symmetry

MDPI

Article

ICLSTM: Encrypted Traffic Service Identification Based on Inception-LSTM Neural Network

Bei Lu, Nurbol Luktarhan *, Chao Ding and Wenhui Zhang

College of Information Science and Engineering, Xinjiang University, Urumqi 830000, China; lubei@stu.xju.edu.cn (B.L.); dingchao@stu.xju.edu.cn (C.D.); zwh@stu.xju.edu.cn (W.Z.)
* Correspondence: nurbol@xju.edu.cn

Abstract: The wide application of encryption technology has made traffic classification gradually become a major challenge in the field of network security. Traditional methods such as machine learning, which rely heavily on feature engineering and others, can no longer fully meet the needs of encrypted traffic classification. Therefore, we propose an Inception-LSTM(ICLSTM) traffic classification method in this paper to achieve encrypted traffic service identification. This method converts traffic data into common gray images, and then uses the constructed ICLSTM neural network to extract key features and perform effective traffic classification. To alleviate the problem of category imbalance, different weight parameters are set for each category separately in the training phase to make it more symmetrical for different categories of encrypted traffic, and the identification effect is more balanced and reasonable. The method is validated on the public ISCX 2016 dataset, and the results of five classification experiments show that the accuracy of the method exceeds 98% for both regular encrypted traffic service identification and VPN encrypted traffic service identification. At the same time, this deep learning-based classification method also greatly simplifies the difficulty of traffic feature extraction work.

Keywords: encrypted traffic service identification; neural network; inception; LSTM

Citation: Lu, B.; Luktarhan, N.; Ding, C; Zhang,W. ICLSTM: Encrypted Traffic Service Identification Based on Inception-LSTM Neural Network. *Symmetry* **2021**, *13*, 1080. https://doi.org/10.3390/sym13061080

Academic Editor: José Carlos R. Alcantud

Received: 16 May 2021
Accepted: 8 June 2021
Published: 17 June 2021

1. Introduction

In recent years, traffic encryption has been widely used on the Internet due to advanced encryption technology. A large number of services and applications use encryption algorithms as the primary method for protecting information. Gartner estimated that more than 80% of enterprise network traffic was encrypted by 2019, and 94% of Google network traffic was encrypted by May 2019. This encryption technology not only protects the freedom, privacy, and anonymity of network users, but also enables them to circumvent firewall detection and surveillance systems [1]. However, encryption has also been exploited by unscrupulous individuals to gain illegal benefits. For example, in 2020, more than 70% of malware campaigns used some kind of encryption to hide malware delivery, commands and data leakage. Therefore, the identification and classification of encrypted traffic have received a lot of attention from academia and industry [2]. The development of encryption technology makes the data packets change from plaintext to ciphertext after passing through the encryption algorithm (for example, symmetric cryptography or asymmetric cryptographic algorithm, etc.). A lot of information becomes no longer visible, which also brings great difficulties to the encrypted traffic classification. Practical scenarios often require the identification of specific protocols or application types. For encrypted application traffic, this also makes the traffic classification difficult because there are more application types and there is little difference between different types.

With the good results achieved by deep learning in image classification, speech recognition, natural language, and so forth, deep learning methods are also gradually applied to the field of network security because of their advantages of automatic feature extraction. To

learn spatial features, researchers have used convolutional neural networks (CNNs), including one-dimensional convolutional neural networks (1dCNN) and two-dimensional convolutional neural networks (2dCNN) [3] to obtain better classification results. Javaid et al. [4] proposed a network intrusion detection scheme using sparse autoencoders (SAEs). Reference [5] used Long Short Term Memory (LSTM) to extract time series features between traffic groupings. However, the commonly used CNN, LSTM, and other methods improve the classification results while the problem of network computational complexity cannot be ignored.

Deep learning has emerged as a highly desirable approach for traffic classification as an end-to-end method. It is able to learn the nonlinear relationship between the original input and the corresponding output without decomposing the problem into subproblems of feature selection and classification [6,7]. One of the advantages of deep learning is higher learning capability than the traditional ML methods [8]. Another advantage is that it can automatically select features by training, does not need domain experts to select features, and does not need to rely on complex feature engineering. In addition, due to the different popularity of various applications, the problem of class imbalance in traffic samples often arises when constructing traffic datasets [9]. Based on the above analysis, we propose a new deep learning model for encrypted traffic service identification—a neural network structure based on the Inception module [10] in parallel and LSTM module. The model uses a convolutional neural network introducing the Inception module for packet local space feature extraction and an LSTM module for packet time series feature extraction. The accuracy and effectiveness of the model are finally evaluated by the public ISCX 2016 traffic dataset, and weights are assigned to different categories for the imbalanced dataset during the experiment. The main contributions of this paper are summarized as follows:

- A new encrypted traffic identification method-ICLSTM is proposed, which can automatically extract traffic features using neural networks, and no complex feature engineering is required.
- A model architecture containing two neural networks is proposed for feature extraction of encrypted traffic. One-dimensional convolutional neural network embedded the Inception module is used to extract local features of the traffic, and the LSTM model is used to extract the temporal features of the packets within a session. Then the extracted features are fused, which extends the feature information and enhances the characterization of packet features. Experiments show that our method can achieve better results in encrypted traffic service identification.
- A processing scheme for unbalanced data sets is proposed to enhance the symmetry of the data by adopting the method of assigning weights to different categories to effectively alleviate the data imbalance problem.

Section 2 presents the work related to the classification of encrypted traffic. Section 3 presents the proposed method in this paper. Section 4 focuses on the experimental results and evaluation. Section 5 gives the conclusion and future work.

2. Related Work

The rapid growth of encrypted network traffic makes traffic classification more difficult. The existing solutions to traffic classification are mainly divided into three categories: port-based, deep packet inspection (short for DPI) [11,12] based and machine learning approaches [13,14]. Classical port-based approaches perform well for applications with specific port numbers, but do not classify all protocols due to the dynamic port assignment used by many applications in the current state [15]. DPI can analyze the entire packet data to identify its network protocol and application, but this inspection is difficult to properly analyze encrypted traffic because the packet payload is encrypted into a pseudo random format, containing fewer common features used for traffic classification. Current research on encrypted traffic classification mainly focus on automatic classification of applications using machine learning algorithms, classification of well-known applications such as HTTP, SMTP, FTP, Skype, and so forth. Among them, flow features (duration,

bytes per second) and packet features (packet size, inter-packet duration) are the most commonly used features. Michael J. De Lucia et al. used support vector machine (SVM) malicious communication detection mechanism and achieved good results and low false positive rate (FPR) [16]. Gil et al. used time-dependent features, such as flow duration, bytes of traffic per second, forward inter arrival time and backward inter arrival time, and so forth, to describe network traffic using K nearest neighbors (KNN) and C4.5 decision tree algorithm [17]. They used the C4.5 algorithm to describe six major categories of traffic such as web browsing, email, chat, and VoIP, achieving a recall rate of about 92%. On the same dataset over VPN tunnels, they achieved a recall of about 88% using the C4.5 algorithm. The literature [18] proposed a multi-level P2P traffic classification technique using C4.5 decision trees and statistical features of flows for P2P classification, which was also applicable to encrypted traffic. Similarly, there is literature [19,20] that used machine learning (KNN, SVM) for fine-grained classification of encrypted traffic classification. However, the machine learning-based methods relies heavily on effective feature extraction and selection, which is a waste resource for traffic feature extraction. Moreover, traditional threat detection to decrypt encrypted traffic is not feasible as it not only consumes a lot of computational resources but also compromises privacy and data integrity [21].

The deep learning approach can effectively solve these problems because deep learning methods can read information directly from the raw data and can achieve a high accuracy rate. Not much work has been done to classify network traffic using deep learning methods. Wang [22] proposed an end-to-end encrypted traffic classification method based on one-dimensional convolutional neural networks. The method integrated feature extraction, feature selection, and classifier into a unified end-to-end framework designed to automatically learn the nonlinear relationship between the original input and the desired output, and obtains a high accuracy rate. The author illustrated the effectiveness of convolutional neural networks for encrypted traffic classification, but did not tune the model as well as did not consider whether the dataset is balanced. Reference [23] proposed a deep packet framework embedded with stacked self-encoders and convolutional neural networks that can handle both traffic features and application recognition and can identify encrypted traffic, which can achieve 98% recall in application recognition and 94% in traffic classification. Following this line of research, Zhuang Zou [24] et al. proposed a new deep neural network combining convolutional and recurrent neural networks to improve the accuracy of classification and also illustrated the effectiveness of hidden temporal features among traffic packets for classification of encrypted traffic. Similarly, there are, for example, Stack Auto Encoder (SAE), CNN, LSTM networks [25–27], and so forth. Deep networks require a large number of implicit layers and a large number of neurons to improve their accuracy. Inspired by this, the deep neural network model combining Inception network and LSTM network in our work can learn data features directly from the raw traffic and classify the application traffic for the purpose of encrypted traffic service identification. For the class imbalance problem, we consider assigning different weights to different classes separately to reduce the impact of class imbalance on the classification effect.

3. Methodology

In this work, we design an Inception-LSTM architecture, which consists of two deep learning methods, namely Inception module and LSTM module, for application service identification. Before training the model, we need to process the prepared traffic data into the format needed by the model so that it can be correctly input the model. For this purpose, we preprocess the dataset and use the preprocessed data to train a neural network model to predict the category to which the packets belong. The details of the preprocessing stage, the neural network architecture, and the method design are described in detail below. Figure 1 shows the architecture of the ICLSTM.

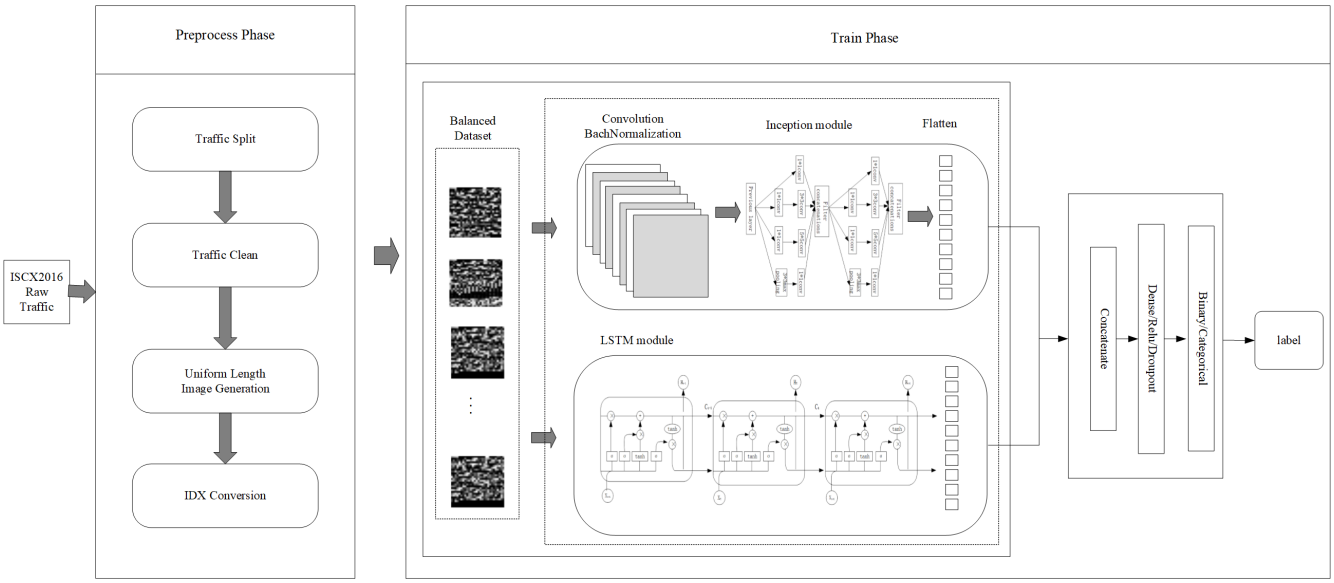


Figure 1. System architecture diagram.

3.1. Dataset

The data source used in this paper is "ISCX 2016" [17], which consists of captured traffic from different applications. In this dataset, the captured packets are classified into different categories based on the application that generated the packet (e.g., Skype, Hangouts, etc.) and the specific activity that the application engaged in during the captured session (e.g., voice call, chat, file transfer, or video call, etc.). The published ISCX 2016 dataset includes seven types of regular encrypted traffic and seven types of VPN tunneling traffic. In this paper, a total of 5.17 GB of raw traffic of six types of regular encrypted traffic and six types of VPN tunnel transmission encrypted traffic are selected as the sample data, and the sample datasets are in the PCAP file format. Table 1 shows the details of the sample data set after processing in this paper.

Table 1. Dataset description.

Encryption Type	Traffic Service Type	Applications	Sample Size
Regular encrypted traffic	Chat	ICQ/AIM/Skype/Facebook/Hangous	9285
	Email	Email/Gmail	6983
	File Transfer	File Transfer	51,235
	P2P	Torrent	940
	Streaming	Netflix/Spotify/Netflix/Vimeo/Youtube	1680
	VoIP	Hangouts	71,093
VPN encrypted traffic	VPN-Chat	Aim/Facebook/Hangous/icq/skype	3621
	VPN-Email	Email	268
	VPN-File Transfer	Ftps/Sftp/Skype	915
	VPN-P2P	Bittorrent	429
	VPN-Streaming	Youtube/Vimeo/Netflix/Spotify/Facebook	590
	VPN-VoIP	Facebook/Hangous/Skype/Voipbuster	6330

3.2. Preprocessing

3.2.1. Traffic Representation Options

There are five types of network traffic segmentation—TCP connection, flow, session, service and host. Currently, the segmentation methods based on flow and session are widely used in research. Flow are all packets with the same five-tuple (source IP address, source port, destination IP, destination port, and transport layer protocol). Session are all packets that consist of bidirectional flow, that is, the source and destination IP/port of the five-tuple are interchangeable.

Raw flow: the set P of all packets, each packet has five-tuple information.
Flow: The set P is divided into multiple subsets according to the five-tuple information, and the packets in each subset are arranged in chronological order within a certain time window, making it a flow f.
Session: The difference with flow is that the source and destination IP/port of its five-tuple are interchangeable, so it is also called bidirectional flow. The current research is more utilized also based on session flow, so we also chose the session flow.

In addition, analyzed from the protocol layer perspective, although the traffic feature is mainly reflected in the application layer, it is also reflected in different forms in each protocol layer. Therefore, to avoid missing key features, all protocol layer session flow data are selected in this paper for data pre-processing.

3.2.2. Data Preprocessing

To reduce the redundant information in the raw traffic and adapt it to the input form suitable for deep learning model, we utilized the USTC-TK2016 toolset [22] to process the raw traffic through the following four steps: traffic split, traffic clean, uniform the input size, generate gray images, and convert the IDX files.

(1) Pcap-session segmentation: continuous raw traffic is divided into multiple discrete traffic units according to a certain granularity [28].
(2) Traffic clean: packet files without application layer generate bin files without actual content. The packet files with the same content for sessions or flows generate duplicate files. So it is necessary to clean up the chopped traffic data and only retain the needed traffic data.
(3) Uniform the input size and generate gray images: using deep learning Neural network to train data requires a fixed amount of input, and we unifies the session segments in the above steps to 784 bytes in size. On the one hand, there are relevant papers proving that 784 bytes are effective for classification, and on the other hand, 784 bytes are more lightweight for some literature dealing with 1500 bytes. If the segment size is larger than 784 bytes, it is trimmed to 784 bytes. If the segment size is smaller than 784 bytes, add 0×00 at the end to supplement to 784 bytes and convert it to a gray images of size 28×28.
(4) Conversion to IDX: IDX format is a common file format in the field of deep learning. We converted the generated traffic gray images to the IDX file format which commonly used by neural network models.

We randomly select four preprocessed generated images from each service category in both regular encrypted traffic and VPN encrypted traffic. The results of the data pre-processing are analyzed using visualization techniques, as shown in Figure 2. Through the visualized image, it is obvious that there is a great distinction between different categories of encrypted traffic, and the data of the same category are similar. Therefore, we believe that these unstructured abstract data can be explored by neural networks to classify the encrypted traffic using richer features.

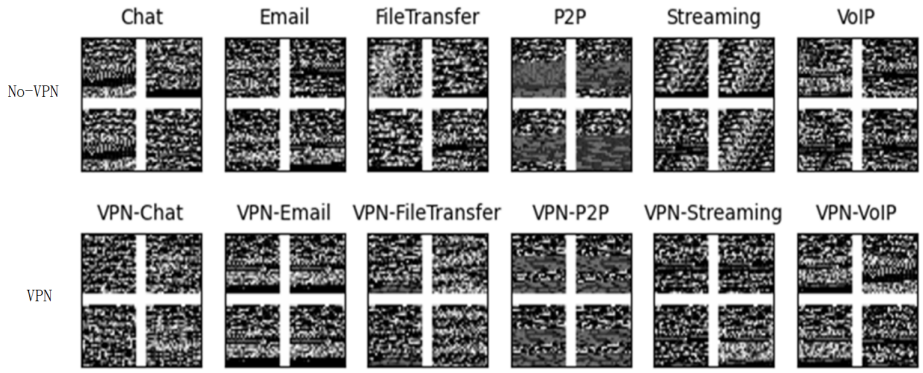


Figure 2. Data visualization.

3.3. Class Imbalance

From Table 1, we can clearly see that there is a significant difference in the number of traffic in each class. In the classification task, if there is a significant difference in the number of samples in each class, the model obtained is generally better at predicting classes with a larger number of samples and worse at predicting classes with fewer samples, so we need to achieve a more balanced training set. To solve this problem, we automatically assign a weight to the different categories of data in the training set [29], and the category data with a large percentage are assigned to smaller weights, conversely, larger weights. This method makes the loss function pay more attention to the data with insufficient sample size. In this way, the problem of data imbalance is alleviated and the generalization ability of the model is enhanced. This is calculated in Equations (1) and (2).

$$score = log(\frac{\mu \times total}{label\_nums_{key}}). \tag{1}$$

$$class\_weight_{key} = \begin{cases} score & score > 1 \\ 1 & score \leq 1 \end{cases}, \tag{2}$$

where μ = 0.15; the *log* can smooth the weights of imbalance classes; *total* represents the number of training samples; $label\_nums_{key}$ represents the number of training samples labeled as *key*, this task is only used for the model training phase.

3.4. Background on Neural Networks

Neural networks (NNs) are networks consisting of a large number of interconnected artificial neurons. These networks are usually composed of a large number of building blocks called neurons, which also represent a specific output function, called the activation function. The neurons are connected to each other by a number of linkages. Each connection between two neurons represents a weighted value, called a weight, for the signal passing through that connection. It is equivalent to the memory of an artificial neural network. During training, a large number of data samples are used to train the neural network to achieve the desired output of the neural network. Deep learning framework can be considered as a special kind of neural network with many hidden layers. Nowadays, with the rapid growth of computing power and the availability of graphics processing units (GPUs), deep neural network training has become more feasible [23]. Therefore, the current deep learning framework has attracted the attention of researchers in various fields. In the following, we will briefly review the two most important deep neural networks used in our proposed network traffic classification scheme, named the Inception module and the LSTM module.

3.4.1. Inception Module

CNNs are a class of feedforward neural networks that include convolutional computation and have a deep structure with representational learning capability to classify input information in a translation invariant manner according to its hierarchical structure, and are one of the core algorithms for image classification problems [30,31]. Its input layer can handle multidimensional data and data with local relevance, for example, the common one-dimensional convolutional neural network can receive one or two-dimensional arrays. It is good at mining the internal features of hidden images and is well suited for processing sequence data or language-based data. Some convolutional neural networks can also recognize words in images and also can be combined with RNNs to extract character features and sequence labeling from images, which also provides the possibility of successful applications of convolutional neural networks on network traffic [32]. For example, ref. [33] used MLP, SAE and CNN to classify over 200,000 samples of encrypted data from 15 applications. The core of convolutional neural networks are: local perceptual weight sharing and down-sampling. However, for the traditional CNN architecture, the mainstream network structure tends to increase the number of layers and neurons, which also greatly increases the disadvantages of the network: more parameters, easier overfitting; the larger the network, the greater the complexity and difficult to apply; the deeper the network, the easier the gradient disappears. Inspired by the GoogLeNet network [10], we introduce two Inception modules on the basis of convolutional neural network to reduce the parameters while increasing the depth and width of the network. The structure of the Inception module is shown in Figure 3.

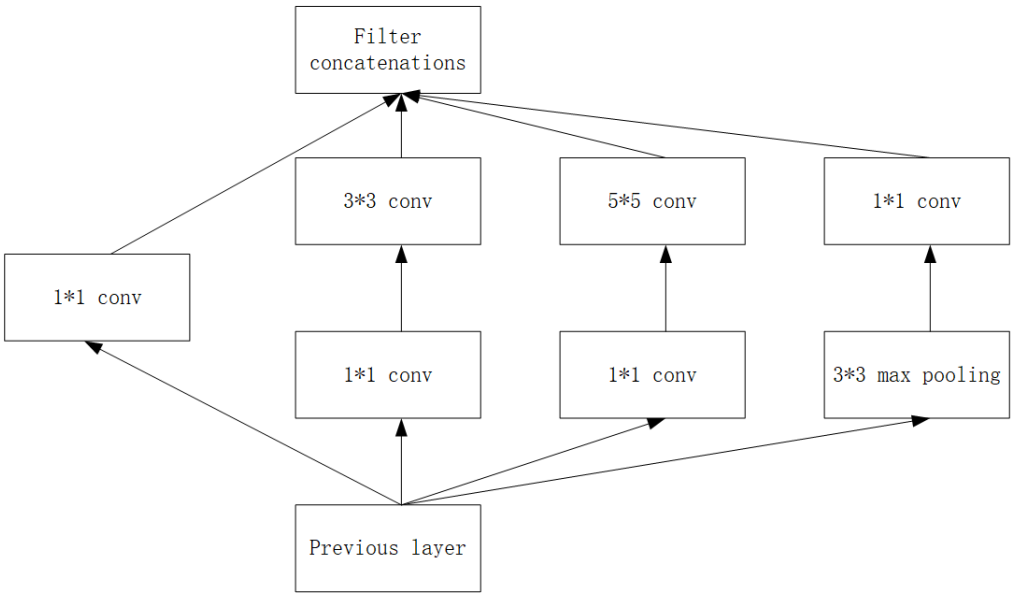


Figure 3. The structure of Inception.

The Inception module is a change to the convolutional layers to increase the network width, enhance the adaptability of the network to the scale, and improve the network performance. In each Inception module, convolutional kernels of different sizes, which can be interpreted as different receptive fields, and then connected to enrich the information in each layer. The Inception module uses convolution kernels with different scales (1*1, 3*3, and 5*5) to extract features from the input. Features with different scales can be extracted, and the output features are not uniform distribution. The features with strong relevance are clustered together, and the relevant non-critical features are then weakened. In addition, the module introduces 1*1 convolution kernel to reduce feature dimensionality, parameters and computational complexity. The output of this method has less "redundant" information, and this feature set is transmitted layer by layer and used as the input of reverse calculation, the rate of convergence is faster. The output part of the module has a

concatenate layer, the purpose of which is to merge four sets of features of different types but the same size to form a new feature map, which is also the local spatial features inside the obtained data packet.

3.4.2. Long Short-Term Memory Module

Nowadays, it is common to treat network traffic as a continuous one-dimensional byte stream with a hierarchical structure, and its whole stream structure (bytes, packets, sessions, communication streams, etc.) is similar to the structure of characters, words, sentences and so forth. In the field of natural language, with sequential correlation. LSTM [34], as a special recurrent neural network, overcomes the problems of gradient disappearance and gradient explosion of traditional RNN, Moreover, it is better at extracting long-term dependence of sequential data, and is more suitable for dealing with network traffic data with temporal characteristics. The LSTM is a chain structure with four internal network layers, the structure is shown in Figure 4.

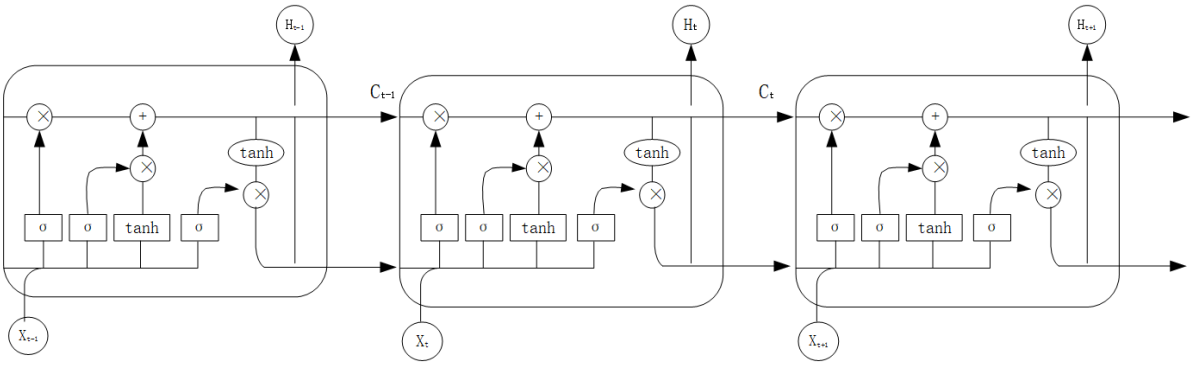


Figure 4. The structure of LSTM.

LSTM is able to remove or add information to cell states through a structure called gate, which is a combination of a sigmoid layer and a dot product operation that selectively decides which information can pass. LSTM contains three gates to control cell states, namely: a forget gate, an input gate and an output gate.

(I) Forget gate. The first step of LSTM is to decide what information needs to be discarded from the cell state, which is done by the sigmoid cell of the oblivion gate to decide what information needs to be removed from the LSTM memory. 0: Completely discarded, 1: Completely retained. Calculated as Equation (3).

$$f_t = \sigma(W_f \bullet [h_{t-1}, x_t] + b_f), \tag{3}$$

where b_f is a constant, called the bias value.

(II) Input gates. The second step of the LSTM is to use the input gates to decide what information to add to the cell state into the LSTM memory. The output is calculated as Equations (4)–(6).

$$i_t = \sigma(W_i \bullet [h_{t-1}, x_t] + b_i). \tag{4}$$

$$\widetilde{C}_t = tanh(W_c \bullet [h_{t-1}, x_t] + b_c). \tag{5}$$

$$C_t = f_t * C_{t-1} + i_t * \widetilde{C}_t, \tag{6}$$

where i_t indicates whether the value needs to be updated, $\widetilde{C}_t$ denotes a new vector of candidate values, C_t is the cell information, and f_t is the parameter of the forgetting gate with a value between 0 and 1, 0: Completely discarded, 1: Completely retained.

(III) Output gate. After updating the cell states it is necessary to determine which state characteristics of the output cells are based on the input h_{t-1} and x_t. The calculation is as Equations (7) and (8).

$$O_t = \sigma(W_o \bullet [h_{t-1}, x_t] + b_o). \tag{7}$$

$$h_t = O_t * tanh(C_t), \tag{8}$$

where O_t is the output value and h_t is a value between -1 and 1. This step determines which part of the cell state will be output to the next neural network or the next moment.

The temporal backward propagation algorithm of LSTM starts from the last value of the time series, gradually calculates the gradient of each parameter in reverse, and finally updates the network parameters with the gradient of each moment respectively. Firstly, the partial derivatives corresponding to the output of the memory cell are calculated, and then the partial derivatives of the output gate are calculated. The partial derivatives corresponding to the memory cell state, forget gate and and input gate are calculated separately. Finally, the model connection weights are updated using the gradient descent method.

3.5. Model Architectures

In recent years, both CNN and LSTM have been successfully used in NLP, such as sentiment analysis and text classification [35–37]. We are inspired by these studies to design an Inception-LSTM framework that includes two deep learning methods, namely Inception module and LSTM module for encrypted traffic classification, and compare the performance with the latest model.

The input of the model is the processed encrypted traffic session data and the output is the object labels to be estimated. The model is executed by two parallel neural networks. The proposed Inception component is based on a one-dimensional CNN convolutional layer which introduces two identical Inception modules and a normalization layer [38] to enhance the computational as well as generalization capabilities of the network. The LSTM module extracts the temporal features of the packets within a session. The model receives processed network traffic data, which is considered as essentially hierarchically structured byte flow data.

The Inception architecture consists of two identical Inception modules that are stacked on top of each other, and the structure of each Inception module is shown in Section 3.4.1, Figure 3. Firstly, the Inception architecture reads the 28*28*1 traffic images from the IDX file and uses the Inception module to extract the local features inside the packets as much as possible. These image pixels are normalized from [0, 255] to [0, 1]. Each Inception module is superimposed by multiple convolutions to extract richer features, so the inputs of this model are processed by multiple parallel convolutional branches and then the outputs of these branches are combined into a single tensor. In the structure of the system proposed in this paper, the outputs of the two Inception modules are passed through a flatten layer and the outputs are converted to 1*1240 one-dimensional vectors to be fused with the outputs of the LSTM modules.

Similarly, The LSTM module extract the backward and forward dependencies as the temporal relationships inside the packets. The module first reads a 28*28*1 traffic image, and then connects a fully connected layer. In addition, the dropout technique is used behind each layer to avoid overfitting problems, and the LSTM module can obtain a 1*200 one-dimensional vector. Finally, the local spatial features extracted by the Inception module and the inter-packet timing features extracted by the LSTM module are fused to extend the packet feature information and enhance the characterization ability of the internal features of the packet. We feed the obtained 1*1440 one-dimensional vectors into the two fully connected layers and use the softmax for the traffic classification.

4. Experimental

4.1. Experimental Configuration

This experiment is based on Python version 3.7, the experimental environment uses the win10 64-bit operating system, the processor is an Intel (R) Core (TM) i5-3470 CPU @3.20 GHz, and there is 20G of memory.

The experiment uses five-fold cross-validation, and the pre-processed data is randomly divided into five parts, the four of them are used as training data and one as test data in turn for the experiment. The main parameters of the model and hyperparameters are as follows: batchsize is 256, epoch is 200, and Adam (lr = 0.0005, beta_1 = 0.95, beta_2 = 0.999, epsilon = 10^{-8}) is chosen as the optimizer for parameter learning.

Reference [22] demonstrated that sessions are more suitable as the type of traffic representation used for encrypted traffic classification. Because sessions contain bidirectional flows that contain more interaction information compared to unidirectional flows, the end-to-end method can learn more features from sessions than flows. From the protocol layer perspective, in order to avoid missing protocol layer information, five experiments will be conducted in this paper using traffic samples from all layers of the session. Our experiments are listed in Table 2 below.

Table 2. Experiment content.

Experiment	Description	Classifier
1	Protocol Encapsulation Identification	2
2	Mixed encrypted traffic service identification	6
3	Regular encrypted traffic service identification	6
4	VPN encrypted traffic service identification	6
5	encrypted traffic service identification	12

Experiment 1 involves protocol encapsulation identification, mainly to implement VPN encapsulation encryption or regular encrypted traffic classification. Experiment 2 mixed encryption traffic service identification, without considering the encryption method or encapsulation type, which only consider the service type identification in the case of the same service type, and the same traffic type of traffic into one category, a total of six categories (Regular encrypted traffic and VPN encrypted traffic with the same type of service are grouped together). Experiments 3 and 4 are regular encrypted traffic service identification and VPN encrypted traffic service identification, with six service types respectively. Experiment 5 is the traffic service type identification considering encapsulation method, identification of traffic service types under regular encrypted traffic (six service types) and VPN encrypted traffic (six service types), for a total of twelve categories.

4.2. Evaluation Indicators

In this paper, we use four evaluation metrics: Accuracy, Precision, Recall and F1-score. Accuracy is used to evaluate the overall performance of the classifier and this is calculated in Equation (9). Precision (Equation (10)), Recall (Equation (11)) and F1-score (Equation (12)) are often used to evaluate the performance of a certain class of traffic. *TP* is the number of instances correctly classified as X, *TN* is the number of instances correctly classified as non-X, *FP* is the number of instances incorrectly classified as X and *FN* is the number of instances incorrectly classified as non-X. The indicator is calculated as follows.

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP}. \tag{9}$$

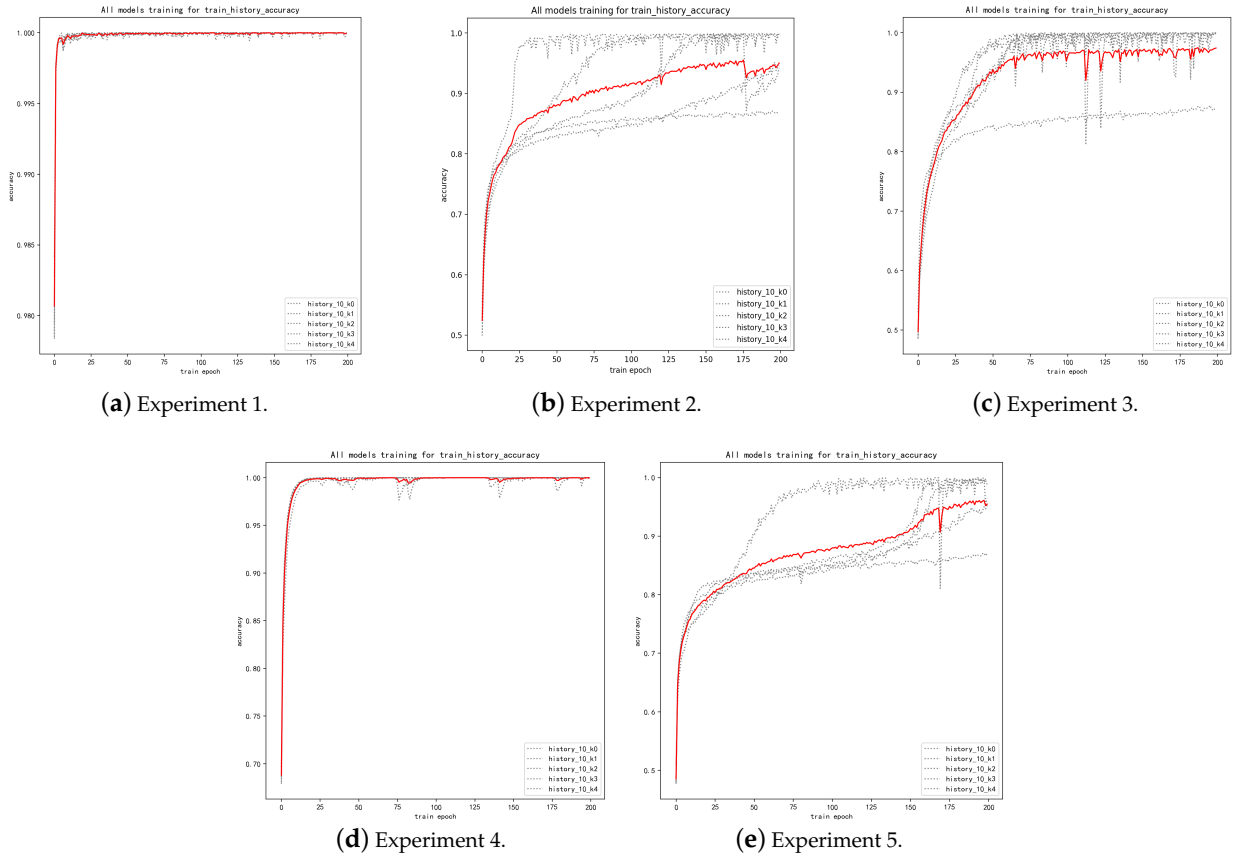
$$Precision = \frac{TP}{TP + FP}. \tag{10}$$

$$Recall = \frac{TP}{TP + \mathrm{FN}}. \tag{11}$$

$$F1\text{-}score = 2 \times \frac{precision \times recall}{precision + recall}. \tag{12}$$

4.3. Evaluation

We use the ISCX 2016 dataset and five-fold cross-validation approach for experiments, and the experimental results are the performance of each category metric after cross-validation on each classification task. Figure 5 shows the variation of the five-fold cross-validation training accuracy for the five experiments.



(**a**) Experiment 1. (**b**) Experiment 2. (**c**) Experiment 3.

(**d**) Experiment 4. (**e**) Experiment 5.

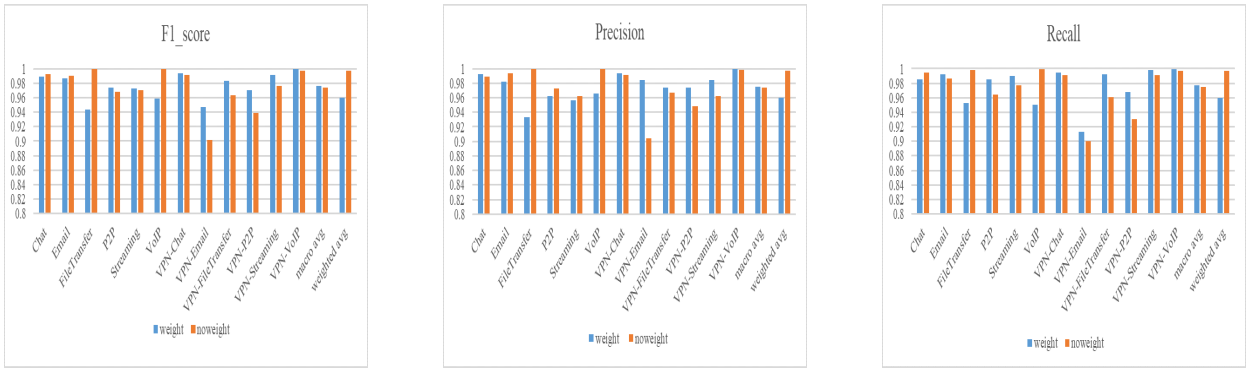
Figure 5. Experiments 1, 2, 3, 4, 5 training effect. (**a**) Protocol Encapsulation Identification. (**b**) Mixed encrypted traffic service identification. (**c**) Regular encrypted traffic service identification. (**d**) VPN encrypted traffic service identification. (**e**) Encrypted traffic service identification.

From Figure 5a–e, we can see that the overall experimental average results are stable, and the training accuracy gradually reaching a peak and leveling off with the increase of training batches. In particular, the best training effect in protocol encapsulation identification (Experiment 1), regular encrypted traffic service identification (Experiment 3) and VPN encrypted traffic service identification (Experiment 4), and their test accuracy can reach nearly 100% and 98.7% and 98.9% respectively. Besides, in the mixed encrypted traffic service identification (Experiment 2) and encrypted traffic service identification (Experiment 5), the overall training effect is satisfactory and meets the expectation except for a few folds in which the effect is not good enough. However, the mixed encrypted traffic service identification has the problem of insufficient training stability, we believe that because the VPN encrypted traffic and regular encrypted traffic are affected by the encryption protocol which are differences, it causes the service identification performance of each application to vary greatly.

4.3.1. Impact of Setting Category Weights

To alleviate the category imbalance, we consider using assigning weights to different categories in the training phase. Taking encrypted traffic service identification as an example, Figure 6 shows the effect of setting weights on F1-score, precision and recall of encrypted traffic service identification (Experiment 5).

Combined with the sample size analysis of the dataset in Table 1, Figure 6a–c show that the F1-score, precision, and recall of the most of small sample traffic types achieved better values in encrypted traffic service identification after assigning weights respectively. For example, in terms of F1-score, some small sample traffic categories (P2P, Streaming, VPN-Email, etc.) have improved, and with a maximum improvement of 4%. In terms of accuracy, traffic categories with smaller samples (VPN-File Transfer, VPN-P2P, VPN-Streaming and VPN-Email, etc.) improved by 1% to 8%. Similarly, recall of the small sample categories, which improved with a maximum of 4%. The same situation exists in other identification tasks. Meanwhile, we also found a phenomenon that the F1-score, precision and recall of the large sample traffic categories (Chat, File Transfer and VoIP) decreased by 0.3% to 5% through experiments. A slight decrease in the identification of large sample traffic types is expected, this is because we focus more on small sample traffic types when assigning weights to categories.



(**a**) (**b**) (**c**)

Figure 6. Effect of setting category weights. (**a**) Comparison of F1-score for encrypted traffic service identification. (**b**) Comparison of Precision for VPN encrypted traffic service identification. (**c**) Comparison of Recall for encrypted traffic service identification.

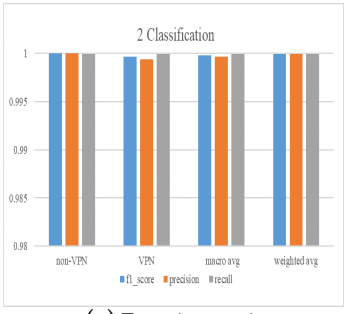
4.3.2. Analysis of Results

Figures 5a and 7a show that the accuracy of the proposed method in this paper can reach close to 1 on average in the protocol encapsulation identification, the precision, recall and F1-score also achieve very good results. Figure 7b–e show the results of the proposed method (ICLSTM) for the traffic service identification on the test set. We can see that the overall model classification effect is more stable. Figure 7 shows that the precision, recall and F1-score reach high values in most experiments of traffic identification. For example, Figure 7c,d show the regular encrypted traffic service identification and VPN encrypted traffic service identification respectively (Experiment 3 & 4). The accuracy can reach 98.7% and 99%, the precision achieved was 98.8% and 97.6%, and the recall can achieve 99% and 97.5% respectively. In addition, in the mixed encrypted traffic service identification experiments, we ignore the protocol encapsulation type and group VPN encrypted traffic and regular encrypted traffic of the same application traffic into the same category for identification, and the results are shown in Figures 5b and 7b and Table 3 (Experiment 2). The accuracy can reach 98.2% on average, and the test precision, recall and F1-score are 98%, 98.4% and 98.2% respectively. Figure 7e shows the performance of the encrypted traffic service identification, which can achieve an average accuracy of 98.1% on the test set, and the precision, recall and F1-score can reach 98%, 98.2% and 98.1% respectively.

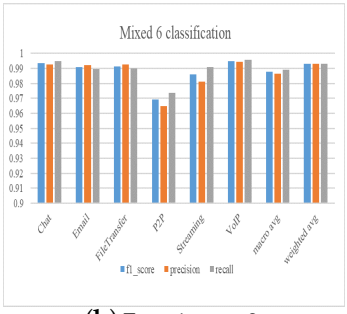
As the Table 3 shows, our model ICLSTM achieved high accuracy, precision, recall and F1-score for each traffic service identification task on the test set. This also shows that our model can achieve autonomous feature extraction as well as successful differentiation of each application service, which illustrates the effectiveness of the method proposed in this paper.

Table 3. Test set performance.

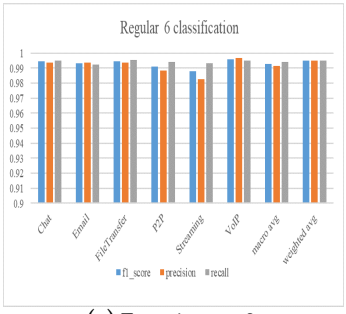
Experiment	Accuracy	Precisionl	Recall	F1-Score
1	100	99.9	100	100
2	98.2	98	98.4	98.2
3	98.7	98.8	99	98.8
4	99	97.6	97.6	97.5
5	98.1	98	98	98.1



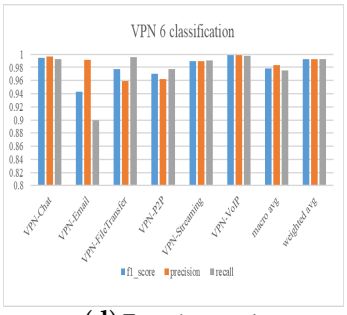
(**a**) Experiments 1.



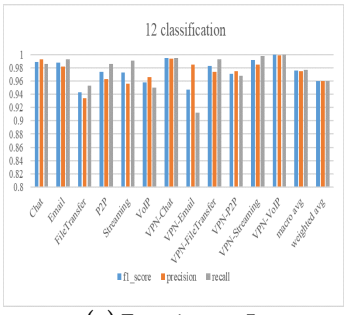
(**b**) Experiments 2.



(**c**) Experiments 3.



(**d**) Experiments 4.



(**e**) Experiments 5.

Figure 7. Experiments 1,2, 3, 4, 5 result. (**a**) Protocol Encapsulation Identification. (**b**) Mixed encrypted traffic service identification. (**c**) Regular encrypted traffic service identification. (**d**) VPN encrypted traffic service identification. (**e**) Encrypted traffic service identification.

4.3.3. Model Comparison

To better analyze the performance of ICLSTM on encrypted traffic service identification, we compare with the results of other methods proposed in recent studies. Such as 1DCNN, Text Convolution [39] and SAE [19], and so forth. The results are shown in the following table. Since F1-score are not shown in some papers, some experimental comparison metrics are selected to compare the performance only in terms of precision and recall. The precision, recall and F1-score in the table are the average values of the corresponding category identification results.

Table 4 shows that the proposed method -ICLSTM can achieve almost the same performance as 1DCNN in protocol encapsulation identification, both the accuracy and recall are substantially improved relative to the method proposed in the literature [17]. Table 5 shows that the precision and recall are significantly improved in both regular encrypted traffic service identification and VPN encrypted traffic service identification, especially the best result in regular encrypted traffic service identification with an improvement of 13.6%.

In addition, in the mixed encrypted traffic service identification, no comparison was made because other papers did not have relevant experiments. In the encrypted traffic service identification (Experiment 5), Tables 3 and 6 show that the average accuracy of ICLSTM can reach 98.1%, the regular encrypted traffic service identification achieves better performance. The precision, recall and F1-score are 96.6%, 97.6% and 97% respectively, with nearly 10% improvement. At the same time, VPN encrypted traffic identification although achieved a better recognition results, compared with the literature [23], the results are not significantly improved, which is also worthy of future research. Table 7 shows the performance of each category in encrypted traffic service identification, and it can be seen from the table that our model has the highest overall precision, recall and F1-score. Last but not least, the most papers do not consider the category imbalance, we address this issue by assigning weights to different categories. Through the above experiments, it can be found that the proposed method in this paper can obtain better identification and classification results on both encrypted traffic service identification, which also verifies the effectiveness of our proposed method.

Table 4. Protocol Encapsulation Identification (Experiment 1).

Work	Model	Non-VPN		VPN	
		Precision	Recall	Precision	Recall
Draper-Gil [17]	C4.5	90.6	88.8	89	92
Wang [22]	1DCNN	100	99.9	99.9	100
This paper	ICLSTM	100	100	99.9	100

Table 5. 6 Classification (Experiment 2 & 3 & 4).

Work	Model	Experiment 2		Experiment 3		Experiment 4	
		Precision	Recall	Precision	Recall	Precision	Recall
Draper-Gil [17]	C4.5	N/A	N/A	89	85.5	84	87.6
Wang [22]	1DCNN	N/A	N/A	85.5	85.8	94.9	97.3
This paper	ICLSTM	98.6	98.9	99.1	99.4	98.3	97.6

Table 6. Encrypted traffic service identification (Experiment 5).

Work	Model	Non-VPN			VPN		
		Precision	Recall	F1-Score	Precision	Recall	F1-Score
Draper-Gil [17]	C4.5	84.3	79.3	81.7	78.2	81.3	79.7
Wang [22]	1DCNN	85.8	85.9	85.8	92	95.2	93.5
Song [39]	Text-CNN	87.6	87.3	87.5	95.2	97.4	96.1
Lotfollahi [23]	CNN	88.8	88.8	88.5	99.1	99.1	99.1
	SAE	86.6	88.8	87.3	97.8	96.3	97
This paper	ICLSTM	96.6	97.6	97	98.5	97.8	98

Table 7. Performance for each type of encrypted traffic service identification.

Class	ICLSTM			SAE [23]		
	Precision	Recall	F1	Precision	Recall	F1
Chat	99.2	98.6	98.9	82	68	74
Email	98.2	99.3	98.7	97	93	95
FileTransfer	93.4	95.3	94.3	98	99	99
P2P	96.3	98.6	97.4	97	99	98
Streaming	95.6	99	97.3	82	84	83
VoIP	96.6	95	95.8	64	90	75
VPN-Chat	99.3	99.5	99.4	95	94	94
VPN-Email	98.5	91.2	94.7	97	93	95
VPN-FileTransfer	97.4	99.2	98.3	98	95	97
VPN-P2P	97.5	96.7	97	99	97	98
VPN-Streaming	98.5	99.8	99	99	99	99
VPN-VoIP	99.9	100	99.9	99	100	99
macro avg	97.5	97.7	97.6	N/A	N/A	N/A
weighted avg	96	95.9	96	92	92	92

5. Conclusions and Future Work

In this paper, we propose an ICLSTM architecture to implement encrypted traffic service identification. We utilize Inception and LSTM, which can simultaneously process the local spatial information of packets and the inter-packet timing information, enhance the characterization ability of traffic features. Firstly, these feature values are fused to obtain a one-dimensional feature vector, which are fed into the fully connected layer, and the probability distribution of the labels is output through the softmax layer for the purpose of classification. Then, the method used in this paper considers the problem of data imbalance, which assigns weights to different categories by introducing weight parameters, render the model pays more attention to small sample categories. In addition, we compare this paper with other traffic classification methods. The experimental results show that the accuracy of ICLSTM can achieve more than 98% for each identification task. The precision, recall and F1-score of mixed encrypted traffic service identification, regular encrypted traffic service identification and VPN encrypted traffic service identification all reach 97%, which are better than other methods. Our method does not need to rely on feature engineering, and has better identification capability in encrypted traffic service identification. The experimental results illustrate that the deep learning method is superior to traditional methods in traffic classification and will be the core of future work. In future work, we plan to propose new methods to solve the problem of data imbalance as well as more lightweight methods to make encrypted traffic classification easier and make our proposed methods more suitable for real-time encrypted traffic service identification, which can be applied to practical scenarios.

Author Contributions: Conceptualization, B.L. and N.L.; methodology, B.L.; software, B.L.; validation, B.L., C.D. and W.Z.; formal analysis, B.L.; investigation, C.D.; resources, B.L.; data curation, W.Z.; writing—original draft preparation, B.L.; writing—review and editing, B.L.; visualization, C.D. and W.Z.; supervision, N.L.; project administration, N.L.; funding acquisition, N.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the Innovation Environment Construction Special Project of Xinjiang Uygur Autonomous Region under Grant PT1811, and in part by the Key grant Project of The National Social Science Fund of China under Grant 20&ZD293.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors would like to thank the anonymous reviewers for their contribution to this peper.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

MLP	Muti-Layer Perception
CNN	Convolutional Neural Network
SAE	Staked Autoencoder
LSTM	Long short-term Memory Network
NN	Neural Networks
TP	True Positives
TN	True Negatives
FP	False Positives
FN	False Negatives

References

1. Cisco Encrypted Traffic Analytics 2019. Available online: https://www.cisco.com/c/en/us/solutions/collateral/enterprise-networks/enterprise-network-security/nb-09-encrytd-traf-anlytcs-wp-cte-en.html (accessed on 10 February 2021).
2. Soleymanpour, S.; Sadr, H.; Beheshti, H. An Efficient Deep Learning Method for Encrypted Traffic Classification on the Web. In Proceedings of the 2020 6th International Conference on Web Research (ICWR), Tehran, Iran, 22–23 April 2020; pp. 209–216. [CrossRef]
3. Wang, W.; Zhu, M.; Zeng, X.; Ye, X.; Sheng, Y. Malware traffic classification using convolutional neural network for representation learning. In Proceedings of the 2017 International Conference on Information Networking (ICOIN), Da Nang, Vietnam, 11–13 January 2017; pp. 712–717. [CrossRef]
4. Javaid, A.Y.; Niyaz, Q.; Sun, W.; Alam, M. A Deep Learning Approach for Network Intrusion Detection System. *EAI Endorsed Trans. Security Safety* **2016**, *3*, e2. [CrossRef]
5. Vu, L.; Thuy, H.V.; Nguyen, Q.U.; Ngoc, T.N.; Nguyen, D.N.; Hoang, D.T.; Dutkiewicz, E. Time Series Analysis for Encrypted Traffic Classification: A Deep Learning Approach. In Proceedings of the 2018 18th International Symposium on Communications and Information Technologies (ISCIT), Bangkok, Thailand, 26–29 September 2018; pp. 121–126. [CrossRef]
6. Anderson, B.; McGrew, D.A. Identifying Encrypted Malware Traffic with Contextual Flow Data. In *Proceedings of the 2016 ACM Workshop on Artificial Intelligence and Security, AISec@CCS 2016, Vienna, Austria, 28 October 2016*; Freeman, D.M., Mitrokotsa, A., Sinha, A., Eds.; ACM: New York, NY, USA, 2016; pp. 35–46. [CrossRef]
7. Al-Obaidy, F.; Momtahen, S.; Hossain, M.F.; Mohammadi, F.A. Encrypted Traffic Classification Based ML for Identifying Different Social Media Applications. In Proceedings of the 2019 IEEE Canadian Conference of Electrical and Computer Engineering, CCECE 2019, Edmonton, AB, Canada, 5–8 May 2019; pp. 1–5. [CrossRef]
8. Alshammari, R.; Zincir-Heywood, A.N. Can encrypted traffic be identified without port numbers, IP addresses and payload inspection? *Comput. Netw.* **2011**, *55*, 1326–1350. [CrossRef]
9. Wang, P.; Li, S.; Ye, F.; Wang, Z.; Zhang, M. PacketCGAN: Exploratory Study of Class Imbalance for Encrypted Traffic Classification Using CGAN. In Proceedings of the 2020 IEEE International Conference on Communications, ICC 2020, Dublin, Ireland, 7–11 June 2020; pp. 1–7. [CrossRef]
10. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.E.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, 7–12 June 2015; pp. 1–9. [CrossRef]
11. Pimenta Rodrigues, G.A.; De Oliveira Albuquerque, R.; Gomes de Deus, F.E.; De Sousa, R.T., Jr.; De Oliveira Júnior, G.A.; García Villalba, L.J.; Kim, T.-H. Cybersecurity and Network Forensics: Analysis of Malicious Traffic towards a Honeynet with Deep Packet Inspection. *Appl. Sci.* **2017**, *7*, 1082. [CrossRef]
12. Ning, J.; Poh, G.S.; Loh, J.; Chia, J.; Chang, E.-C. PrivDPI: Privacy-Preserving Encrypted Traffic Inspection with Reusable Obfuscated Rules. In Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (CCS'19), London, UK, 11–15 November 2019; Association for Computing Machinery: New York, NY, USA, 2019; pp. 1657–1670. [CrossRef]
13. Velan, P.; Cermák, M.; Celeda, P.; Drasar, M. A survey of methods for encrypted traffic classification and analysis. *Int. J. Netw. Manag.* **2015**, *25*, 355–374. [CrossRef]
14. Yao, Z.; Ge, J.; Wu, Y.; Lin, X.; He, R.; Ma, Y. Encrypted traffic classification based on Gaussian mixture models and Hidden Markov Models. *J. Netw. Comput. Appl.* **2020**, *166*, 102711. [CrossRef]
15. Madhukar, A.; Williamson, C.L. A Longitudinal Study of P2P Traffic Classification. In Proceedings of the 14th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS 2006), Monterey, CA, USA, 11–14 September 2006; pp. 179–188. [CrossRef]
16. Lucia, M.J.D.; Cotton, C. Detection of Encrypted Malicious Network Traffic using Machine Learning. In Proceedings of the 2019 IEEE Military Communications Conference, MILCOM 2019, Norfolk, VA, USA, 12–14 November 2019; pp. 1–6. [CrossRef]
17. Draper-Gil, G.; Lashkari, A.H.; Mamun, M.S.I.; Ghorbani, A. Characterization of Encrypted and VPN Traffic using Time-related Features. In Proceedings of the 2nd International Conference on Information Systems Security and Privacy, ICISSP 2016, Rome, Italy, 19–21 February 2016; Camp, O., Furnell, S., Mori, P., Eds.; SciTePress: Setúbal, Portugal, 2016; pp. 407–414. [CrossRef]
18. Bhatia, M.; Sharma, V.; Singh, P.; Masud, M. Multi-Level P2P Traffic Classification Using Heuristic and Statistical-Based Techniques: A Hybrid Approach. *Symmetry* **2020**, *12*, 2117. [CrossRef]
19. Ma, C.; Du, X.; Cao, L. Improved KNN Algorithm for Fine-Grained Classification of Encrypted Network Flow. *Electronics* **2020**, *9*, 324. [CrossRef]
20. de Toledo, T.R.; Torrisi, N.M. Encrypted DNP3 Traffic Classification Using Supervised Machine Learning Algorithms. *Mach. Learn. Knowl. Extr.* **2019**, *1*, 384–399. [CrossRef]
21. Anderson, B.; Paul, S.; McGrew, D.A. Deciphering malware's use of TLS (without decryption). *J. Comput. Virol. Hacking Tech.* **2018**, *14*, 195–211. [CrossRef]
22. Wang, W.; Zhu, M.; Wang, J.; Zeng, X.; Yang, Z. End-to-end encrypted traffic classification with one-dimensional convolution neural networks. In Proceedings of the 2017 IEEE International Conference on Intelligence and Security Informatics (ISI), Beijing, China, 22–24 July 2017; pp. 43–48. [CrossRef]
23. Lotfollahi, M.; Siavoshani, M.J.; Zade, R.S.H.; Saberian, M. Deep packet: A novel approach for encrypted traffic classification using deep learning. *Soft Comput.* **2020**, *24*, 1999–2012. [CrossRef]

24. Zou, Z.; Ge, J.; Zheng, H.; Wu, Y.; Han, C.; Yao, Z. Encrypted Traffic Classification with a Convolutional Long Short-Term Memory Neural Network. In Proceedings of the 20th IEEE International Conference on High Performance Computing and Communications; 16th IEEE International Conference on Smart City; 4th IEEE International Conference on Data Science and Systems, HPCC/SmartCity/DSS 2018, Exeter, UK, 28–30 June 2018; pp. 329–334. [CrossRef]
25. van Roosmalen, J.; Vranken, H.P.E.; van Eekelen, M.C.J.D. Applying deep learning on packet flows for botnet detection. In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing, SAC 2018, Pau, France, 9–13 April 2018*; Haddad, H.M., Wainwright, R.L., Chbeir, R., Eds.; ACM: New York, NY, USA, 2018; pp. 1629–1636. [CrossRef]
26. Dong, C.; Zhang, C.; Lu, Z.; Liu, B.; Jiang, B. CETAnalytics: Comprehensive effective traffic information analytics for encrypted traffic classification. *Comput. Netw.* **2020**, *176*, 107258. [CrossRef]
27. Xu, L.; Dou, D.; Chao, H.J. ETCNet: Encrypted Traffic Classification Using Siamese Convolutional Networks. In Proceedings of the Workshop on Network Application Integration/CoDesign (NAI'20), Virtual Event, New York, NY, USA, 14 August 2020; ACM: New York, NY, USA, 2020; p. 3. [CrossRef]
28. SplitCap. Available online: https://www.netresec.com/index.ashx?page=SplitCap (accessed on 20 September 2020).
29. Scikitlearn. Available online: https://www.cntofu.com/book/170/docs/5.md (accessed on 5 October 2020).
30. Branson, S.; Horn, G.V.; Belongie, S.J.; Perona, P. Bird Species Categorization Using Pose Normalized Deep Convolutional Nets. *arXiv* **2014**, arXiv:1406.2952.
31. Wang, Z.; Wang, X.; Wang, G. Learning fine-grained features via a CNN Tree for Large-scale Classification. *Neurocomputing* **2018**, *275*, 1231–1240. [CrossRef]
32. Gu, J.; Wang, Z.; Kuen, J.; Ma, L.; Shahroudy, A.; Shuai, B.; Liu, T.; Wang, X.; Wang, G. Recent Advances in Convolutional Neural Networks. *arXiv* **2015**, arXiv:1512.07108,
33. Wang, P.; Ye, F.; Chen, X.; Qian, Y. Datanet: Deep Learning Based Encrypted Network Traffic Classification in SDN Home Gateway. *IEEE Access* **2018**, *6*, 55380–55391. [CrossRef]
34. Siami-Namini, S.; Tavakoli, N.; Namin, A.S. The Performance of LSTM and BiLSTM in Forecasting Time Series. In Proceedings of the 2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, 9–12 December 2019; pp. 3285–3292. [CrossRef]
35. Huang, Q.; Chen, R.; Zheng, X.; Dong, Z. Deep Sentiment Representation Based on CNN and LSTM. In Proceedings of the 2017 International Conference on Green Informatics (ICGI), Fuzhou, China, 15–17 August 2017; pp. 30–33. [CrossRef]
36. Luan, Y.; Lin, S. Research on Text Classification Based on CNN and LSTM. In Proceedings of the 2019 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA), Dalian, China, 29–31 March 2019; pp. 352–355. [CrossRef]
37. Li, C.; Zhan, G.; Li, Z. News Text Classification Based on Improved Bi-LSTM-CNN. In Proceedings of the 2018 9th International Conference on Information Technology in Medicine and Education (ITME), Hangzhou, China, 19–21 October 2018; pp. 890–893. [CrossRef]
38. Ioffe, S.; Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *arXiv* **2015**, arXiv:1502.03167.
39. Song, M.; Ran, J.; Li, S. Encrypted Traffic Classification Based on Text Convolution Neural Networks. In Proceedings of the 2019 IEEE 7th International Conference on Computer Science and Network Technology (ICCSNT), Dalian, China, 19–20 October 2019; pp. 432–436. [CrossRef]