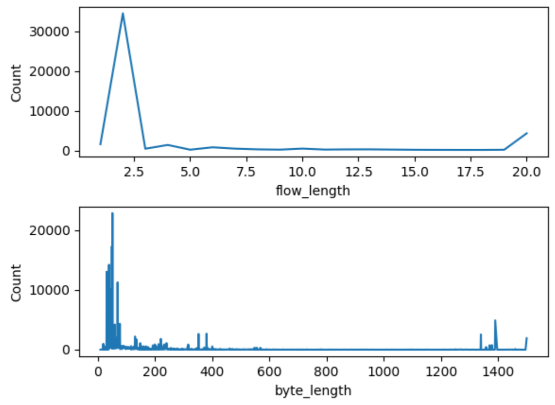


Fig. 3. The distribution of the length of traffic flow and the length of packet.

From Fig. 3, we can conclude that most of the data consists of only two data packets, because the dataset contains many domain name system (DNS) protocol packets. We will construct a comparison experiment in section 5 to verify the impact of the DNS flows. In order to determine the optimal length of traffic flow N, we take experiments with the length of $N = 5$, $N = 10$ and $N = 20$ to determine the hyper-parameter, respectively.

Moreover, in Fig. 3, the number of bytes contained in the packet P_j is mainly distributed at both ends. In order to determine the optimal length of package M, contrast experiments with $M = 500$, $M = 1000$ and $M = 1500$ are conducted. Once the best choices for N and M are determined, a traffic flow can be represented as a matrix of $N \times M$, and the value of each element in the matrix is between 0 and 1, i.e., each traffic flow is represented as a matrix of $N \times M$ as an input sample of LSTM. Fig. 4 portrays the representation of each traffic flow.

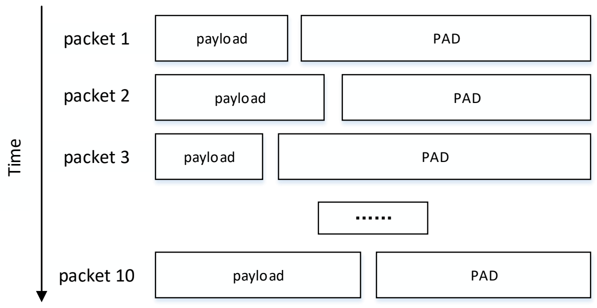


Fig. 4. The matrix representation of traffic flow.

3.2.4 Dataset Segmentation

In the stage of dataset segmentation, in order to ensure the generalization ability of the model and the credibility of the experimental results, we adopt the 10-fold cross-validation method. The dataset is randomly divided into 10 parts. Next, the training-validation-test sets are randomly selected in the manner of 8-1-1 for 10 times, and the final result is averaged. Specifically, the validation set is used to determine the hyper parameters in the experiment, while the test set is conceived for representing the final model effect. Finally, the statistical information of dataset is shown in Table 1.

TABLE 1
The statistical information of dataset.

	VPN	Non-VPN
Training set	15545	22706
Validation set	1943	2838
Test set	1943	2838

4 Attention based LSTM and HAN Architecture

Considering the network traffic as time-series data, we used the improved RNN for modeling it in this section. Because the sequence of traffic data is very long, for example, there are 1500 bytes per packet in our dataset, so we use LSTM [33] instead of the original RNN. The reason is that LSTM can remove or add information to the hidden state vector with the aid of the gate function. This means that LSTM can retain important information in hidden layer vectors.

There are three gate functions, i.e., the forget gate, the input gate and the output gate. The **forget gate** is used to control how much information in C_{t-1} is retained in the process of calculating C_t. The forget vector f_t can be given by

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f), \tag{1}$$

where W_f, and b_f are the parameters of forget gate, x_t is the input vector in step t, and h_{t-1} is the hidden state vector in step $t-1$.

Moreover, the **input gate** decides how much information of x_t is added to C_t, which can be express as:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i), \tag{2}$$

where W_i, b_i are the parameters of input gate, and hence C_t can be calculated relying on forget gate vector f_t as well as on the input gate vector i_t, i.e.

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \widetilde{C_t}, \tag{3}$$

where $\widetilde{C_t} = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$ denotes the information represented in the hidden layer vector.

The **output gate** controls the output in C_t, and we have

$$\begin{aligned} o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o), \\ h_t &= o_t \cdot \tanh(C_t), \end{aligned} \tag{4}$$

where W_C, W_o, b_C, and b_o are the parameters of output gate, and C_t is the internal state in step t. However, the length of the packet is large so that the LSTM model cannot memorize all of the information. Besides, the long sequence may also produce gradient explosions and gradient vanishing during the training process.

To address the long-term dependence of time-series data, in [34], Bahdanau *et al.* utilized the attention mechanism to the seq2seq model, which was used to calculate the weight of all hidden vectors, as shown in the following.

$$u_i = \tanh(W_p h_i + b_p), \tag{5}$$

$$\alpha_i = \frac{\exp(u_i^T u_s)}{\Sigma_j \exp(u_j^T u_s)}, \tag{6}$$

$$c = \sum_i \alpha_i h_i, \tag{7}$$

where W_p, b_p and u_s are all parameters that need to be trained, while u_i is the importance score of each packet, and α_i is the normalized weight. We have $\Sigma_i \alpha_i = 1$. In fact, the above calculations are equivalent to using a fully connected neural network to calculate the weight of each vector, and then weighting each hidden layer vector with the weighted value to obtain the intermediate vector c. Based on this attention mechanism, in the following, we propose the attention based LSTM and HAN [35] for network traffic classification.

4.1 Attention Based LSTM

The attention-based LSTM neural network structure diagram is shown in Fig. 5, where each packet P_i is encoded into an input vector by a Bi-LSTM model. The hidden layer vectors $\overrightarrow{h_i}$ and $\overleftarrow{h_i}$ are connected to form the context vector h_i = $[\overrightarrow{h_i}, \overleftarrow{h_i}]$, where h_i represents the encoding vector of each packet and it contains the information of the preceding and following packets. Then the attention mechanism is used to calculate the weight of h_i. As shown in Eq. (7), we multiply each vector by their weight and add them up. Finally, we obtain the encoder vector c of the traffic flow.

The pseudo-code of Algorithm 1 is shown in Alg. 1.

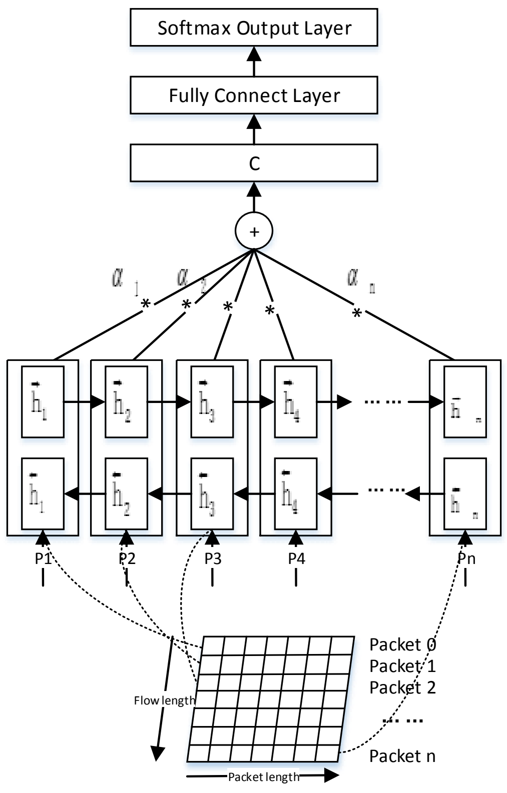


Fig. 5. The architecture diagram of the attention based LSTM model.

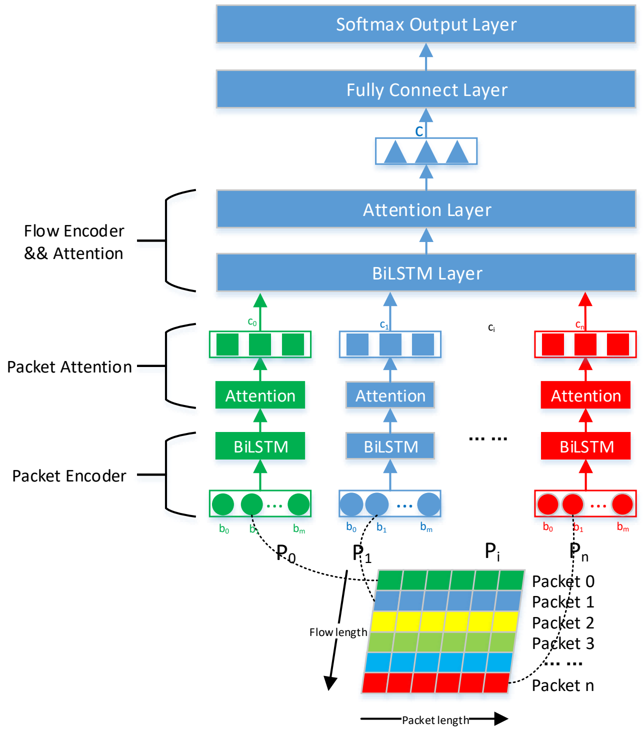


Fig. 6. The architecture diagram of HAN model.

Algorithm 1 Attention based LSTM

```
1: Input: Network flow data $F = \{P_i|i = 0, 1, 2, ..., N\}$
2: Output: The predict label $predict$ of $F$
3: for $P_i$ in $F$:
4:     $\overrightarrow{h_i} = LSTM_1(P_i)$
5:     $\overleftarrow{h_i} = LSTM_2(P_i)$
6:     $h_i = [\overrightarrow{h_i}, \overleftarrow{h_i}]$
7: end for
8: $c = \sum_i \alpha_i h_i$                    (Eq. (5)- (7))
9: $predict = softmax(fullyConnect(c))$
```

4.2 HAN Architecture

The HAN architecture is a kind of neural network structure proposed by [35] in the scenario of text classification. It is similar to attention-based LSTM except that two layers of LSTM networks are used to encode each packet and flow, separately. The architecture diagram is shown in Fig. 6.

The first layer of the LSTM network uses each byte b_j in the packet as input, and processes only one byte and encodes it at each time. Therefore, the rolled step of the LSTM is as large as the length of packet. Each b_j is encoded as a hidden vector $h_j = [\overrightarrow{h_j}, \overleftarrow{h_j}]$. Then, it uses the attention mechanism to calculate the weight of each byte and performs a weighted summation for getting the vector p_i , which represents the information in each packet. The second layer of the LSTM network is used to exactor the encoder vector of the whole flow F, where p_i is encoded as the input of the second LSTM layer at each time, and the attention mechanism is also used to calculate the importance score of each data packet. Moreover, weighted summation is performed to get the representation vector F of the traffic flow. The pseudo-code of Algorithm 2 is shown in Alg. 2.

Algorithm 2 HAN Architecture

```
    Input: Network flow data $F = \{P_i|i = 0, 1, 2, ..., N\}$
2:  Output: The predict label $predict$ of $F$
    for $P_i$ in $F$:
4:      for $b_j$ in $P_i$:
            $\overrightarrow{h_(b_j)} = LSTM_1(b_j)$
6:          $\overleftarrow{h_(b_j)} = LSTM_2(b_j)$
            $h_(b_j) = [\overrightarrow{h_(b_j)}, \overleftarrow{h_(b_j)}]$
8:      end for
        $c_i = \sum_j \alpha_j h(b_j)$                (Eq. (5)- (7))
10:     $\overrightarrow{h_i} = LSTM_1(c_i)$
        $\overleftarrow{h_i} = LSTM_2(c_i)$
12:     $h_i = [\overrightarrow{h_i}, \overleftarrow{h_i}]$
    end for
14: $c = \sum_i \alpha_i h_i$                    (Eq. (5)- (7))
    $predict = softmax(fullyConnect(c))$
```

4.3 Output Layer And Objective Function

We can choose one of the above-mentioned two methods to encode the traffic flow into the vector c, which is the input of the full-connection layer. The processed result is then sent to the softmax layer for obtaining the probability $\hat{y}_i$. We utilize a dropout in the full-connection layer for the sake of increasing the generalization ability of the model, and keep the probability of the dropout as 0.8.

Cost-sensitive learning is used because of the serious class imbalance problem in the training data. Cost-sensitive learning means that the cost of mis-classification in different class is varying, so we introduce a cost vector $c_n \in [0, \infty)^K$ for each sample where each dimension means the cost of classifying the sample as k-th class. The loss function proposed in [36] is used to calculate the loss of each sample as follows.

$$\delta_{n,k} = ln(1 + exp(z_{n,k} \cdot (r_k(x_n) - c_n[k]))). \tag{8}$$

Where $\delta_{n,k}$ represents the loss of classifying the n-th sample as k-th class, $r_k(x_n)$ represents the k-th dimension of the neural network output vector, and $c_n[k]$ represents the k-th dimension of the cost vector, $z_{n,k}$ indicates whether the n-th sample is k-th class, and its calculation formula is $z_{n,k} = 2[c_n[k] = c_n[y_n]] - 1$. The advantage of this loss function is smooth and differentiable so that it can be used as the objective function of the neural network to perform the back propagation directly. Therefore, the objective function is as follows.

$$L(\theta) = \sum_{n=1}^{N} \sum_{k=1}^{K} \delta_{n,k}. \tag{9}$$

In order to get prediction label of the model, the following formula can be used.

$$\hat{y}_n = \mathop{\arg\min}_{1 \le k \le K} r_k(x). \tag{10}$$

5 Experiments and Result Analysis

5.1 Experimental Environment

The experimental environment is listed as follows: Ubuntu 14.04 OS, TensorFlow 1.4.0, python2.7, NVIDIA 1080Ti graphics and 16G memory. In order to prevent the over-fitting phenomenon, the dropout technique is utilized and its probability is set as 0.8 during the training process. As for the optimization method, Adam optimization is employed and the initial learning rate is set as 0.001. Meanwhile, we have used Relu as the activation function. Moreover, the batch size is 64 and the program is trained for 30 epochs. The major parameters in both models include the $lstm\_size$ of LSTM cell, the $hidden\_size$ of the fully connected layer, and the out_size of the output layer which is selected according to the number of output classes. The $lstm\_size$ and $hidden\_size$ are both the dimension of vector which encoded the information of the flow. Because the shape of a flow is similar to the shape of an article, so we can refer to the hyper-parameter setting in the text classification task [35]. A vector of approximately 100 dimensions is sufficient to solve this problem. Specially, in the Attention based LSTM model, we set $lstm\_size$ as 100 and $hidden\_size$ as 128; while in HAN, the first layer is used to encode the packet which is a long sequence, and the second layer is used to encode the entire flow, so we set $lstm\_size$ in both LSTM layers as 128 and $hidden\_size$ as 128. As for the input shape of the model, it is selected according to the following experiment.

5.2 Evaluation Metrics

In order to evaluate the experimental result, we use four evaluation criteria, i.e., accuracy (acc), precision (P), recall (R), and F1 score ($F1$). Because encrypted traffic classification is a multi-category task, we need separately calculate the above indicators for each category. Specially, we use N as the total number of training samples; TP_c to indicate the quantity that originally belongs to category c, and is predicted by the model as c; FP_c indicates the quantity that originally does not belong to category c, but is predicted by the model as c; TN_c indicates the quantity that does not belong to category c, and is not predicted to be class c; FN_c indicates the quantity that it belongs to class c, but is misclassified to other class. Hence, the definition of aforementioned four evaluation metrics can be given by:

$$P_c = \frac{TP_c}{TP_c + FP_c},\ R_c = \frac{TP_c}{TP_c + FN_c},\ F1_c = \frac{2P_cR_c}{P_c + R_c}, \tag{11}$$

$$acc = \frac{\sum_{c=1}^{C} TP_c}{N}. \tag{12}$$

5.3 Experimental Result Analysis

In this subsection, we perform three experiments for evaluating our proposed model. Specifically, the first experiment is conducted for determining the best representation of the traffic flow. The second experiment aims for evaluating the classification performance of each model through four experimental scenarios in comparison with the-state-of-art models. The third experiment is mainly for the visualization of attention mechanism, which shows the importance of each data packet in different categories.

5.3.1 *Traffic Representation Evaluation*

In order to test the performance of the proposed model, four experimental scenarios in [8], [27] are used in this paper. The first task is the protocol encapsulated traffic identification, which is a two-category problem. Moreover, the second task is the regular encrypted traffic classification, which is a six-category problem. The third one is the protocol encapsulated traffic classification, which is also a six-category problem. The difference between second and third task is the datasets used. To elaborate, the second task uses the VPN dateset, while the third task relies on NonVPN dateset. Finally, the last task is the encrypted traffic classification, which the most difficult task, because it is a twelve-category problem. The details of aforementioned experiments are detailed in Table 2.

TABLE 2
The description of contrastive experiments.

	Description	class num
1	Protocol encapsulated traffic identification	2-classes
2	Regular Encrypted traffic classification	6-classes
3	Protocol Encapsulated traffic classification	6-classes
4	Encrypted traffic classification	12-classes

Table 3 shows the results with Attention based LSTM for different dataset processing methods. There are two main variables, whether to remove the DNS flows and whether to use all data or only L7 data. All data means the data after removing data link layer and IP address. The results show that it is better to retain the DNS flows. The reason is that DNS is used for hostname resolution, which is closely related to the host. They are easier to identify than other encrypted flows and bring some bias in the results. So we should remove the DNS flows from the dataset. In addition, the accuracy of using all data is 2%-3% higher than only using L7 data which is similar with [8]. This is because all data contains the transport layer and part of the network layer information, such as the port number and packet length, which are helpful for classification. Therefore, we will use all data to represent the traffic flow and remove the DNS flows.

TABLE 3
The accuracy of different processing methods

With DNS	Yes		No	
packet data	all data	L7 data	all data	L7 data
Exp 1	1.000	0.964	0.997	0.951
Exp 2	0.898	0.859	0.893	0.844
Exp 3	0.970	0.934	0.948	0.920
Exp 4	0.936	0.905	0.912	0.886

In order to find the best network traffic flow representation method, we perform a grid search on both the length of the traffic flow N and the length of the data packet M. The value range of N is [5, 10, 20], and that of M is [500, 1000, 1500]. The grid search performs 9 sets of experiments based on a combination of different values of N and M, while the other parameters remain unchanged. The experimental results are shown in the Table 4.

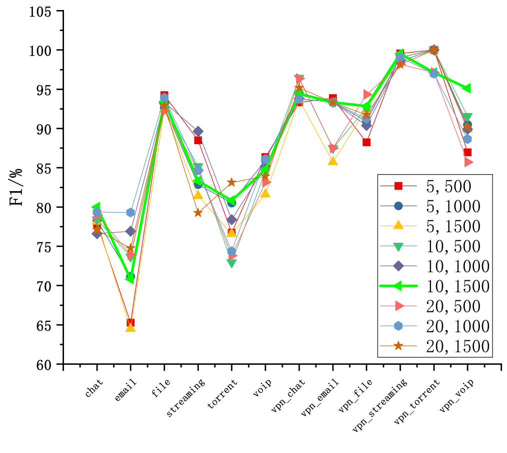


Fig. 7. The F1 score of different length of flow and packet.

In the Table 4, when $N = 10$ and $M = 1500$, the classification accuracy is highest so that we chose it as the hyper-parameters. In order to show the effect of the model in different categories in detail, we drawn Fig. 7 based on the F1 value. After analyzing the F1 score for each category considered, we can conclude that the F1 scores of NonVPN's chat and NonVPN's email protocol are very small, because some relatively short traffic flows may involve more frequent interactions. The distribution difference between traffic flows is relatively large, which leads to a poor performance of the classification. Therefore, we can conclude that when $N = 10$ and $M = 1500$, the classification of the chat class is better. We need to use a larger amount of data to learn this type of protocol with a large difference in data distribution.

5.3.2 *Performance Comparison with the State-Of-The-Art*

We choose $N = 10$ and $M = 1500$ as hyper-parameters, and use the attention-based LSTM and HAN models to perform traffic classification tasks in the above-mentioned four experimental scenarios. The experimental results are shown in Table 5, followed by more details in Appendix B.

TABLE 5
The accuracy of five models in four experimental scenarios

	Attention Based LSTM	HAN	Deep Packet	One-dim CNN	C4.5 Decision Tree	XGBoost
Exp 1	**0.997**	0.995	0.992	0.990	0.900	0.991
Exp 2	**0.893**	0.851	0.868	0.818	0.890	0.841
Exp 3	0.948	0.929	0.923	**0.986**	0.870	0.918
Exp 4	**0.912**	0.895	0.898	0.866	0.800	0.864

From Table 5, we can conclude that the models can handle the two-category classification problem correctly. As described in the introduction, the traffic is encrypted to exhibit a completely different distribution, such that the model can easily distinguish them. In addition to the C4.5 decision tree proposed in [27], all other methods work better on the VPN dataset than the Non-VPN dataset. As shown in [8], each protocol has different distributions after encryption, which can contribute to the distinction. The deep learning method can learn and extract the distribution features of encrypted traffic, thereby more accurately distinguishing the encrypted traffic.

In the scenarios 1, 3 and 4, the attention-based LSTM model achieves the best experimental results and has a significant performance improvement compared with Deep Packet [7], one-dim CNN [8], decision tree [27] and XGBoost models. The accuracy of the two-category classification is directly improved up to 0.997. The most significant improvement is the twelve-category classification in comparison to one-dim CNN, which increases by 5%, to NonVPN, which increases by nearly 7%. The overall performance of HAN is not as good as that of attention-based LSTM, while it is still much

TABLE 4
The accuracy result of different length of flow and packet.

parameter	5,500	5,1000	5,1500	10,500	10,1000	10,1500	20,500	20,1000	20,1500
acc	0.891	0.897	0.891	0.904	0.908	**0.912**	0.899	0.906	0.902

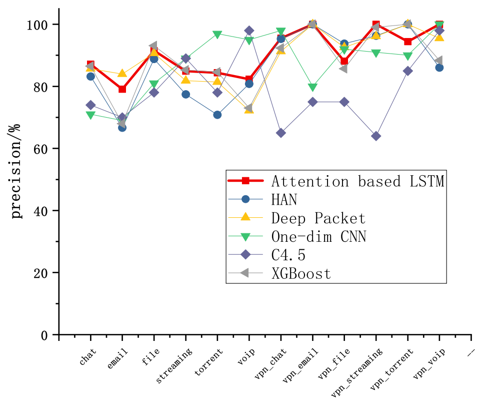


Fig. 8. The precision of all four experiments.

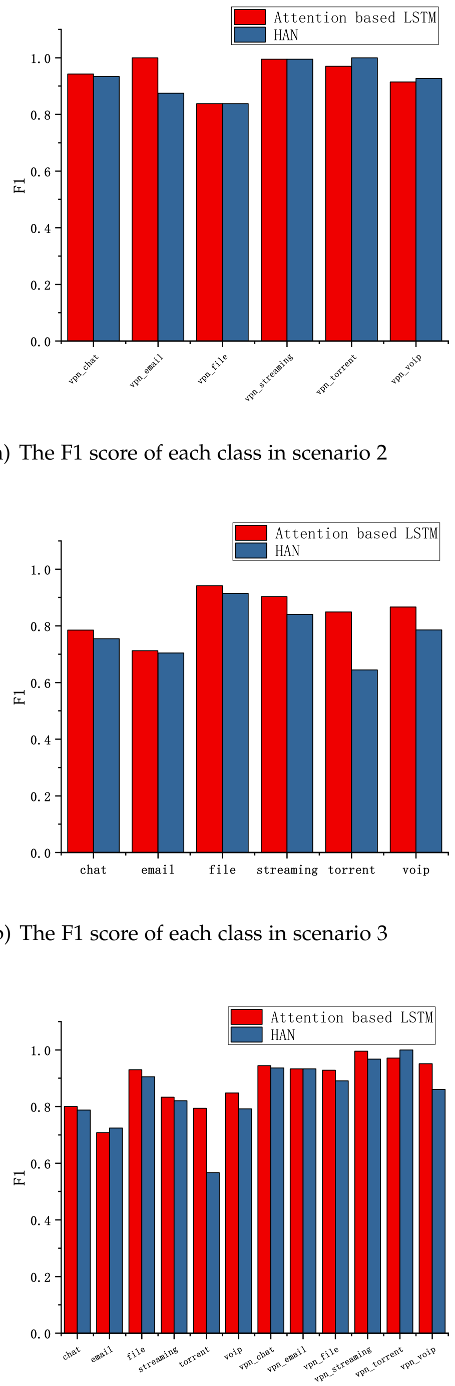
better than that of one-dim CNN and decision tree models in the scenarios 1 and 4. But the performance of HAN in two six-category classification problem is poorer than that of the one-dim CNN. The results of the Deep packet model are the closest to Attention based LSTM but better than the HAN model. The Deep Packet model classifies traffic flow using two layers CNN and seven fully connected layers. It's a little complicated so that the speed of Deep Packet is slower. In terms of processing speed, XGBoost is the fastest, then one-dim CNN, Attention based LSTM, Deep Packet, HAN. The training runtime of these models are shown in Table 6. We can see that the time consumption of HAN is very large, mainly because the length of its first LSTM is too long, while other models are relatively faster.

TABLE 6
The training runtime of five models in experimental 4

	Attention Based LSTM	HAN	Deep Packet	One-dim CNN	XGBoost
one batch	0.05s	1.48s	0.1s	0.02s	-
total	1783.6s	53149s	3600s	720s	300s

In order to analyze the experiment results in detail, we show in Fig. 8 the precision of a model in different categories. It can be seen that the Attention based LSTM performs the best. However, its precision in the streaming and VoIP categories is lower than ond-dim CNN. This is because the traffic in these two categories is too huge for LSTM to learn the long-term relationship.

In addition, the overall performance of the proposed two

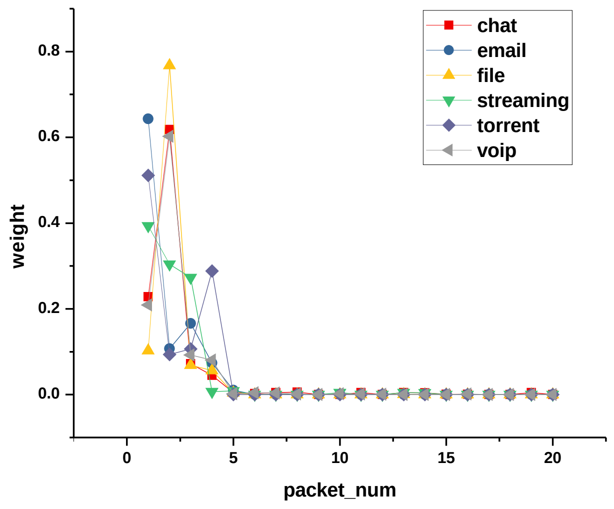


(a) The F1 score of each class in scenario 2

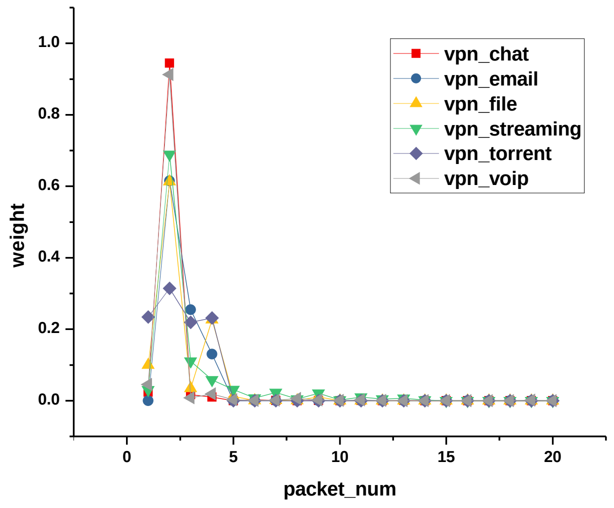
(b) The F1 score of each class in scenario 3

(c) The F1 score of each class in scenario 4

Fig. 9. The F1 score of each class, where the red bar represents F1 score of attention-based LSTM model, while the blue bar represents that of HAN model.



(a) The weight distribution of different packets in NonVPN dataset.



(b) The weight distribution of different packets in VPN dataset.

Fig. 10. The weight distribution of different packets.

models has a relatively high improvement compared with the previous models with the following two reasons.

- Firstly, we treat the network traffic as time-series data and use the LSTM for processing. In compared to the CNN model, RNN is capable of learning the relationship between adjacent packets, where each hidden layer state records relevant information of all previous packets in order to learn its long-term dependency;
- Secondly, we use the attention mechanism to improve the accuracy of the LSTM classification. The weighted summation of all historical state information is used as the final coding vector. Compared with the use of the latest hidden layer vector, more historical information is included.

In order to clearly show the classification effect of attention-based LSTM and HAN, we transform the F1 scores of each class in the scenario 2,3 and 4 into bar graphs, which are shown in Fig. 9.

5.3.3 *Packet Importance Distribution*

For the sake of exploring the importance of each packet in the final classification, we need to visualize the weight value α_i of each packet. The result is shown in Fig. 10. The test set is divided according to the category, and the attention vector c is calculated to obtain the average attention vector of the entire data set. Each category has a different attention vector, which represents the contribution of each packet imposed on different categories. The basic focus is on the first four data packets, because the packets at the end of the traffic flow hardly contribute features except those belong to protocols, such as streaming, torrent and file. As we all know, the first few packets carry more protocol-related information in the transmission of the traffic, such as the three-way handshake of the TCP protocol. In other words, our model does learn this important feature and increase the weight of the first few packets. In addition, we also analyze the importance of different bytes of data. Similar to the distribution of packets, important bytes are concentrated in the header of the data. This is because the bytes that contain category information are often located at the header of each packet in traffic.

6 Summary and Future Works

In this paper, the traffic flow was considered as time-series and stored in the form of a matrix, which facilitated the use of deep learning models for classification. Moreover, attention-based LSTM and HAN neural network architectures were constructed for traffic classification, and both of them outperformed the machine learning based method proposed in [27] and the one-dim CNN method proposed in [8]. In addition, the attention mechanism can also facilitate the visualization of training results for further analyzing the importance of each data packet in the presence of different protocol types.

In the future, we will improve our work from the following three perspectives. First, the current model only used RNN, while we hope to build more complex models, such as CNN and RNN fusion models, where CNN is used to extract traffic features, and RNN is used to learn timing features. Second, we will try different attention mechanisms including the hard attention and the local attention. Finally, we are also intended to propose new data sets that are more suitable for using deep learning methods.

Acknowledgment

This work was supported in part by the Shandong Provincial Natural Science Foundation, China (Grant No. ZR2014FQ018), in part by the BUPT-SICE Excellent Graduate Students Innovation Fund, in part by the National Natural Science Foundation of China (Grant No. 61471056), and in part by the China research project on key technology strategy of infrastructure security for information network development.

References

[1] K. Gai, M. Qiu, and H. Zhao, “Privacy-preserving data encryption strategy for big data in mobile cloud computing,” *IEEE Transactions on Big Data*, pp. 1–1, 2018.

[2] Z. Cao, G. Xiong, Y. Zhao, Z. Li, and L. Guo, "A survey on encrypted traffic classification," in *International Conference on Applications and Techniques in Information Security*, (Berlin, Heidelberg), pp. 73–81, 2014.

[3] R. Alshammari and A. N. Zincir-Heywood, "Investigating two different approaches for encrypted traffic classification," in *2008 Sixth Annual Conference on Privacy, Security and Trust*, pp. 156–166, Oct 2008.

[4] R. Alshammari and A. N. Zincir-Heywood, "Machine learning based encrypted traffic classification: Identifying ssh and skype," in *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, pp. 1–8, July 2009.

[5] M. Dusi, A. Este, F. Gringoli, and L. Salgarelli, "Using gmm and svm-based techniques for the classification of ssh-encrypted traffic," in *2009 IEEE International Conference on Communications*, pp. 1–6, June 2009.

[6] Z. Wang, "The applications of deep learning on traffic identification," *BlackHat, USA*, 2015.

[7] 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.

[8] 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 International Conference on Intelligence and Security Informatics (ISI)*, pp. 43–48, July 2017.

[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. 18042–18050, 2017.

[10] W. Wang, Y. Sheng, J. Wang, X. Zeng, X. Ye, Y. Huang, and M. Zhu, "Hast-ids: Learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection," *IEEE Access*, vol. 6, pp. 1792–1806, 2018.

[11] 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, pp. 355–374, Sep. 2015.

[12] A. Vlăduţu, D. Comăneci, and C. Dobre, "Internet traffic classification based on flows' statistical properties with machine learning," *International Journal of Network Management*, vol. 27, p. e1929, May 2017.

[13] T. T. T. Nguyen and G. Armitage, "A survey of techniques for internet traffic classification using machine learning," *IEEE Communications Surveys Tutorials*, vol. 10, pp. 56–76, Fourth 2008.

[14] P. Casas, A. D'Alconzo, T. Zseby, and M. Mellia, "Big-dama: big data analytics for network traffic monitoring and analysis," in *Proceedings of the 2016 workshop on Fostering Latin-American Research in Data Communication Networks*, pp. 1–3, ACM, 2016.

[15] D. L. Quoc, V. D'Alessandro, B. Park, L. Romano, and C. Fetzer, "Scalable network traffic classification using distributed support vector machines," in *IEEE International Conference on Cloud Computing*, 2015.

[16] Z. Yuan and C. Wang, "An improved network traffic classification algorithm based on hadoop decision tree," in *Online Analysis & Computing Science*, 2016.

[17] G. Aceto, D. Ciuonzo, A. Montieri, V. Persico, and A. Pescapé, "Know your big data trade-offs when classifying encrypted mobile traffic with deep learning,"

[18] C. Bacquet, K. Gumus, D. Tizer, A. N. Zincir-Heywood, and M. I. Heywood, "A comparison of unsupervised learning techniques for encrypted traffic identification," *Journal of Information Assurance and Security*, vol. 5, pp. 464–472, Sep. 2010.

[19] M. Zhang, H. Zhang, B. Zhang, and G. Lu, "Encrypted traffic classification based on an improved clustering algorithm," *Communications in Computer & Information Science*, vol. 320, pp. 124–131, 2012.

[20] A. H. Lashkari, G. D. Gil, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of tor traffic using time based features," in *International Conference on Information Systems Security and Privacy*, (Porto, Portugal), pp. 253–262, 2017.

[21] G. Sun, Y. Xue, Y. Dong, D. Wang, and C. Li, "An novel hybrid method for effectively classifying encrypted traffic," in *2010 IEEE Global Telecommunications Conference GLOBECOM 2010*, pp. 1–5, Dec 2010.

[22] K. Shahbar and A. N. Zincir-Heywood, "How far can we push flow analysis to identify encrypted anonymity network traffic?," in *NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium*, pp. 1–6, April 2018.

[23] 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)*, pp. 439–454, March 2016.

[24] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé, "Multi-classification approaches for classifying mobile app traffic," *Journal of Network and Computer Applications*, vol. 103, pp. 131–145, 2018.

[25] A. Scherrer, N. Larrieu, P. Owezarski, P. Borgnat, and P. Abry, "Non-gaussian and long memory statistical characterizations for internet traffic with anomalies," *IEEE Transactions on Dependable and Secure Computing*, vol. 4, pp. 56–70, Jan 2007.

[26] A. F. Santos, "Network traffic characterization based on time series analysis and computational intelligence," *Journal of Computational Interdisciplinary Sciences*, vol. 2, no. 3, pp. 197–205, 2011.

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

[28] B. Yamansavascilar, M. A. Guvensan, A. G. Yavuz, and M. E. Karsligil, "Application identification via network traffic classification," in *2017 International Conference on Computing, Networking and Communications (ICNC)*, pp. 843–848, Jan 2017.

[29] S. Rezaei and X. Liu, "Deep learning for encrypted traffic classification: An overview," 2018.

[30] W. Wang, M. Zhu, X. Zeng, X. Ye, and Y. sheng, "Malware traffic classification using convolutional neural network for representation learning," in *2017 International Conference on Information Networking (ICOIN)*, pp. 712–717, Jan 2017.

[31] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapè, "Mobile encrypted traffic classification using deep learning," in *2018 Network Traffic Measurement and Analysis Conference (TMA)*, pp. 1–8, June 2018.

[32] R. Li, X. Xiao, S. Ni, H. Zheng, and S. Xia, "Byte segment neural network for network traffic classification," in *2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS)*, pp. 1–10, June 2018.

[33] A. Graves, *Long Short-Term Memory*, pp. 37–45. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012.

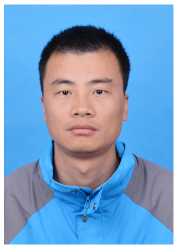
[34] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014.

[35] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, (San Diego, CA), pp. 1480–1489, Jun. 2016.

[36] Y.-A. Chung, H.-T. Lin, and S.-W. Yang, "Cost-aware pre-training for multiclass cost-sensitive deep learning," 2015.

Haipeng Yao (M'16) is an Associate Professor in Beijing University of Posts and Telecommunications. Haipeng Yao received his Ph.D. in the Department of Telecommunication Engineering at University of Beijing University of Posts and Telecommunications in 2011. He has been engaged in research on future internet architecture, network AI, Big Data, cognitive radio networks, and optimization of protocols and architectures for broadband wireless networks. He has published more than 80 papers in prestigious peer-reviewed journals and conferences.

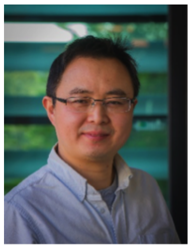
Chong Liu received his bachelor degree from the school of information and communication engineering, Beijing University of Posts and Telecommunications in 2016. He is pursuing his master degree at the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications. His research interests include deep learning, network AI and big data for networking.



Peiying Zhang received his master degree from the College of Computer & Communication Engineering, China University of Petroleum (East China) in 2006. He received his PhD degree at the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications. He is currently an associate professor with the College of Computer & Communication Engineering, China University of Petroleum (East China). His research interests include semantic computing, network artificial intelligence, network virtualization and future network architecture.

Sheng Wu (S'13-M'14) received the B.E. and M.E. degrees from Beijing University of Posts and Telecommunications, Beijing, China, in 2004 and 2007, respectively, and the Ph.D. degree in electronic engineering from Tsinghua University, Beijing, China, in 2014. He was a postdoctoral researcher in the Tsinghua Space Center at Tsinghua University, Beijing, China. He is currently a lecturer with Beijing University of Posts and Telecommunications, Beijing, China. His research interests are mainly in iterative detection and decoding, channel estimation, massive MIMO, and satellite communications.

Chunxiao Jiang (S'09-M'13-SM'15) received the B.S. in information engineering from Beihang University in Jun. 2008 and the Ph.D. in electronic engineering from Tsinghua University in Jan. 2013, both with the highest honors. From Feb. 2013 - Jun.2016, Dr. Jiang was a Postdoc in the Department of Electronic Engineering Tsinghua University, during which he visited University of Maryland College Park and University of Southampton. He is a recipient of the IEEE Globecom Best Paper Award in 2013, the IEEE GlobalSIP Best Student Paper.



Shui Yu is currently a full Professor of School of Software, University of Technology Sydney, Australia. Dr Yu's research interest includes Security and Privacy, Networking, Big Data, and Mathematical Modelling. He has published two monographs and edited two books, more than 200 technical papers, including top journals and top conferences, such as IEEE TPDS, TC, TIFS, TMC, TKDE, TETC, ToN, and INFOCOM. Dr Yu initiated the research field of networking for big data in 2013. His h-index is 32. Dr Yu actively serves his research communities in various roles. He is currently serving the editorial boards of IEEE Communications Surveys and Tutorials, IEEE Communications Magazine, IEEE Internet of Things Journal, IEEE Communications Letters, IEEE Access, and IEEE Transactions on Computational Social Systems. He has served more than 70 international conferences as a member of organizing committee, such as publication chair for IEEE Globecom 2015, IEEE INFOCOM 2016 and 2017, TPC chair for IEEE BigDataService 2015, and general chair for ACSW 2017. He is a Senior Member of IEEE, a member of AAAS and ACM, the Vice Chair of Technical Committee on Big Data of IEEE Communication Society, and a Distinguished Lecturer of IEEE Communication Society.