frontiers in Neurorobotics

ORIGINAL RESEARCH
published: 19 March 2021
doi: 10.3389/fnbot.2021.648374

Robot Communication: Network Traffic Classification Based on Deep Neural Network

Mengmeng Ge, Xiangzhan Yu and Likun Liu*

School of Cyberspace Science, Harbin Institute of Technology, Harbin, China

With the rapid popularization of robots, the risks brought by robot communication have also attracted the attention of researchers. Because current traffic classification methods based on plaintext cannot classify encrypted traffic, other methods based on statistical analysis require manual extraction of features. This paper proposes (i) a traffic classification framework based on a capsule neural network. This method has a multilayer neural network that can automatically learn the characteristics of the data stream. It uses capsule vectors instead of a single scalar input to effectively classify encrypted network traffic. (ii) For different network structures, a classification network structure combining convolution neural network and long short-term memory network is proposed. This structure has the characteristics of learning network traffic time and space characteristics. Experimental results show that the network model can classify encrypted traffic and does not require manual feature extraction. And on the basis of the previous tool, the recognition accuracy rate has increased by 8%

Keywords: traffic classification, capsule neural network, encrypted traffic, network security, deep learning

OPEN ACCESS

Edited by:
Zhaoquan Gu,
Guangzhou University, China

Reviewed by:
Tingting Chen,
California State Polytechnic University, Pomona, United States
Longfei Wu,
Fayetteville State University, United States
Aiping Li,
National University of Defense Technology, China

****Correspondence:***
Xiangzhan Yu
yxz@hit.edu.cn

Received: *31 December 2020*
Accepted: *18 February 2021*
Published: *19 March 2021*

Citation:
Ge M, Yu X and Liu L (2021) Robot Communication: Network Traffic Classification Based on Deep Neural Network. Front. Neurorobot. 15:648374. doi: 10.3389/fnbot.2021.648374

INTRODUCTION

With the rapid development of technology, humanoid robots can do more things on behalf of people, such as helping people guide paths, serving coffee, and turning on lights. While humanoid robots liberate people's labor, there are also some risks of security and privacy leakage in these processes.

Robots need to interact with people or server commands when they are working (Gleeson et al., 2013; Mavridis, 2015). When robots and people interact through voice, everyone can hear the commands issued by people. When people want to hide the behavior and content of the commands, people can Use codes instead, such as a cough that means a command to turn on the light. This is the easiest way to hide the content of communication between humans and robots. When the server communicates with the robot, it is impossible for the server to cough and issue a command like a human (Su et al., 2020). He will put the control command in the network message and send it to the robot in a specific protocol format. When a stranger repeatedly observes the behavior of coughing, the robot will light up. He speculates that the coughing behavior may correspond to the command to turn on the light. Therefore, by observing the communication process between the control server and the robot in the network, and through learning and training, the communication protocol between the server and the robot can be identified, and further, the command line in the communication process between the server and the robot can be inferred as the type (Kanda et al., 2010). In this article, we have studied the protocol identification of network messages, which can identify the type and protocol of network communication traffic, which is of great significance to

the discovery of malicious network attack traffic in the communication process of humanoid robots. It can also be used for web traffic detection (Tian et al., 2020) and IoT traffic detection (Shafiq et al., 2020).

The development of traffic classification technology has gone through three stages: port-based, payload-based, and flow-based statistical characteristics. Port-based classification methods infer the types of mobile services or applications by assuming that most applications always use "well-known" TCP or UDP port numbers (Li et al., 2000; Hjelmvik and John, 2009; Wang et al., 2017). However, the emergence of port masquerading, random ports, and tunneling technologies quickly lost these methods Effectiveness. The payload-based method, that is, DPI (Deep Packet Inspection) (Moore and Papagiannaki, 2005; Finsterbusch et al., 2013; Wang et al., 2019) technology cannot handle encrypted traffic because it needs to match the content of the data packet and has a high computational overhead (Madhukar and Williamson, 2006). In order to try to solve the problem of encrypted traffic identification, a flow-based method has emerged, which usually relies on statistics or time series features and uses machine learning (Zuev and Moore, 2005; Liu et al., 2007; Fan and Liu, 2017; Shafiq et al., 2018) algorithms, such as Naive Bayes, Support Vector Machine (Li et al., 2007; Yuan et al., 2010; Groleat et al., 2012; Ebrahimi et al., 2017), Decision tree, random forest (Siahaan et al., 2019), k nearest neighbor (KNN)(Este et al., 2009; Wu et al., 2015; Sun et al., 2018). In addition, some statistical models, such as Gaussian Mixture Model (Alizadeh et al., 2015; Kornycky et al., 2016; Pacheco et al., 2018) and Hidden Markov Model (Yin et al., 2012), are used to identify and classify encrypted traffic.

Although classic machine learning methods can solve many problems that cannot be solved by methods based on ports and payloads, it still has some limitations: (1) It is difficult to obtain manually extracted traffic characteristics, and these characteristics always depend on domain experts' experience. Therefore, it is impossible to automatically extract and select features, which will cause great uncertainty and confusion in classic machine learning methods when ML is applied to mobile service traffic classification. (2) Flow characteristics are easily outdated quickly and need to be constantly updated. (3) How to combine a large, easily accessible unlabeled data set with some expensive labeled data sets for traffic classification to reduce the need for labeled data is a very critical research topic. (4) For traffic classification tasks, category imbalance is not a small problem. However, current data enhancement methods cannot accurately generate samples as close to the original data distribution as possible.

Unlike most traditional machine learning algorithms, deep learning (Gu et al., 2020) can perform automatic feature extraction without manual intervention. This paper uses the algorithm based on the capsule convolutional neural network (Vinayakumar et al., 2017; Rezaei and Liu, 2019) and the self-attention LSTM neural network to identify the encrypted network traffic (Fu et al., 2016; Si et al., 2019). The results show that this method does not require manual feature extraction and has excellent classification effects.

RELATED BACKGROUND CONTENT

Capsule Neural Network

Convolutional Neural Networks (CNN) have good image recognition performance, but they still have some shortcomings. When the photos are crowded and blurred, the classification effect will worsen, and the output of the model does not respond well to small changes in the input. The Capsule Network (CapsNet) (Xiang et al., 2018) solves some of the traditional neural convolutional network problems because the capsule network is composed of directional neuron groups, capsules instead of neurons. The traditional training feature of each neuron, learned which area in the spatial feature is not fixed, it is entirely random, but in the capsule network, each neuron group learns a fixed area in a certain area in the picture Features, such as the eyes and nose of a human face. The capsule network is also composed of multiple layers. As shown in **Figure 1A**, the vector capsule is at the bottom of the network (Zhu et al., 2019). The capsule network (Deng et al., 2018) also has a perceptual domain, just like the traditional CNN. However, for each vector capsule, their perceptual domain is a fixed part of the spatial feature, and they only learn the features of that region. During the training process, the parameters are modified continuously to improve the accuracy of classification and recognition. Further, some small capsules will be gathered into large capsules, called routing capsules, to learn more extensive spatial features. For example, vector capsules identify network traffic packet space features, and routing capsules identify the space of multiple network traffic packets. And then, the characteristics of the entire flow or stream are recognized.

Anti-fragility

The capsule network output is not the individual value output by the traditional artificial neuron but the output vector. The parameters of the dynamic combination of different vector capsules in the routing can be calculated through its unique dynamic routing mechanism, which can be trained for different perspectives.

Robustness

Redundant homomorphism can solve complex problems. Many small features will be trained by one Capsule. Although the parameters between capsules are independent of each other, many capsules will train similar substructures. Simultaneously, the high-level capsule can learn the relationship between the various structures of the bottom-level capsule. Even though the image to be recognized becomes blurred or shifted and other perspective changes, it can be correctly recognized through this redundant network structure.

Interpretability

It is possible to know what each capsule is responsible for training and which sub-structural features each capsule recognizes so that each parameter of the neural network is no longer a complete black box.

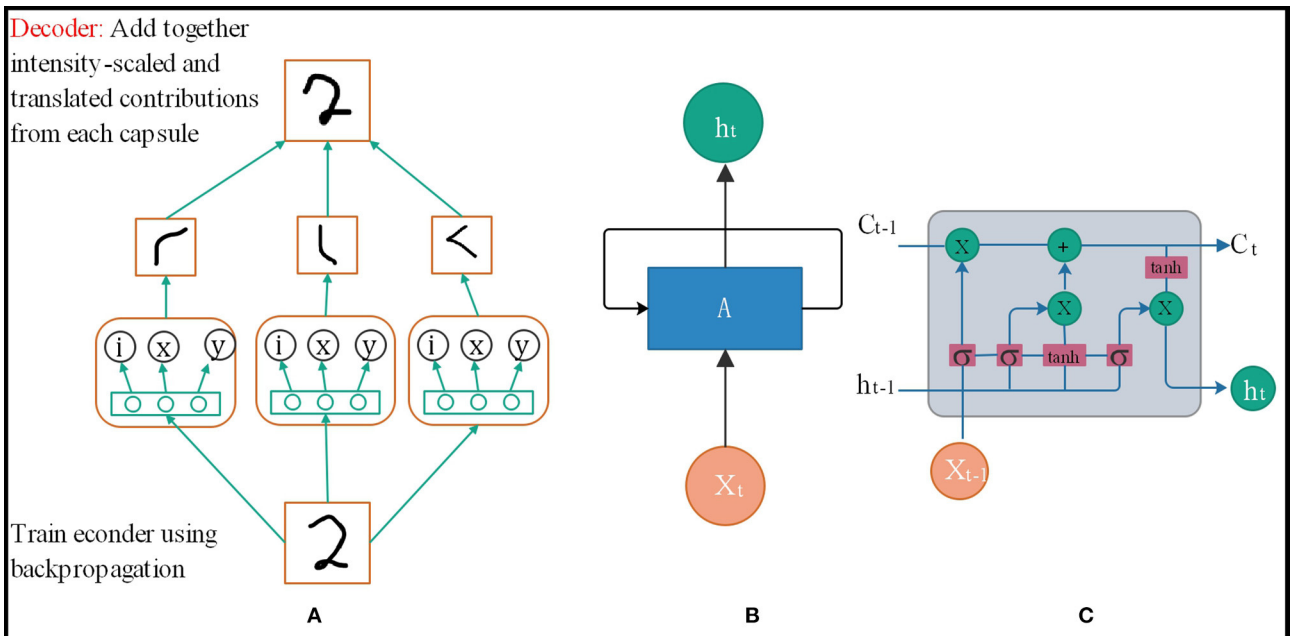

FIGURE 1 | (A) Capsule neural network structure. **(B)** RNN neural network structure. **(C)** LSTM neural network structure.

Self-Attention Long Short-Term Memory

When analyzing the flow characteristics between network sessions, it can be found that it is more like a sentence that conforms to the special established rules, except that it is composed of bytes to form a data packet and then a session. It is very similar to the structure of natural language problems where words form sentences and sentences form paragraphs. The principle of using a time series neural network to identify which application a session belongs to is also similar to the principle of sub-classification. The essence is to use the form of an array instead of the data itself as input. A similar time-series relationship of similar data is obtained through the model to achieve classification.

As shown in **Figure 1B**, the traditional RNN network model cannot fully handle the timing problem because it needs to determine the parameters before it can be predicted, but in the process of training the parameters of the neural network model, the chain derivation rule to determine the gradient increment of the parameter is essential of. In this case, when the problem sequence to be dealt with is relatively long, it will be relatively large, and the upper bound of the derivative of the common activation function is, so when it is very large, there will be many less than the numerical value in the chain derivation rule. When the time sequence is very long, the parameter increments updated by the chain derivation rule will be close, causing the parameters to be unable to be updated. In other words, RNN can handle timing problems, but when the sequence of timing problems is very long, the information cannot be very effective Was saved, so researchers proposed an improved model LSTM[38]. The network structure is shown in **Figure 1C**.

The reason why LSTM can solve the long time sequence problem of t is because the cell state C_i is used to assist in the transfer of data to time t, instead of relying on the hidden layer information at time h_{t-1}, and the update formula of cell state C_i can also be used. Knowing that it is updated through addition. The advantage of this is that when the chain rule is updated, the cell state C_i will not appear as the updated parameter increment in the RNN chain derivation rule will be close to 0.

The attention mechanism was first proposed in the field of visual images, and has gradually been widely used in natural language since 2014. Today, the combination of various attention mechanisms and deep learning network models has achieved good results in natural language problems. As a result, this article also considers adding a self-attention mechanism when using the time-series deep learning model LSTM to process traffic.

Self-attention (Tao et al., 2020). The above process can be abstracted as the calculation of similarity between query, key and value, which can be roughly divided into three stages:(1) query and key_i use the similarity function that matches the task to calculate the similarity, and get the parameter s_i. (2) Normalize s_i with $softmax()$ to get α_i. (3) After multiplying the corresponding α_i and $value_i$, and then summing, the self-attention value can be finally obtained. The calculation process is as follows:

$$f(Q, K_i) = Q^T K_i$$

$$\alpha_i = softmax\left(f(Q, K_i)\right) = \frac{\exp\left(f(Q, K_i)\right)}{\sum_i \exp\left(f(Q, K_i)\right)}$$

$$self - attention(Q, K, V) = \sum_i \alpha_i V_i (Q = K = V)$$

FRAMEWORK AND METHODS

Our framework logically consists of two parts: the "pre-processor" and the "traffic classifier." The former has performed all tasks that allow us to model the network traffic into data, which can easily be handled by a deep learning model. The latter performs specific classification tasks. Before using a deep learning traffic classifier, it must be trained with a large amount of labeled traffic data. The traffic data we use comprises the public dataset ISCX2012 and the data generated by artificially stimulating the apps.

Network Traffic Pre-processing

There are two traffic forms: flow and session. Usually, we use five tuples to determine flow and session. Flow: a time-ordered sequence of packets exchanged between two peers during a single TCP session. f $= < p_1, p_2, \ldots, p_N >$, N is the number of packets consist of flow f. packet $p_i = (a_i,\ l_i,\ t_i)$. a_i is a 5-tuple that consists of source address, source port, a destination address, destination port, and the protocol type. l_i is the length of the packet p_i and t_i is the time of packet p_i arrival. The total flow length L $= \sum_{i=1}^{n} l_i$ and $t_1 \leq t_2, \ldots, \leq t_n$. Session and flow are similar. The difference is that the source and destination addresses can be exchanged, which is a bidirectional data flow. Network traffic can be divided into four levels, application layer, transport layer, network layer, and all layers. The input data of the traffic classifier, a combination of different traffic forms and different network layers, are eight types. The pre-processing is shown as **Figure 2**.

Packers Filtering

Due to network congestion, traffic load balancing, or other unpredictable network behaviors, data packets may be lost and arrive out of order. When TCP detects these problems, it will retransmit network data packets and rearrange out-of-order data packets. Repeated transmission packets will affect the characteristics of network traffic, and network flows without data content will not use in identification. Therefore, we filter out retransmissions and packets with only ACK flags with zero payloads. The sequence of network packets has an essential impact on recognition. We will rearrange out-of-order data packets to obtain a correct network flow sequence.

Traffic Data Processing

Traffic classification should remove data related to the hardware environment and network environment, such as IP information in the network layer and MAC addresses in all layers. Therefore, training data and test data may have different physical addresses and IP addresses. We need to remove these features that may change to ensure the feature consistency of the training data and test data.

The input length of the classifier is consistent to ensure the correct subsequent recognition. At the same time, the selected data length has a significant impact on the recognition result. Through analysis of the contribution of bytes in the stream, we found that the first 400 bytes in the stream have an impact on the traffic classification because the recognition contribution is the highest. Therefore, we choose 400 as the input of the convolutional neural network. Data streams with a length of <400 will be dropped, and data streams with a length of more than 400 bytes will only take the first 400 bytes of the data stream as classifier input.

Many previous research works have shown that CNN neural network has a good classification performance for image recognition. Here we take the first 400 bytes of a network stream, and we can get a 20*20 two-dimensional matrix. Each value range of the matrix is 0–255, so each value can be regarded as the gray value of a pixel. Thus, picture data containing the first 400 bytes of information in the data stream can be obtained. When the network traffic is processed and converted into pictures, we get the features of the traffic stream. This paper uses the CapsuleNet neural network, capsule represented by a vector can learn the spatial feature relationship of the flow graph well.

Traffic Classification

Convolutional Neural Network

The first 400 bytes of encrypted data traffic is converted into a 20*20 grayscale image, which is the input of the traffic classifier. The Inception-CapsuleNet network designed in this paper is divided into nine layers. The first four layers extract the characteristics of the traffic, the middle four layers combine the characteristics, and the last layer is the category output layer, as shown in **Table 1**. Before the grayscale image entry the model, the mean value is zeroed first so that the model converges quickly.

The first layer is a convolutional layer, which extracts local features of grayscale images. In order to learn more about the local features of the input data, the step size is 1. The second layer uses batch normalization, which can prevent the data distribution from changing greatly after passing through the previous layer, and can avoid the gradient disappearance and overfitting problems. In the third inception layer, convolution kernels of different sizes are used for feature processing.

Convolutions of different sizes can extract features from different images in different fields of view, which can increase the ability of the network to extract features. Finally, the outputs of different convolution kernels are spliced together to obtain a feature with a dimension of 256. There are 32 capsules in the PrimaryCaps layer, each of which will convolve all the inputs of the previous layer. Here the activation parameter is squash, and an out tensor is u_i, with a shape of 4*4*8*32. The input data in the DigitCaps layer is the vector u_i. The calculation process is as follows:

$$\hat{u}_{j|i} = w_{ij} \cdot u_{ij}$$
$$b_{ij} = b_{ij} + \hat{u}_{j|i} \cdot v_j$$
$$c_{ij} = softmax\left(b_{ij}\right) = \frac{\exp\left(b_{ij}\right)}{\sum_k \exp\left(b_{ik}\right)}$$
$$s_j = \sum c_{ij} \cdot \hat{u}_{j|i}$$
$$v_{ij} = \frac{\| s_j \|^2}{1 + \| s_j \|^2} \cdot \frac{s_j}{\| s_j \|}$$

Where s_j is the final input, the final output vector is v_j and $b_{ij}^{(0)} = 0$. After the activation function Squash, b_{ij}, c_{ij} can be

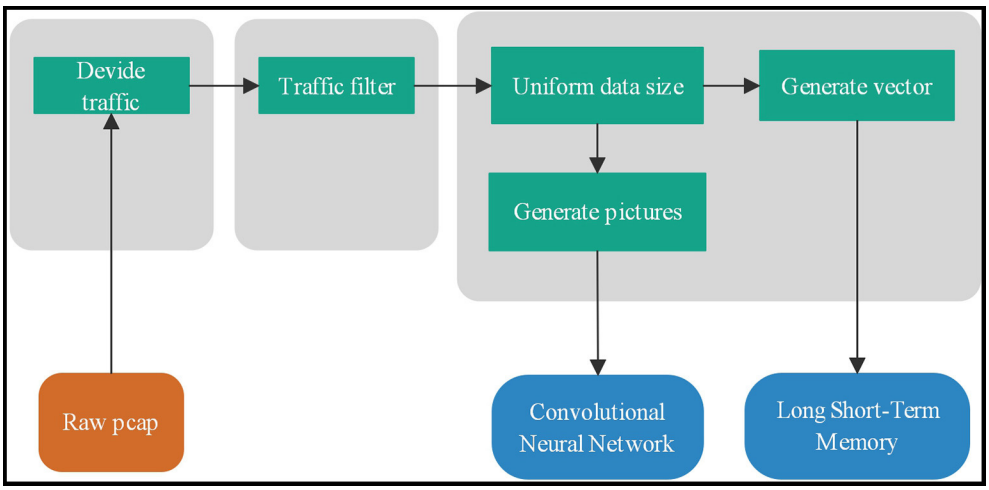

FIGURE 2 | Network traffic data packet processing process: flow division, flow filtering, uniform size and classification.

TABLE 1 | Capsule convolutional neural network.

Layer	Name	Activation function	Input size	Convolution kernel	Step	Output size
1	Condv2	ReLU	20*20*1	9*9*256	1	12*12*128
2	Batch Norm	–	12*12*128	–	–	12*12*128
3	Inception	ReLU	12*12*128	–	–	12*12*256
4	Primary Caps	Squash	12*12*256	6*6*256*8	2	4*4*8*32
5	DigitCaps	Squash	4*4*8*32	–	–	10*12
6	Full Connect	ReLu	10*12	–	–	256
7	Full Connect	ReLU	256	–	–	128
8	Full Connect	ReLU	128	–	–	64
9	Full Connect	Softmax	64	–	–	8

updated by Equations (2) and (3). The best parameter selection can be achieved by continuously repeating the above process. At the same time, in order to prevent over-fitting, the number of iterations here is selected as three, and finally, ten capsules are output, and each capsule is a 12-dimensional vector.

The following are three fully connected layer classification networks, and finally, a vector of length 64 is obtained, and the last one is a fully connected softmax activation function classifier.

Long Short-Term Memory Network

In order to make full use of the characteristics of the network flow, the network structure of this article first uses the self-encoding method to train the data packets into a unified array specification as the information representing the encrypted flow. Then use the time series neural network to extract the timing behavior characteristics of the data packet exchange process at both ends of the conversation, and use the characteristics to classify the encrypted network traffic of the application. Because the flow is in the process of data packet exchange, the encrypted content of the current data packet may be determined by the protocol in a previous data packet. The pure LSTM time series network cannot well capture the characteristic information generated by this behavior. This article will add from The attention mechanism allows each data packet unit to better correspond to its own related data packets during the training process, hoping to obtain more comprehensive network traffic characteristics and achieve higher recognition results.

The data processing process is shown in **Figure 3**. Before the model training is carried out, after the traffic is cleaned, the data packets are sorted according to the flow or session according to the packet sending time of the packet header. After the traffic enters the model, first select the number of reserved data packets, then select the reserved byte length of each data packet, and then convert multiple data packet bytes into an array vector through the method of self-encoder. Finally, it enters the model's feature extraction classifier training stage.

In the stage of training the word vector, the traditional method is to use the word form to represent the word vector. But when the number of dictionaries is too long, the word vector cannot be used in deep learning algorithms. The relevance of the word vector is very poor, so this article uses the Distributed method to train the word vector. The vector obtained by this method can be controlled to a shorter length.

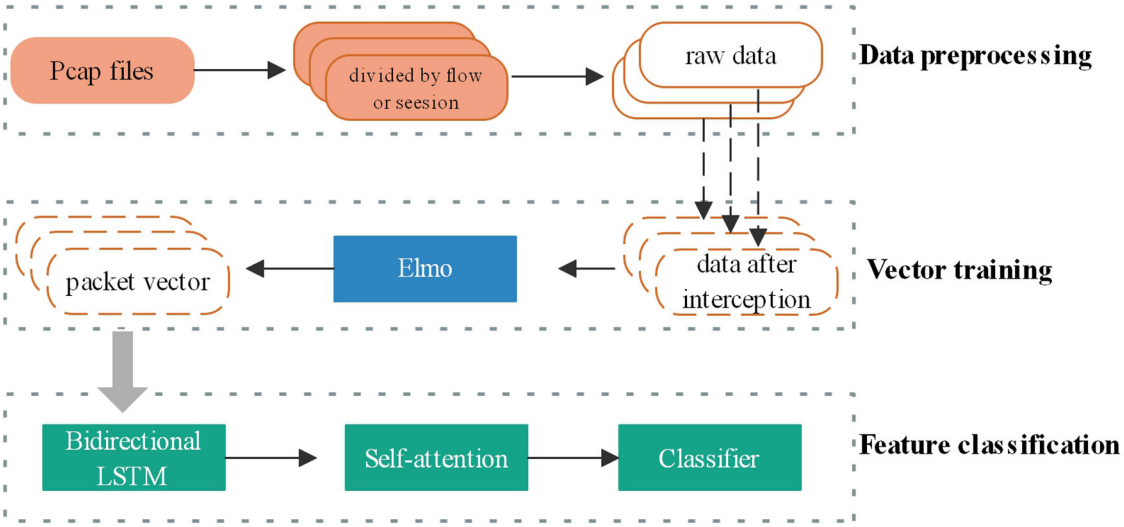

FIGURE 3 | LSTM neural network data processing process.

When training the word vector of the data packet, it is considered that the meaning of the same byte in the encrypted traffic protocol of different applications is very different. If a traditional word vector training method, such as word2vec is used to obtain a fixed word vector, this does not conform to the characteristics of the data in the encrypted traffic recognition model. Elmo's two-layer bidirectional LSTM network pre-training structure can obtain word vectors that meet the complexity of the protocol, and the same byte corresponds to different word vectors in different protocol environments.

Regarding the hexadecimal bytes in the pcap packet, this article converts it into decimal data as the original input, and then intercepts the 96 bytes of the first six packets of each stream or session as input. Less than 96 bytes, using the method of adding 0 to fill in, the form of the array converted into an array of 6*96 dimensions. Its form is similar to the way of analyzing a sentence category in text classification. For a 6-word sentence, the word vector length of each word is fixed at 96. In form, it is similar to the first 6 data contained in a stream or conversation. The information contained in the data packet represents that the two ends of the conversation are communicating through a certain language, so it is reasonable to input it into the training model of natural language word vector Elmo.

In the Elmo stage, the Embedding of the first layer of LSTM and the Embedding of the second layer of LSTM are multiplied by the corresponding weights, and the final Embedding is $< t_1, t_2, \ldots, t_n >$ Then, where n represents the number of data packets, the value here is 6, that is, 6 data packets are selected, and each data packet intercepts 96 bytes as input. The size of the LSTM of the Elmo stage is 128, that is, the size of the vector t_1 length, and then input Elmo's result data into the LSTM+Self-Attention model.

After entering the LSTM+Sefl-Attention stage, after the first layer of LSTM, the activation function is Relu, the word vector feature of the data packet is extracted and the size is 256 as the input of Batch Normalization to ensure that the data distribution remains unchanged while maintaining 256 The length remains unchanged and enters the next layer of Self-Attention. After this network layer, you can learn an encrypted session or stream. The internal structure of the intercepted data packets and the dependency relationship between the protocol features help The model identifies and classifies encrypted traffic. After passing through the Sefl-Attention network layer, the traffic length remains unchanged at 256, and then passes through a layer of Batch Normalization to enter the fully connected layer. The number of neurons in the first fully connected layer is 64, the activation function is Relu, and the second fully connected layer. The number of neurons is 7, the activation function is Softmax, which is used to finally output the probability that the encrypted traffic belongs to the target application.

CNN-LSTM Joint Network

A reasonable network structure plays an important role in the process of deep learning to identify encrypted traffic. This article draws on the idea of bagging, and designs a neural network with convolutional time series to identify encrypted traffic, as shown in **Figure 4**. The model has two inputs for the same sample. On the left side of the model, the input is the overall picture converted from the session bytes recombined from the data packet. This side of the model learns the structure information characteristics of the encryption suite of the traffic data; On the right side of the model, the input is bytes intercepted by multiple consecutive data packets in a session. The right side of the model learns the behavioral communication characteristics between traffic data, combines the two feature vectors together through splicing, and then passes through the neural network Layers are classified. Compared with the previous convolutional neural network and time series neural network, the combined

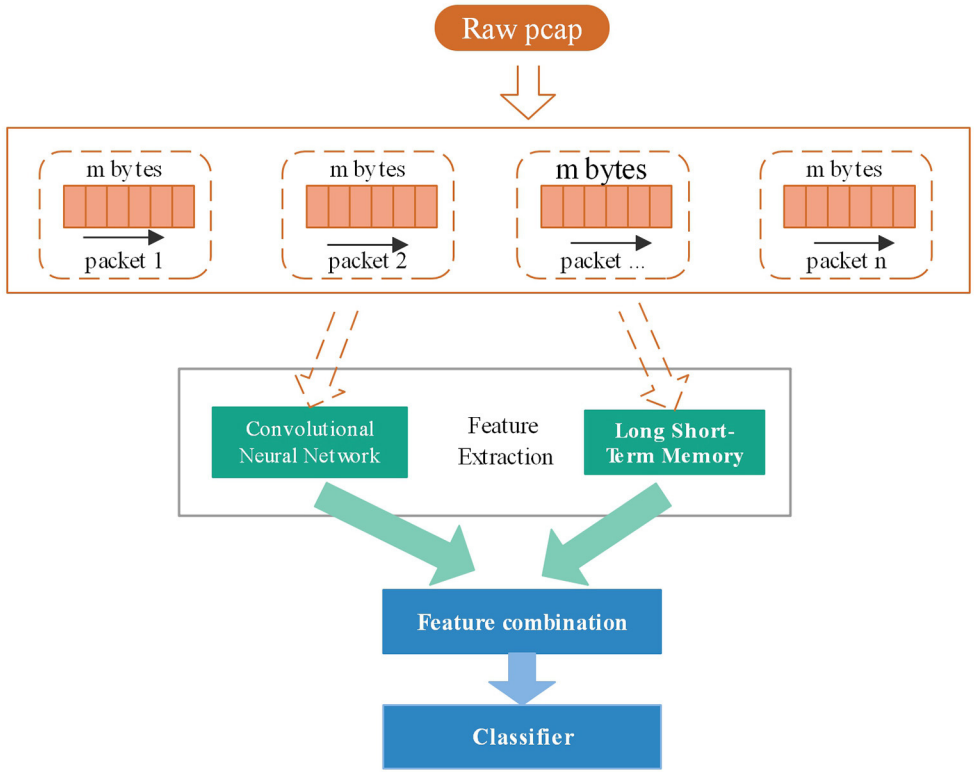

FIGURE 4 | Convolutional neural network and LSTM neural network joint processing model.

TABLE 2 | CNN network structure on the left.

layer	Name	Activation function	Convolution kernel	Kernel number	Step
1	Condv2	ReLU	3*3	256	1
2	Condv2	ReLU	3*3	128	1
3	Batch Norm	–	–	–	–
4	Inception-1	ReLU	1*1	128	1
4	Inception-2	ReLU	1*1 3*3	64	1
4	Inception-3	–	3*3 1*1	64	1
5	Inception-concact	–	–	–	–
6	PrimaryCaps	Squash	6*6*256	8	2
7	DigitCaps	Squash	–	–	–

neural network designed in this paper has an accuracy increase of nearly 4%, which further proves that a reasonable neural network structure is essential for the improvement of encrypted traffic recognition.

In **Figure 4**, two neural network structures are adopted to extract the different characteristics of encrypted traffic. The convolutional neural network on the left learns the overall structural characteristics of the traffic data packet, intercepting the first m data packets of the session, each The data packet intercepts n bytes, and each byte corresponds to two hexadecimal digital representations in the original traffic, which can be converted into a 1–255 decimal representation as input to obtain a matrix of size m*n. The process of the convolutional network refers to the network structure of session2, and uses multiple convolution kernels of different sizes to convolve the same data. The network structure is shown in **Table 2**.

The time series neural network used on the right is used to learn the communication timing characteristics between encrypted traffic data packets. Its input is the same as the data format on the left. It intercepts the first m data packets of

the session, and each data packet intercepts n bytes. A byte corresponds to two hexadecimal digital representations in the original traffic, which can be converted into a 1–255 decimal representation as input, and a two-dimensional array is obtained, so that the input representation is the same as the natural language text analysis The input method of a sentence is the same. Compared with the traditional time series network structure, this paper adds a self-attention mechanism, so that it can capture the dependence of learning data packet communication behavior and learn more timing characteristics. The network structure is shown in **Table 3**. The output data of the network structure on the left and right sides are spliced together as the input of the classifier. This part of the network structure is shown in **Table 4**.

TABLE 3 | LSTM network structure on the right.

Layer	Name	Activation function	Weight matrix
1	Elmo-1-Forward LSTM	ReLU	96*128
1	Elmo-1-ReverseLSTM	ReLU	96*128
2	Elmo-2-Forward LSTM	ReLU	128*128
2	Elmo-2-Reverse LSTM	ReLU	128*128
3	Elmo-concact	–	–
2	Two-Way LSTM	ReLU	128*256
3	Batch Norm	–	–
4	Self-Attention	–	–
5	Batch Norm	–	–

TABLE 4 | Classification layer network structure.

Layer	Name	Activation function	Length
1	Concact	–	–
2	Full Connect	ReLU	512
3	Full Connect	ReLU	64
4	Full Connect	Softmax	7

RESULT

The system of the experimental environment is Ubuntu 16.0, based on Keras running framework. RAM is 96G and video memory is 16G.

Different input byte length will affect the discrimination effect of the classifier, so choosing the appropriate byte input length is very important for the classifier. We have studied the impact of the byte input length from 50 to 750 on the classification results, as shown in **Figure 5A**. As the input byte length increases, the more features the classifier can use, the better the classification effect of the classifier. When the length increases at a certain threshold, the classification effect has not improved significantly. In order to save computing resources and time, we select the input data length as short as possible.

The choice of loss function will affect the classification effect of the classifier. The loss function calculates the distance between the probability distribution p and the probability distribution q predicted by the classifier. This article uses a loss function optimized based on focal loss *Loss(p,q)=L*.

$$L = \sum_{i=1}^{K} \sum_{j=1}^{K'} \alpha_i \left(1 - q\left(x_{ij}\right)\right)^{\gamma} p\left(x_{ij}\right) \lg\left(q\left(x_{ij}\right)\right)$$

For a certain type of sample, the higher the value of $q\left(x_{ij}\right)$ is, the smaller the value of $1 - q\left(x_{ij}\right)$ is, thus reducing the weight of this type of sample. The value of $q\left(x_{ij}\right)$ is small, and the $1 - q\left(x_{ij}\right)$ will be large, which can increases the weight of the sample recognized

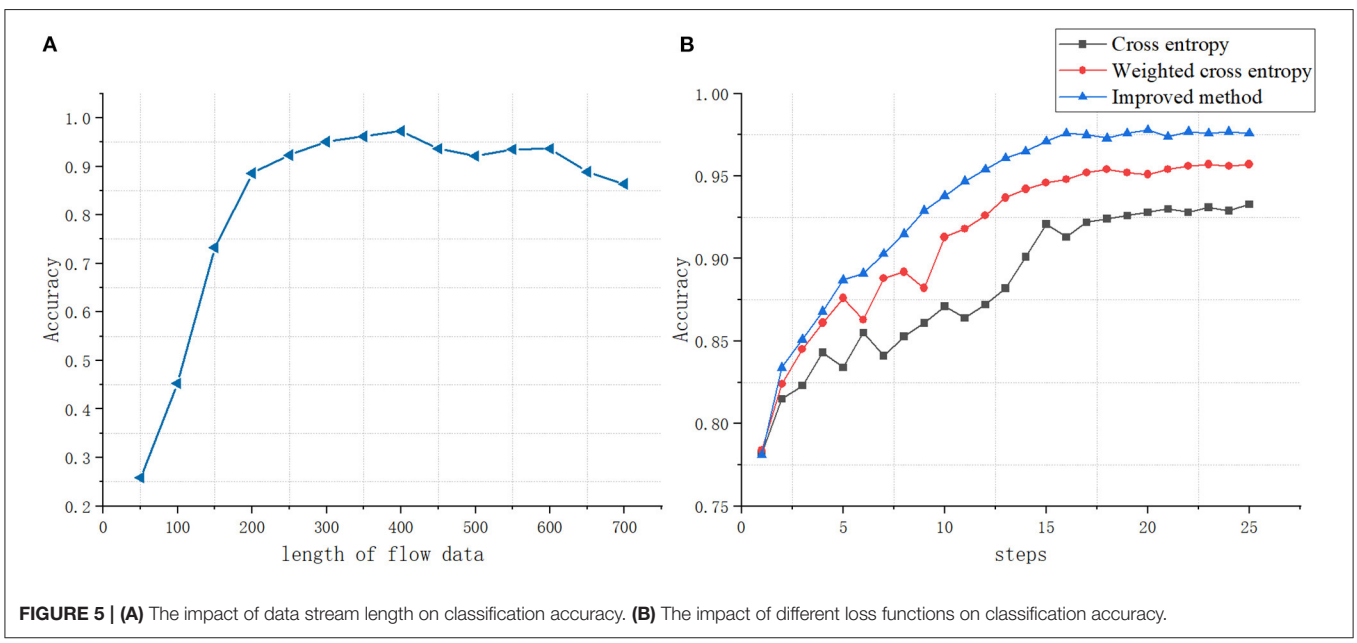

FIGURE 5 | (A) The impact of data stream length on classification accuracy. **(B)** The impact of different loss functions on classification accuracy.

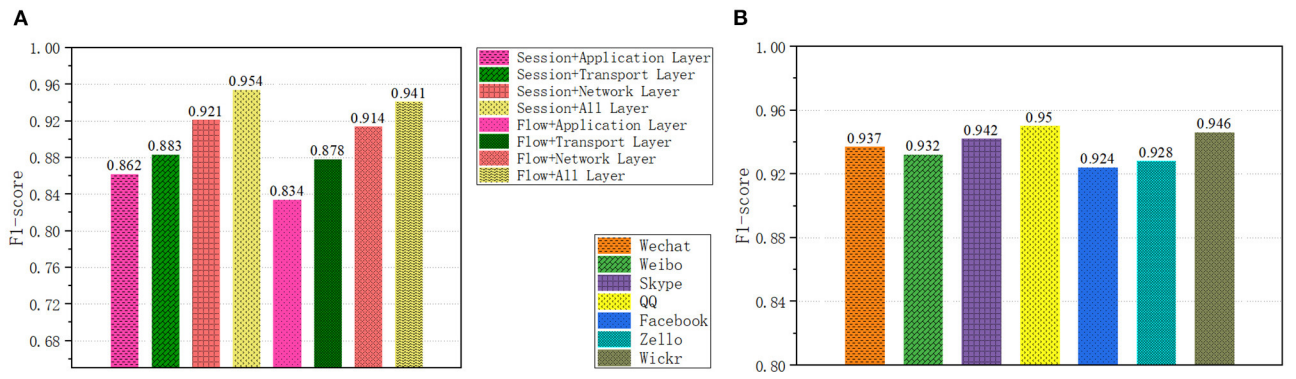

FIGURE 6 | (A) Classification results of different network traffic forms. **(B)** Application classification results of convolutional neural networks.

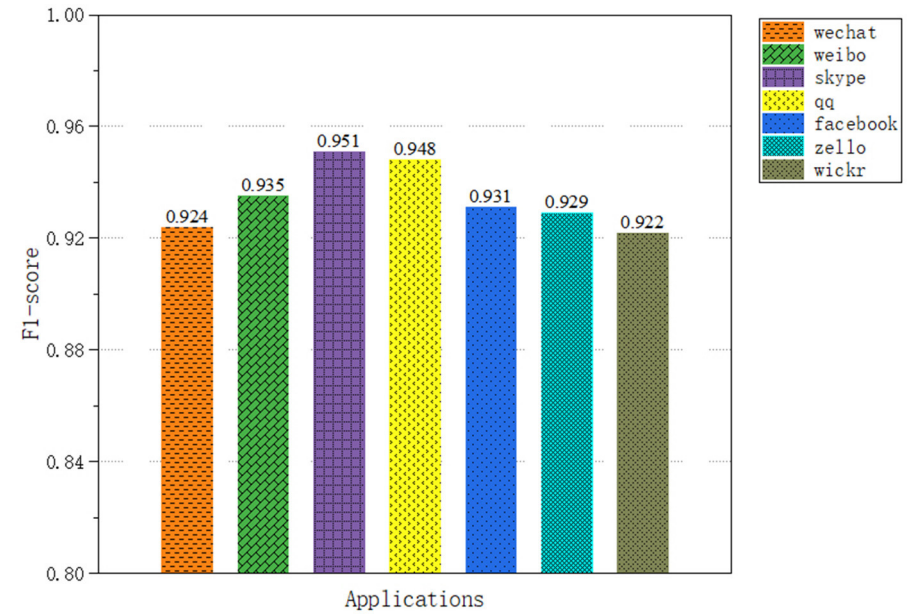

FIGURE 7 | Application classification results of LSTM neural network.

hard. The value of α_i is inversely proportional to the number of samples of each type. Use the parameter γ to automatically adjust the ratio of the loss function. This not only considers the imbalance of sample categories in traffic identification but also solves the difference of recognization cost in different samples. **Figure 5B** shows the comparison with the cross-entropy loss function and the weighted cross-entropy loss function.

Different traffic forms and network layer divisions contain different data content. We have studied the impact of different traffic forms and network layers on the classifier, as shown in **Figure 6A**. A total of eight types of samples were obtained for the two traffic forms and the four network layers. On the data set ISCX (Draper-Gil et al., 2016), we compared the recognition effects of the eight forms. As can be seen from **Figure 6A**, Session + All Layer performs best. The accuracy rate is 0.942, the recall rate is 0.973, and the F1-score is 0.955, because it contains more traffic characteristics.

In application classification, we selected seven applications, a total of 52,155 encrypted network traffic samples, with an average of 7,000 sample data for each application. The results of the capsule convolutional neural network are shown in **Figure 6B** and the results of Long short-term memory network are shown in **Figure 7**. The results show that these two methods have excellent performance for traffic classification.

Many scholars have performed classification method evaluation on the data set ISCX (Moore and Zuev, 2005; Alberto et al., 2006; Huang et al., 2014). Here we compare the Inception-CapsNet classifier with them. As shown in **Table 6**, it can be seen that the accuracy and recall rate of the other four types of methods have been improved. For the decision tree C4.5

algorithm, the accuracy rate has increased by 4.3%, and the recall rate has increased by 7.0%.

The CNN-LSTM joint model experimental results are shown in **Figure 8**. For Hangout and Bittorrent, the convolution method has a high recognition accuracy of 96%. For Facebook, Skype has a low recognition accuracy rate of only 87%, while for ATM, Hangout, and Bittorrent, The recall rate is high, reaching 98.

The results show that the recognition accuracy of the time series method for Facebook, Skype, Hangout is as high as 97%, and the recognition accuracy for AIM and uTorrent is low, only 91%. Facebook's recognition recall rate is even higher, reaching 99%.

TABLE 5 | Comparison of classification performance with other traffic classification algorithms on Datasets ISCX.

Algorithm	Accuracy	Recall
CNN-LSTM	0.981	0.995
C4.5	0.901	0.903
SVM	0.943	0.929
1dCNN	0.933	0.951
2dCNN	0.936	0.955
Apriori	0.931	0.911
Naïve Bayes	0.911	0.927
Hmm-crf	0.955	0.967

TABLE 6 | Comparison of classification performance with other traffic classification algorithms on datasets UNIBS.

Algorithm	Accuracy	Recall
LSTM	0.945	0.973
SVM	0.959	0.953
Multi-classifier	0.924	0.971
Random forest	0.936	0.992
Xgboost	0.931	0.961
CNN-LSTM	0.986	0.987

For the combined model, the recognition accuracy has been significantly improved. Among them, Facebook, Hangout, and Bittorrent have a high recognition accuracy of 98.5%; for ATM, FTPS, Skype, and uTorrent, the recognition accuracy is high. Reached 96%; and for the recall rate, the recall rate of encrypted traffic of AIM, Facebook, Hangout, Bittorrent, and Skype reached 99%. For the traffic FTPS, the recall rate of the two encrypted traffic of uTorrent reached 96%.

From the above results, it can be seen that different network structures extract features of different dimensions, and have different preferences for the quasi-curvature and recall rate of encrypted traffic recognition of the same application. For some traffic convolutional neural networks, the recognition accuracy rate is higher. But the recall rate may be relatively low, such as FTPS. For the encrypted traffic of this kind of application, although the recognition quasi-curvature of the time series neural network is moderate, the recall rate is high. In the experimental results of, this conclusion has also been proved, the accuracy rate of its encrypted traffic has reached more than 96%, the recall rate is 99%, only two kinds are lower, and it is also more than 96%. The effect is compared with the previous neural network. The organization has very distinctive features. This proves that extracting features from different dimensions and different feature spaces helps to capture more recognizable features and enhance the model effect.

Table 5 shows the comparison results on ISCX, where the number of features used by the machine learning method is different, reflecting the complexity of the manual design, such as SVM using 21 manual design Features. The decision tree uses 18 artificially designed features, but the two methods have only ten common features, and it is difficult to generalize to other data sets. **Table 6** shows the comparison results on the UNIBS data (Gringoli et al., 2009). In this data set, the results obtained by the relevant literature are given by the SVM model, but the accuracy of the combined model exceeds the model, and the recall rate is far higher. For SVM, it is 0.05 less than the random forest model with the best recall rate, ranking second, and the overall effect is

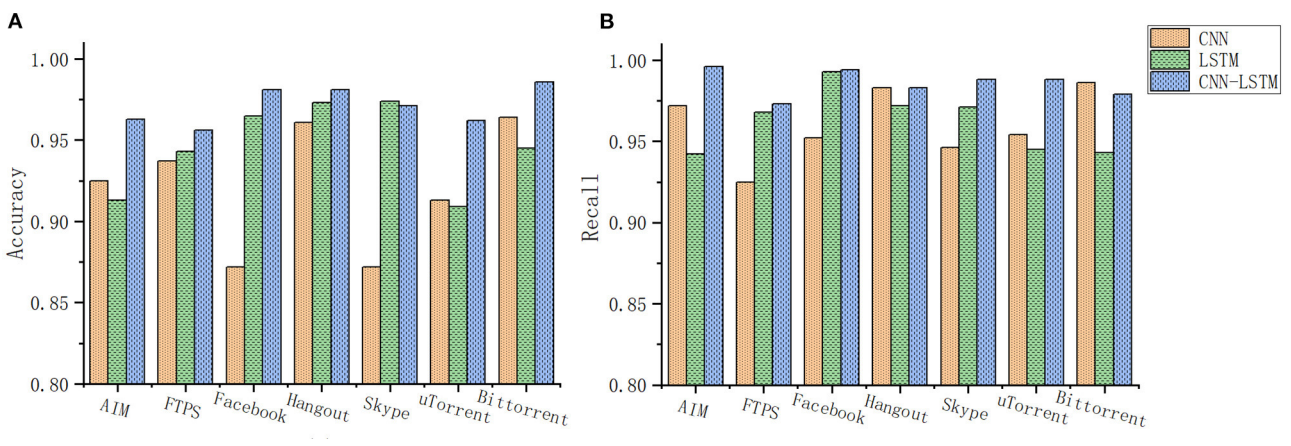

FIGURE 8 | (A) CNN-LSTM joint network application classification accuracy rate. **(B)** CNN-LSTM joint network application classification recall rate.

very good. The results show that the accuracy of the CNN-LSTM network is 8% higher than other methods.

CONCLUSION

In this literature, we propose a capsule convolutional neural network joint the long short-term memory network traffic classification framework. For the problem of imbalance of sample categories in flow recognition, an objective function related to weight and sample recognition accuracy is designed to reduce the classification impact caused by sample imbalance. Besides, the inception structure is added to allow the model to learn diverse features, and the capsulenet structure is added to allow the model to learn the correlation of high and low dimensional features. This model can automatically identify a variety of encrypted traffic and seek the global optimal classification result. The experimental results show that this method can effectively classify the encrypted traffic and is better than previous research work. At the same time, our work proves that the optimized neural network structure can achieve better recognition results.

As future work, We believe that we should try to pay attention to the characteristics of network traffic of different behaviors, so as to more comprehensively describe the communication process between users and robots.

DATA AVAILABILITY STATEMENT

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding author/s.

AUTHOR CONTRIBUTIONS

MG contributed the idea of the paper, verified the idea through experiments, and wrote the paper at the same time. XY guided the idea of the paper and LL assisted the experiment. All authors contributed to the article and approved the submitted version.

FUNDING

This work was supported by National Key R&D Program of China (2016QY05X1000) and National Natural Science Foundation of China (201561402137).

REFERENCES

Alberto, D., Antonio, P., and Pierluigi, S. R. (2006). "An HMM approach to internet traffic modeling," in *Proceeding of IEEE GLOBECOM* (San Francisco, CA). doi: 10.1109/GLOCOM.2006.453

Alizadeh, H., Khoshrou, A., and Zuquete, A. (2015). "Traffic classification and verification using unsupervised learning of Gaussian Mixture Models," in *2015 IEEE International Workshop on Measurements & Networking* (Coimbra), 1–6. doi: 10.1109/IWMN.2015.7322980

Deng, F., Pu, S., Chen, X., Shi, Y., Yuan, T., and Pu, S. (2018). Hyperspectral image classification with capsule network using limited training samples. *Sensors*. 18:3153. doi: 10.3390/s18093153

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

Ebrahimi, M. A., Khoshtaghaza, M. H., Minaei, S., and Jamshidi, B. (2017). Vision-based pest detection based on SVM classification method. *Comput. Electron. Agric*. 137, 52–58. doi: 10.1016/j.compag.2017.03.016

Este, A., Gringoli, F., and Salgarelli, L. (2009). Support vector machines for TCP traffic classification. *Comput. Netw*. 53, 2476–2490. doi: 10.1016/j.comnet.2009.05.003

Fan, Z., and Liu, R. (2017). "Investigation of machine learning based network traffic classification," in *2017 International Symposium on Wireless Communication Systems* (Bologna), 1–6. doi: 10.1109/ISWCS.2017.8108090

Finsterbusch, M., Richter, C., Rocha, E., Muller, J. A., and Hanssgen, K. (2013). A survey of payload-based traffic classification approaches. *IEEE Commun. Surv. Tutor*. 16, 1135–1156. doi: 10.1109/SURV.2013.100613.00161

Fu, R., Zhang, Z., and Li, L. (2016). "Using LSTM and GRU neural network methods for traffic flow prediction," in *2016 31st Youth Academic Annual Conference of Chinese Association of Automation* (Wuhan), 324–328. doi: 10.1109/YAC.2016.7804912

Gleeson, B., MacLean, K., Haddadi, A., Croft, E., and Alcazar, J. (2013). "Gestures for industry intuitive human-robot communication from human observation," in *2013 8th ACM/IEEE International Conference on Human-Robot Interaction* (Tokyo), 349–356. doi: 10.1109/HRI.2013.6483609

Gringoli, F., Salgarelli, L., Dusi, M., Cascarano, N., Risso, F., and Claffy, K. C. (2009). Gt: picking up the truth from the ground for internet traffic. *ACM SIGCOMM Comput. Commun. Rev*. 39, 12–18. doi: 10.1145/1629607.1629610

Groleat, T., Arzel, M., and Vaton, S. (2012). "Hardware acceleration of SVM-based traffic classification on FPGA," in *2012 8th International Wireless Communications and Mobile Computing Conference* (Limassol). doi: 10.1109/IWCMC.2012.6314245

Gu, Z., Hu, W., Zhang, C., Lu, H., Yin, L., and Wang, L. (2020). Gradient shielding: towards understanding vulnerability of deep neural networks. *IEEE Trans. Netw. Sci. Eng*. doi: 10.1109/TNSE.2020.2996738

Hjelmvik, E., and John, W. (2009). "Statistical protocol identification with spid: preliminary results," in *Swedish National Computer Networking Workshop* (Uppsala), 4–5.

Huang, C., Gao, D., Hu, S., Geng, H., and Peng, Y. (2014). Association rule analysis of vessel traffic accidents based on Apriori algorithm. *J. Shanghai Maritime Univ*. 35, 18–22.

Kanda, T., Shiomi, M., Miyashita, Z., Ishiguro, H., and Hagita, N. (2010). A communication robot in a shopping mall. *IEEE Trans. Robot*. 26, 897–913. doi: 10.1109/TRO.2010.2062550

Kornycky, J., Abdul-Hameed, O., Kondoz, A., and Barber, B. C. (2016). Radio frequency traffic classification over WLAN. *IEEE/ACM Trans. Netw*. 25, 56–68. doi: 10.1109/TNET.2016.2562259

Li, F., Seddigh, N., Nandy, B., and Matute, D. (2000). "An empirical study of today's Internet traffic for differentiated services IP QoS," in *Proceedings ISCC 2000. Fifth IEEE Symposium on Computers and Communications* (Antibes-Juan Les Pins), 207–213. doi: 10.1109/ISCC.2000.860640

Li, Z., Yuan, R., and Guan, X. (2007). "Accurate classification of the internet traffic based on the SVM method," in *2007 IEEE International Conference on Communications* (Glasgow), 1373–1378. doi: 10.1109/ICC.2007.231

Liu, Y., Li, W., and Li, Y. (2007). "Network traffic classification using k-means clustering," in *Second International Multi-Symposiums on Computer and Computational Sciences* (Iowa City, IA), 360–365. doi: 10.1109/IMSCCS.2007.52

Madhukar, A., and Williamson, C. (2006). "A longitudinal study of P2P traffic classification," in *14th IEEE International Symposium on Modeling, Analysis, and Simulation* (Monterey, CA), 179–188. doi: 10.1109/MASCOTS.2006.6

Mavridis, N. (2015). A review of verbal and non-verbal human–robot interactive communication. *Robot. Auton. Syst*. 63, 22–35. doi: 10.1016/j.robot.2014.09.031

Moore, A. W., and Papagiannaki, K. (2005). "Toward the accurate identification of network applications," in *International Workshop on Passive and Active Network Measurement* (Berlin; Heidelberg), 41–54. doi: 10.1007/978-3-540-31966-5_4

Moore, A. W., and Zuev, D. (2005). "Internet traffic classification using bayesian analysis techniques," in *Proceedings of the 2005 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems* (Banff, AB), 50–60. doi: 10.1145/1064212.1064220

Pacheco, F., Exposito, E., Gineste, M., Baudoin, C., and Aguilar, J. (2018). Towards the deployment of machine learning solutions in network traffic classification: a systematic survey. *IEEE Commun. Surv. Tutor.* 21, 1988–2014. doi: 10.1109/COMST.2018.2883147

Rezaei, S., and Liu, X. (2019). Deep learning for encrypted traffic classification: an overview. *IEEE Commun. Mag.* 57, 76–81. doi: 10.1109/MCOM.2019.1800819

Shafiq, M., Tian, Z., Bashir, A. K., Du, X., and Guizani, M. (2020). Corrauc: a malicious bot-iot traffic detection method in iot network using machine learning techniques. *IEEE Internet Things J.* 8, 3242–3254. doi: 10.1109/JIOT.2020.3002255

Shafiq, M., Yu, X., Bashir, A. K., Chaudhry, H. N., and Wang, D. (2018). A machine learning approach for feature selection traffic classification using security analysis. *J. Supercomput.* 74, 4867–4892. doi: 10.1007/s11227-018-2263-3

Si, C., Chen, W., Wang, W., Wang, L., and Tan, T. (2019). "An attention enhanced graph convolutional LSTM network for skeleton-based action recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Long Beach, CA), 1227–1236. doi: 10.1109/CVPR.2019.00132

Siahaan, H., Mawengkang, H., Efendi, S., Wanto, A., and Windarto, A. P. (2019). Application of classification method C4. 5 on selection of exemplary teachers. *J. Phys. Conf. Ser.* 1235:1. doi: 10.1088/1742-6596/1235/1/012005

Su, S., Tian, Z., Liang, S., Li, S., Du, S., and Guizani, N. (2020). A reputation management scheme for efficient malicious vehicle identification over 5G networks. *IEEE Wireless Commun.* 27, 46–52. doi: 10.1109/MWC.001.1900456

Sun, G., Chen, T., Su, Y., and Li, C. (2018). Internet traffic classification based on incremental support vector machines. *Mobile Netw. Appl.* 23, 789–796. doi: 10.1007/s11036-018-0999-x

Tao, W., Li, C., Song, R., Cheng, J., Liu, Y., Wan, F., et al. (2020). EEG-based emotion recognition via channel-wise attention and self attention. *IEEE Trans. Affect. Comput.* doi: 10.1109/TAFFC.2020.3025777

Tian, Z., Luo, C., Qiu, J., Du, X., and Guizani, M. (2020). A distributed deep learning system for web attack detection on edge devices. *IEEE Trans. Ind. Inform.* 16, 1963–1971. doi: 10.1109/TII.2019.2938778

Vinayakumar, R., Soman, K. P., and Poornachandran, P. (2017). "Applying convolutional neural network for network intrusion detection," in *2017 International Conference on Advances in Computing, Communications and Informatics* (Udupi), 1222–1228. doi: 10.1109/ICACCI.2017.8126009

Wang, P., Chen, X., Ye, F., and Sun, Z. (2019). A survey of techniques for mobile service encrypted traffic classification using deep learning. *IEEE Access* 7, 54024–54033. doi: 10.1109/ACCESS.2019.2912896

Wang, W., Zhu, M., Wang, J., Zeng, X., and Yang, Z. (2017). "End-to-end encrypted traffic classification with one-dimensional convolution neural networks," in *2017 IEEE International Conference on Intelligence and Security Informatics* (Beijing), 43–48. doi: 10.1109/ISI.2017.8004872

Wu, D., Chen, X., Chen, C., Zhang, J., Xiang, Y., and Zhou, W. (2015). "On addressing the imbalance problem: a correlated KNN approach for network traffic classification," in *International Conference on Network and System Security* (New York, NY), 138–151. doi: 10.1007/978-3-319-11698-3_11

Xiang, C., Zhang, L., Tang, Y., Zou, W., and Xu, C. (2018). MS-CapsNet: a novel multi-scale capsule network. *IEEE Signal Process. Lett.* 25, 1850–1854. doi: 10.1109/LSP.2018.2873892

Yin, C., Li, S., and Li, Q. (2012). Network traffic classification via HMM under the guidance of syntactic structure. *Comput. Netw.* 56, 1814–1825. doi: 10.1016/j.comnet.2012.01.021

Yuan, R., Li, Z., Guan, X., and Xu, L. (2010). An SVM-based machine learning method for accurate internet traffic classification. *Inform. Syst. Front.* 12, 149–156. doi: 10.1007/s10796-008-9131-2

Zhu, Z., Peng, G., Chen, Y., and Gao, H. (2019). A convolutional neural network based on a capsule network with strong generalization for bearing fault diagnosis. *Neurocomputing* 323, 62–75. doi: 10.1016/j.neucom.2018.09.050

Zuev, D., and Moore, A. W. (2005). "Traffic classification using a statistical approach," in *International Workshop on Passive and Active Network Measurement* (New York, NY), 321–324. doi: 10.1007/978-3-540-31966-5_25

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.